亚马逊云科技

用户指南

Amazon Relational Database Service

Amazon Relational Database Service: 用户指南

Amazon Web Services 文档中描述的 Amazon Web Services 服务或功能可能因区域而异。要查看适用于中国区域的差异，请参阅 中国的 Amazon Web Services 服务入门 (PDF)。

Table of Contents

什么是 Amazon Relational Database Service (Amazon RDS)？

Amazon Relational Database Service (Amazon RDS) 是一项 Web 服务，让用户能够在 Amazon Web Services 云 云中更轻松地设置、操作和扩展关系数据库。可为用户提供一个经济有效、容量可调的符合行业标准的关系数据库，并承担常见的数据库管理任务。

> Note
>
> 本指南涵盖 Amazon Aurora 以外的 Amazon RDS 数据库引擎。有关使用 Amazon Aurora 的更多信息，请参阅 Amazon Aurora 用户指南。

如果不熟悉 Amazon 产品和服务，请使用以下资源了解更多信息：

- 有关所有 Amazon 产品的概述，请参阅什么是云计算？
- Amazon Web Services 提供了丰富的数据库服务。要详细了解 Amazon 上提供的各种数据库选项，请参阅选择 Amazon 数据库服务和在 Amazon 上运行数据库。

Amazon RDS 的优势

Amazon RDS 是一项托管式数据库服务。它负责大多数管理任务。通过消除繁琐的手动过程，Amazon RDS 使您可以专注于应用程序和用户。

与非完全托管式数据库部署相比，Amazon RDS 具有以下主要优势：

- 您可以使用您已熟悉的数据库引擎：IBM Db2、MariaDB、Microsoft SQL Server、MySQL、Oracle Database 和 PostgreSQL。
- Amazon RDS 可以管理备份、软件修补、自动故障检测和恢复。
- 您可以开启自动备份，也可以手动创建您自己的备份快照。您可以使用这些备份还原数据库。Amazon RDS 还原过程将可靠且高效地工作。
- 可以通过主数据库实例和在发生问题时可向其执行失效转移操作的同步辅助数据库实例来实现高可用性。您还可以使用只读副本扩展读取。
- 除了保证数据库包的安全外，还可以使用 Amazon Identity and Access Management（IAM）定义用户和权限来控制访问权限。您还可以将数据库置于 Virtual Private Cloud (VPC) 中，这样有助于保护数据库。

与 Amazon EC2 和本地部署的职责比较

我们建议使用 Amazon RDS 作为大多数关系数据库部署的默认选择。以下替代方法的缺点是会让您花更多的时间来管理软件和硬件：

本地部署

购买本地服务器时，您会一并获得 CPU、内存、存储和 IOPS。您对服务器、操作系统和数据库软件承担全部责任。

Amazon EC2

Amazon Elastic Compute Cloud (Amazon EC2) 在 Amazon Web Services 云 中提供可扩展的计算容量。与本地服务器不同，CPU、内存、存储和 IOPS 是分开的，因此您可以独立扩展它们。Amazon 管理硬件层，这消除了管理本地数据库服务器的一些负担。

在 Amazon EC2 上运行数据库的缺点是更容易出现用户错误。例如，手动更新操作系统或数据库软件时，可能会意外导致应用程序停机。您可能会花几个小时来检查每一项更改，以识别和修复问题。

下表比较了本地数据库、Amazon EC2 和 Amazon RDS 的管理模型。

功能	本地管理	Amazon EC2 管理	Amazon RDS 管理
应用程序优化	客户	客户	客户
扩展	客户	客户	Amazon
高可用性	客户	客户	Amazon
数据库备份	客户	客户	Amazon
数据库软件修补	客户	客户	Amazon
数据库软件安装	客户	客户	Amazon
操作系统 (OS) 修补	客户	客户	Amazon
操作系统安装	客户	客户	Amazon
服务器维护	客户	Amazon	Amazon

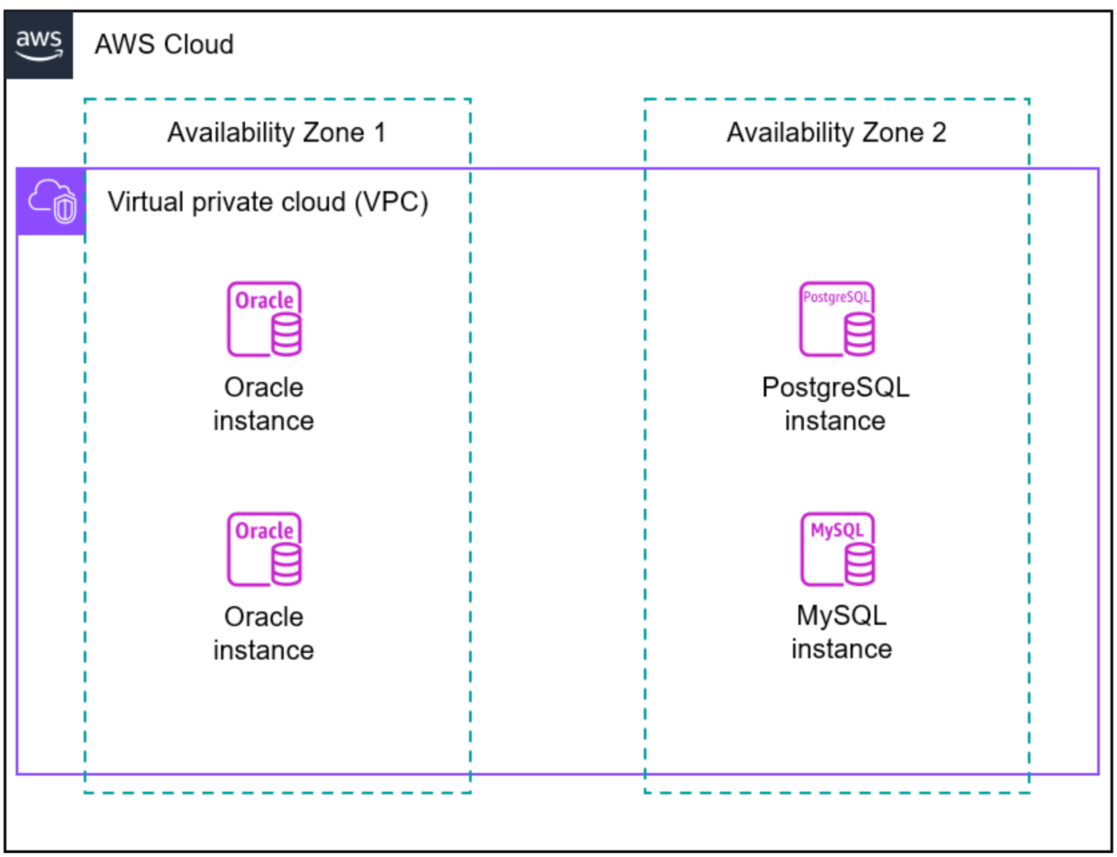
功能	本地管理	Amazon EC2 管理	Amazon RDS 管理
硬件生命周期	客户	Amazon	Amazon
电源、网络和冷却	客户	Amazon	Amazon

Amazon RDS 责任共担模式

Amazon RDS 负责托管数据库实例和数据库集群的软件组件和基础设施。您负责查询优化，这是调整 SQL 查询以提高性能的过程。查询性能高度依赖于数据库设计、数据大小、数据分布、应用程序工作负载和查询模式，这些可能会有很大差异。监控和优化是您自己的 RDS 数据库所拥有的高度个性化流程。您可以使用 Amazon RDS 性能详情和其他工具来识别有问题的查询。

Amazon RDS 数据库实例

数据库实例 是在 Amazon Web Services 云 中运行的独立数据库环境。Amazon RDS 的基本构建基块是数据库实例。您的数据库实例可以包含一个或多个由用户创建的数据库。下图显示了包含两个可用区的虚拟私有云（VPC），每个可用区包含两个数据库实例。

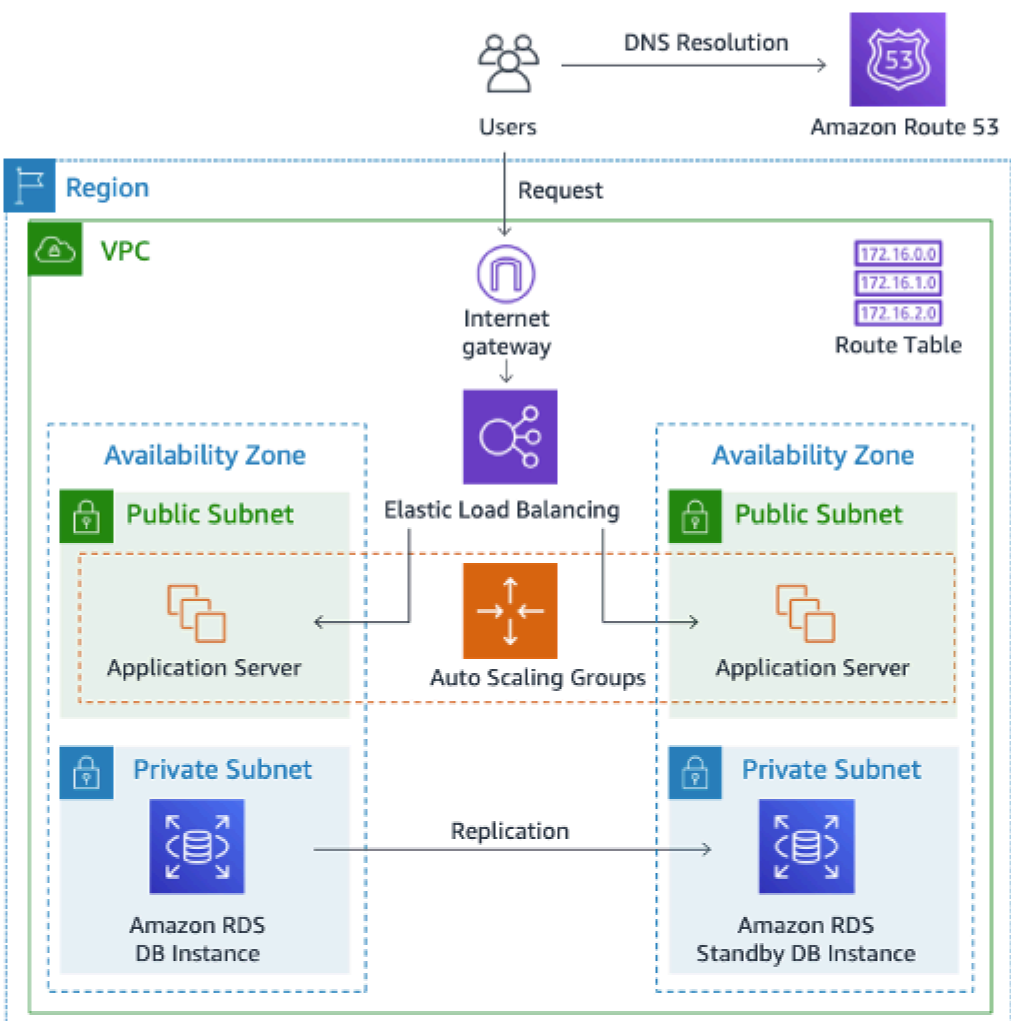


可以使用与处理独立数据库实例时所用的相同工具和应用程序来访问您的数据库实例。您可以使用 Amazon Command Line Interface（Amazon CLI）、Amazon RDS API 或 Amazon Web Services Management Console创建和修改数据库实例。

主题

Amazon RDS 应用程序架构：示例

下图显示了将 Amazon RDS 数据库实例用于数据库存储的动态网站的典型用例：



上述架构的主要组件如下所述：

Elastic Load Balancing

Amazon 通过弹性负载均衡来路由用户流量。负载均衡器跨多个计算资源 (如虚拟服务器) 分布工作负载。在此示例用例中，弹性负载均衡器将客户端请求转发到应用程序服务器。

应用程序服务器

应用程序服务器与 RDS 数据库实例进行交互。Amazon 中的应用程序服务器通常托管在 EC2 实例上，这些实例提供可扩展的计算容量。应用程序服务器位于同一虚拟私有云（VPC）内具有不同可用区（AZ）的公有子网中。

RDS 数据库实例

EC2 应用程序服务器与 RDS 数据库实例进行交互。数据库实例位于同一虚拟私有云（VPC）内不同可用区（AZ）的私有子网中。由于子网是私有的，因此不允许来自互联网的请求。

主数据库实例会复制到另一个称为只读副本的数据库实例。这两个数据库实例都位于 VPC 内的私有子网中，这意味着互联网用户无法直接访问它们。

数据库引擎

数据库引擎是在数据库实例上运行的特定关系数据库软件。Amazon RDS 支持以下数据库引擎：

- IBM Db2

  有关更多信息，请参阅 Amazon RDS for Db2。
- MariaDB

  有关更多信息，请参阅 Amazon RDS for MariaDB。
- Microsoft SQL Server

  有关更多信息，请参阅 Amazon RDS for Microsoft SQL Server。
- MySQL

  有关更多信息，请参阅 Amazon RDS for MySQL。
- Oracle Database

  有关更多信息，请参阅 Amazon RDS for Oracle。

- PostgreSQL

  有关更多信息，请参阅 Amazon RDS for PostgreSQL。

每个数据库引擎均有自己支持的功能，并且每个版本的数据库引擎可能包括特定的功能。对 Amazon RDS 功能的支持因 Amazon Web Services 区域和每个数据库引擎的特定版本而异。要查看不同引擎版本和区域的功能支持，请参阅 Amazon RDS 中受 Amazon Web Services 区域和数据库引擎支持的功能。

此外，每个数据库引擎在数据库参数组中均有一组参数，用于控制其管理的数据库的行为。有关参数组的更多信息，请参阅 Amazon RDS 的参数组。

数据库实例类

数据库实例类确定数据库实例的计算和内存容量。数据库实例类由数据库实例类类型和大小共同组成。Amazon RDS 支持以下实例类类型，其中星号（*）表示代别、可选属性和大小：

- 通用型 – db.m*
- 内存优化型 – db.z*、db.x*、db.r*
- 计算优化型 – db.c*
- 可突增性能 – db.t*

每种实例类提供不同的计算、内存和存储功能。例如，db.m7g 是由 Amazon Graviton3 处理器提供支持的第 7 代通用型数据库实例类类型。创建数据库实例时，可以指定数据库实例类，例如 db.m7g.2xlarge，其中 2xlarge 是大小。有关不同实例类的硬件规格的更多信息，请参阅数据库实例类的硬件规格。

您可以选择最能满足您需求的数据库实例类。如果您的需求随时间推移而变化，则可以更改数据库实例类。例如，可以将 db.m7g.2xlarge 实例纵向扩展到 db.m7g.4xlarge。有关更多信息，请参阅 数据库实例类。

> **Note**
>
> 有关数据库实例类的定价信息，请查看 Amazon RDS 产品页的定价部分。

数据库实例存储

Amazon EBS 提供块级别的持久存储卷，您可将这些卷附加到正在运行的实例。数据库实例存储有以下类型：

- 通用型 (SSD)

  这种经济高效的存储类型非常适合在中等规模的数据库实例上运行的各种工作负载。通用型存储最适合开发和测试环境。
- 预置 IOPS (PIOPS)

  此存储类型旨在满足 I/O 密集型工作负载（尤其是数据库工作负载）的需求，此类工作负载需要低 I/O 延迟和一致的 I/O 吞吐量。预调配 IOPS 存储最适合生产环境。
- 磁介质

  Amazon RDS 支持磁性存储以实现向后兼容。建议您采用通用型 SSD 或预调配 IOPS SSD 来满足所有新存储需求。

存储类型在性能特点和价格上有所不同。您可以根据数据库的需求定制存储性能和成本。

每个数据库实例都具有最小和最大存储要求，具体取决于存储类型和其支持的数据库引擎。重要的是要有足够的存储空间，这样数据库才有增长的空间。此外，足够的存储空间可确保数据库引擎的功能有空间写入内容或日志条目。有关更多信息，请参阅 Amazon RDS 数据库实例存储。

Amazon Virtual Private Cloud（Amazon VPC）中的数据库实例

您可以使用 Amazon Virtual Private Cloud (Amazon VPC) 服务，在虚拟私有云 (VPC) 上运行数据库实例。使用 VPC 时，您的虚拟联网环境完全由您控制。您可以选择自己的 IP 地址范围、创建子网以及配置路由和访问控制列表。

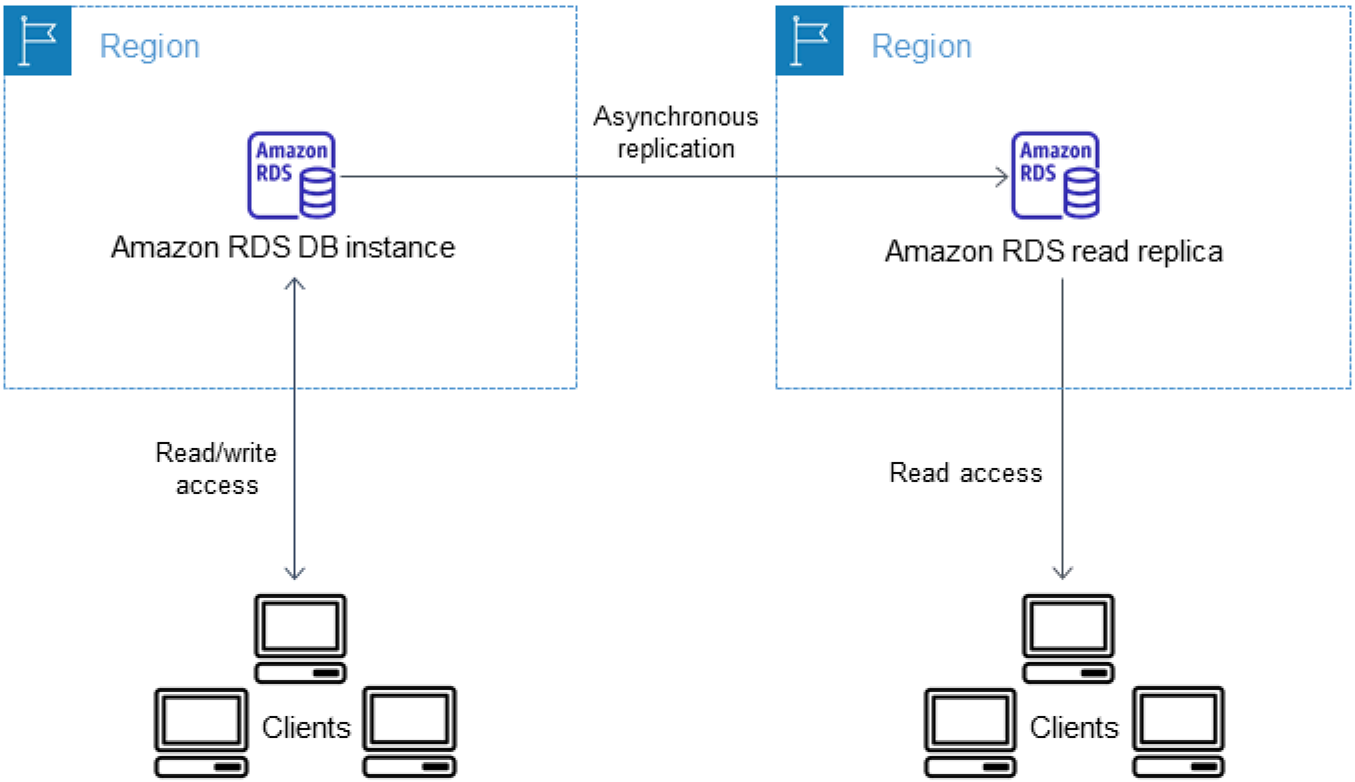
无论是否在 VPC 中运行，Amazon RDS 的基本功能都是相同的。Amazon RDS 可以管理备份、软件修补、自动故障检测和恢复。在 VPC 中运行数据库实例没有额外费用。有关将 Amazon VPC 与 RDS 结合使用的更多信息，请参阅Amazon VPC 和 Amazon RDS。

Amazon RDS 使用网络时间协议（NTP）来同步数据库实例上的时间。

Amazon Web Services 区域 和可用区

Amazon 云计算资源安置在世界不同地区 (例如：北美、欧洲或亚洲) 的高度可用的数据中心设施内。每个数据中心位置称为一个 Amazon Web Services 区域。借助 Amazon RDS，可以在多个区域中创建数据库实例。

以下场景显示了一个区域中的 RDS 数据库实例，该实例异步复制到另一个区域中的备用数据库实例。如果一个区域变为不可用，则另一个区域中的实例仍然可用。



可用区

每个 Amazon 区域包含很多称为可用区或 AZ 的不同位置。每个可用区设计为可隔离其他可用区的故障。每个可用区都设计为向同一 Amazon 区域中的其他可用区提供低成本、低延迟的网络连接。通过启动独立可用区内的数据库实例，可以保护应用程序不受单一位置故障的影响。有关更多信息，请参阅区域、可用区和 Local Zones。

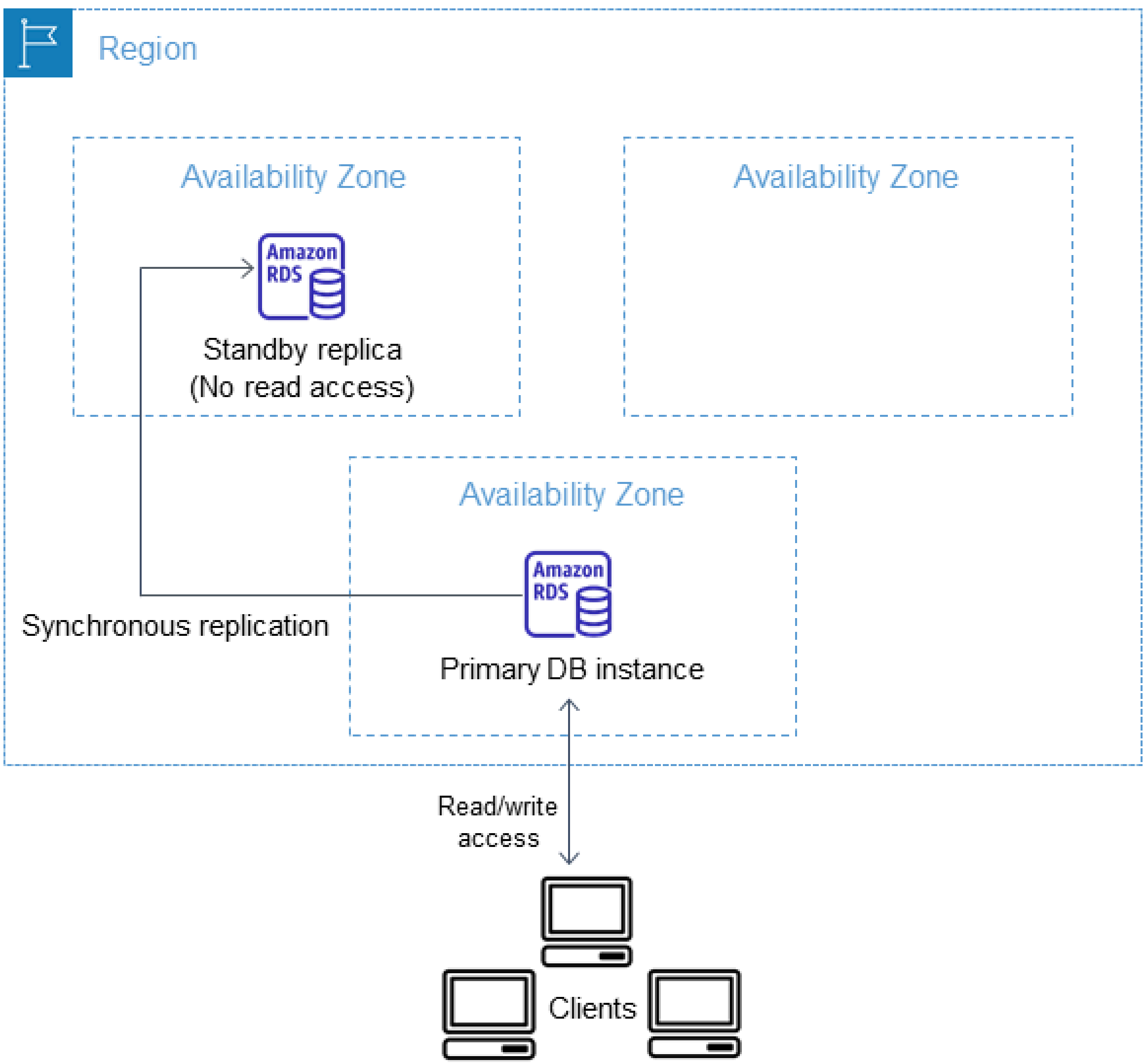
多可用区部署

您可以在多个可用区运行数据库实例，此选项称为多可用区部署。当您选择此选项时，Amazon 会自动在不同的可用区中预置和维护一个或多个辅助备用数据库实例。主数据库实例可跨可用区复制到每个备用数据库实例。

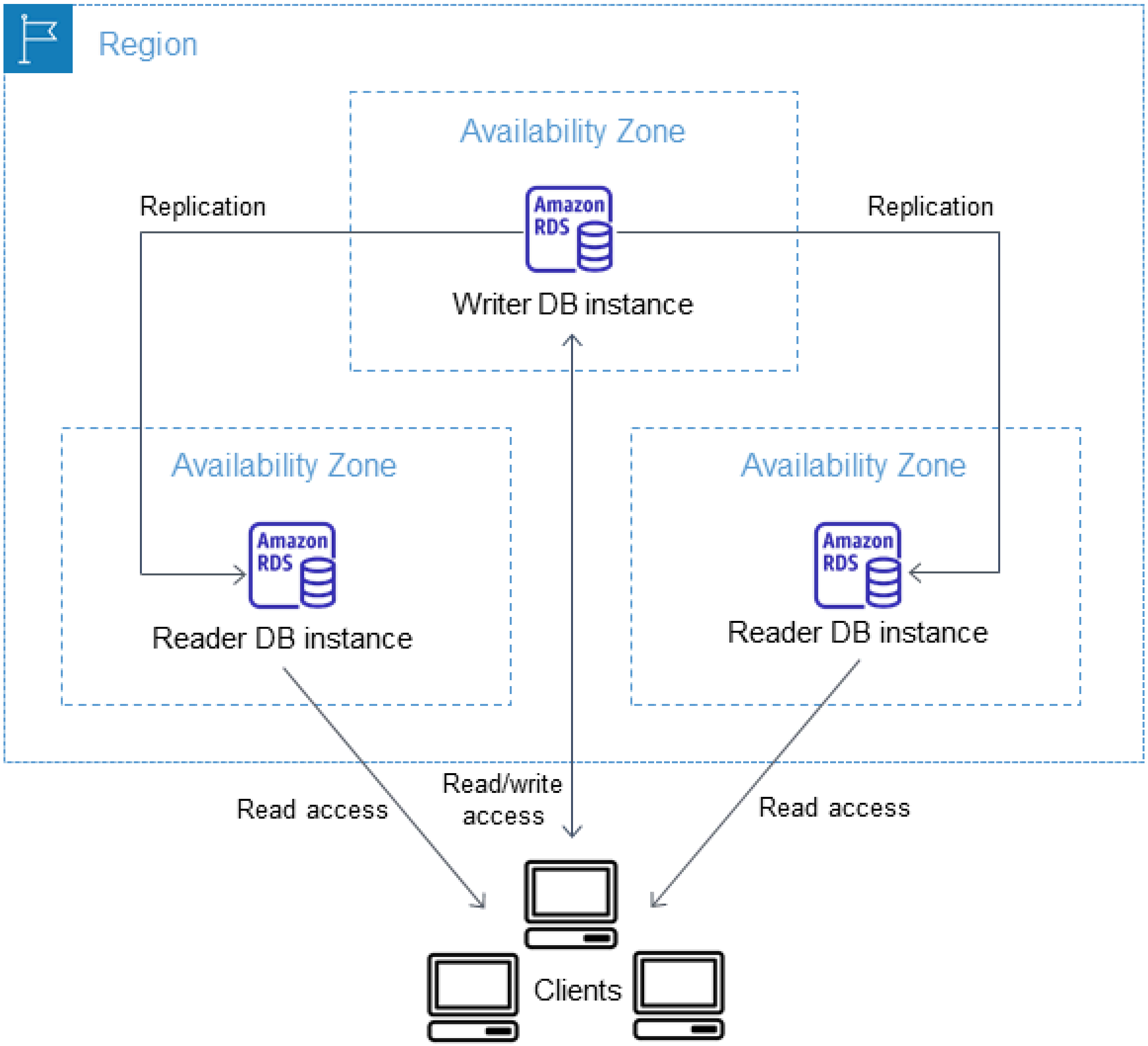
多可用区部署提供以下优势：

- 提供数据冗余和失效转移支持
- 消除 I/O 冻结
- 最大限度地减少系统备份期间的延迟峰值
- 在辅助数据库实例上提供读取流量（仅限多可用区数据库集群部署）

下图描绘了一个多可用区数据库实例部署，其中 Amazon RDS 会自动在不同可用区中预置和维护一个同步备用副本。副本数据库不提供读取流量。



下图描绘了一个多可用区数据库集群部署，此部署在同一个 Amazon Web Services 区域的三个独立可用区中有一个写入器数据库实例和两个读取器数据库实例。所有三个数据库实例都可以提供读取流量。

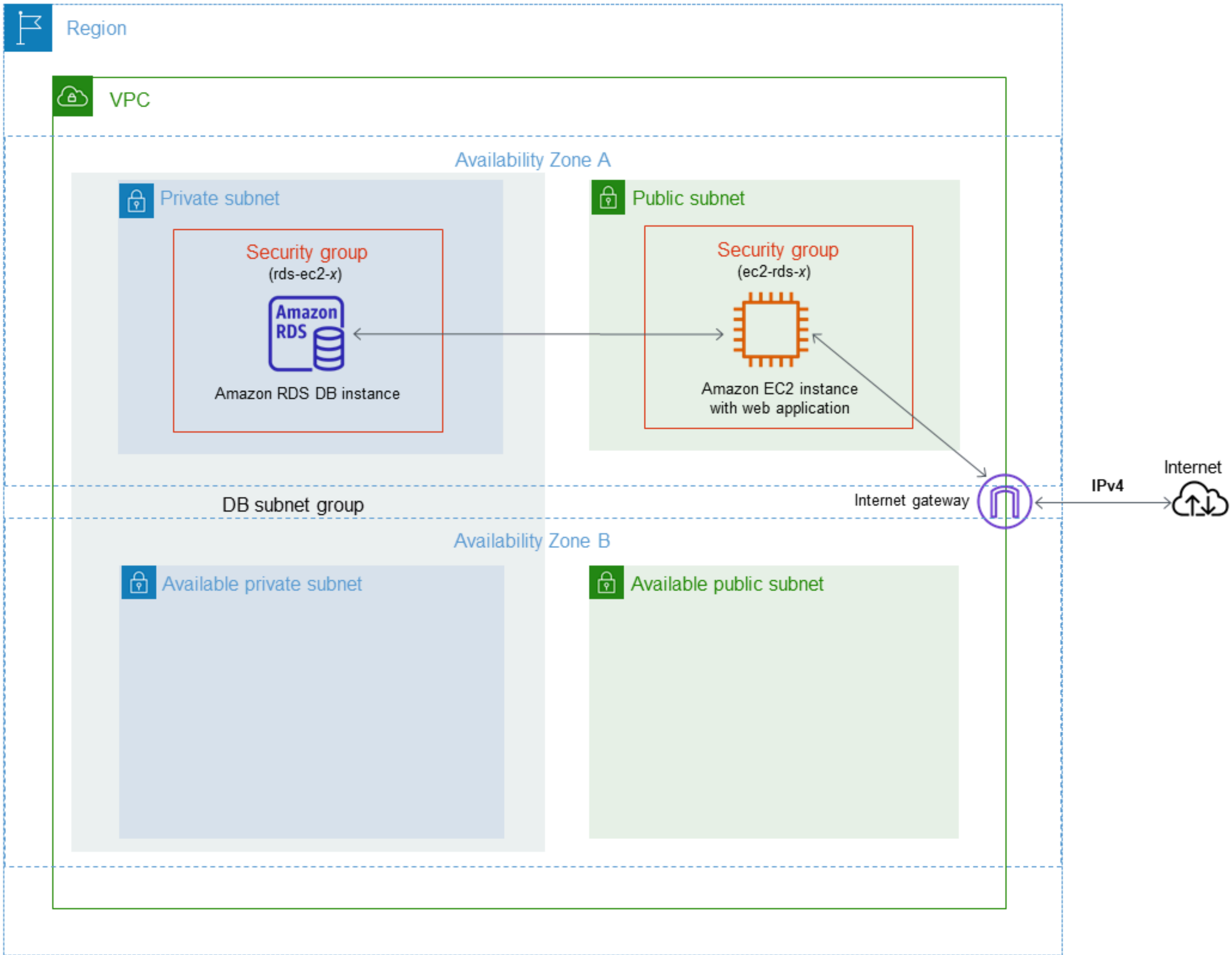


有关更多信息，请参阅 [配置和管理 Amazon RDS 的多可用区部署]。

使用安全组控制访问权限

安全组 通过支持访问您指定的 IP 地址范围或 Amazon EC2 实例，控制对数据库实例的访问权限。您可以将安全组应用于一个或多个数据库实例。

VPC 中数据库实例的一种常见用途是与同一 VPC 中的应用程序服务器共享数据。以下示例使用 VPC 安全组 `ec2-rds-x` 来定义使用客户端应用程序的 IP 地址作为源的入站规则。应用程序服务器属于此安全组。名为 `rds-ec2-x` 的第二个安全组指定 `ec2-rds-x` 作为源并连接到 RDS 数据库实例。根据安全组规则，客户端应用程序不能直接访问数据库实例，但 EC2 实例可以访问数据库实例。



有关安全组的更多信息，请参阅 Amazon RDS 中的安全性。

Amazon RDS 监控

监控是保持 Amazon RDS 和其它 Amazon 解决方案的可靠性、可用性和性能的重要方面。Amazon 提供各种监控工具来监控 Amazon RDS、在出现错误时进行报告并适时自动采取措施。

可以使用各种自动和手动工具跟踪数据库实例的性能和运行状况：

Amazon RDS 数据库实例状态和建议

通过使用 Amazon RDS 控制台、Amazon CLI 或 RDS API，查看有关实例的当前状态的详细信息。还可以对有关数据库资源（例如数据库实例、只读副本和数据库参数组）的自动建议做出响应。有关更多信息，请参阅 来自 Amazon RDS 的建议。

Amazon RDS 的 Amazon CloudWatch 指标

您可以使用 Amazon CloudWatch 服务监控数据库实例的性能和运行状况。在 Amazon RDS 控制台中显示 CloudWatch 的性能图表。Amazon RDS 每分钟自动向 CloudWatch 发送一次每个活动数据库的指标。对于 CloudWatch 中的 Amazon RDS 指标，您不会获得额外费用。

使用 Amazon CloudWatch 警报，可以在特定时间段内监控单个 Amazon RDS 指标。然后，您可以根据相对于您设置的阈值的指标值来执行一个或多个操作。有关更多信息，请参阅 使用 Amazon CloudWatch 监控 Amazon RDS 指标。

Amazon RDS 性能详情和操作系统监控

性能详情可评测数据库上的负载，并确定执行操作的时间和位置。有关更多信息，请参阅 在 Amazon RDS 上使用性能详情监控数据库负载。Amazon RDS 增强监控实时查看操作系统的指标。有关更多信息，请参阅 使用增强监控来监控操作系统指标。

集成 Amazon Web Services 服务

Amazon RDS 与 Amazon EventBridge、Amazon CloudWatch Logs 和 Amazon DevOps Guru 相集成。有关更多信息，请参阅 监控 Amazon RDS 实例中的指标。

Amazon RDS 的用户界面

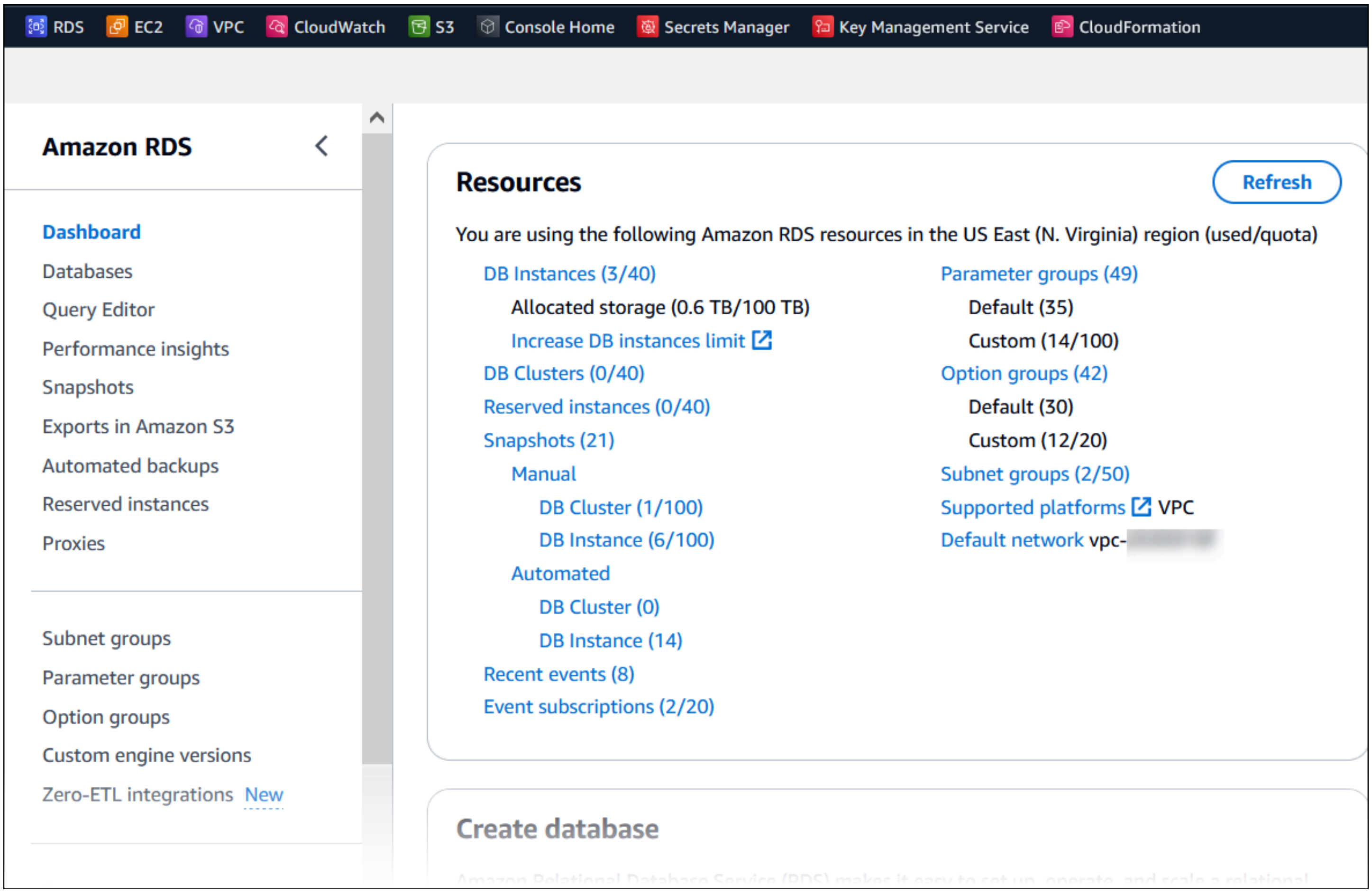
可以通过多种方式与 Amazon RDS 进行交互。

主题

Amazon Web Services Management Console

Amazon Web Services Management Console是一个基于 Web 的简单用户界面。您可以从控制台中管理数据库实例，而无需进行任何编程。要访问 Amazon RDS 控制台，请登录Amazon Web Services

Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。



命令行界面

您可以使用 Amazon Command Line Interface（Amazon CLI）以交互方式访问 Amazon RDS API。要安装 Amazon CLI，请参阅安装 Amazon 命令行界面。要开始使用适用于 RDS 的 Amazon CLI，请参阅适用于 Amazon RDS 的 Amazon Command Line Interface 参考。

Amazon RDS API

如果您是一名开发人员，则可以使用 API 以编程方式访问 Amazon RDS。有关更多信息，请参阅 Amazon RDS API 参考。

对于应用程序开发，我们建议使用 Amazon 软件开发工具包 (SDK) 之一。Amazon 软件开发工具包可处理低级详细信息（如身份验证、重试逻辑和错误处理），让您可以将注意力放在应用程序逻辑上。Amazon软件开发工具包适用于各种语言。有关更多信息，请参阅用于 Amazon Web Services 的工具。

Amazon还提供了库、示例代码、教程和其他资源来帮助您更轻松地入门。有关更多信息，请参阅示例代码和库。

Amazon RDS 如何收费

在使用 Amazon RDS 时，可以选择使用按需数据库实例或预留数据库实例。有关更多信息，请参阅"Amazon RDS 的数据库实例计费"。

有关 Amazon RDS 定价信息，请参阅 Amazon RDS 产品页。

接下来做什么？

上一节为您介绍了 RDS 提供的基本基础设施组件。您下一步该做什么？

开始使用

根据 开始使用 Amazon RDS 中的说明创建数据库实例。

特定于数据库引擎的主题

您可以在以下部分中查看特定数据库引擎的特定信息：

- Amazon RDS for Db2
- Amazon RDS for MariaDB
- Amazon RDS for Microsoft SQL Server
- Amazon RDS for MySQL
- Amazon RDS for Oracle
- Amazon RDS for PostgreSQL

Amazon RDS 数据库实例

数据库实例是在云中运行的独立数据库环境。它是 Amazon RDS 的基本构建块。一个数据库实例可以包含多个由用户创建的数据库，并且可以使用与访问独立数据库实例相同的工具和应用程序进行访问。使用 Amazon 命令行工具、Amazon RDS API 操作或 Amazon Web Services Management Console，可以轻松地创建和修改数据库实例。

> **Note**
>
> Amazon RDS 支持使用任何标准 SQL 客户端应用程序访问数据库。Amazon RDS 不支持直接访问主机，但使用 RDS Custom 除外。

您最多可以拥有 40 个 Amazon RDS 数据库实例，并且具有以下限制：

- 每个 SQL Server 版本（Enterprise、Standard、Web 和 Express）具有 10 个处于“附带许可”模式的实例
- Oracle 具有 10 个处于“附带许可”模式的实例
- Db2 具有 40 个处于“自带许可”（BYOL）许可模式的实例
- MySQL、MariaDB 或 PostgreSQL 具有 40 个实例
- Oracle 具有 40 个处于“自带许可”(BYOL) 许可模式的实例

> **Note**
>
> 如果您的应用程序需要更多数据库实例，则可以使用此表来请求其他数据库实例。

每个数据库实例都有一个数据库实例标识符。在与 Amazon RDS API 和 Amazon CLI 命令交互时，客户提供的名称可唯一标识数据库实例。对于 Amazon 区域中的客户而言，数据库实例标识符必须具有唯一性。

数据库实例标识符是 RDS 分配给实例的 DNS 主机名的一部分。例如，如果将 db1 指定为数据库实例标识符，则 RDS 将自动为实例分配 DNS 端点。示例端点为 `db1.abcdefghijkl.us-east-1.rds.amazonaws.com`，其中 `db1` 是实例 ID。

在示例端点 `db1.abcdefghijkl.us-east-1.rds.amazonaws.com` 中，字符串 `abcdefghijkl` 是 Amazon Web Services 区域和 Amazon Web Services 账户的特定组合的唯一标识符。示例中的

abcdefghijkl 标识符由 RDS 在内部生成，对于指定的区域和账户组合不会发生变化。因此，您在该区域中的所有数据库实例都共享相同的固定标识符。考虑固定标识符的以下特征：

- 如果您重命名数据库实例，则端点会有所不同，但固定标识符相同。例如，如果您将 *db1* 重命名为 *renamed-db1*，则新的实例端点为 *renamed-db1*.*abcdefghijkl*.*us-east-1*.rds.amazonaws.com。
- 如果您删除并重新创建具有相同数据库实例标识符的数据库实例，则端点相同。
- 如果您使用同一个账户在不同的区域创建数据库实例，则内部生成的标识符会有所不同，因为该区域不同，如在 *db2*.*mnopqrstuvwx*.*us-west-1*.rds.amazonaws.com 中。

每个数据库实例均支持数据库引擎。Amazon RDS 当前支持 Db2、MySQL、MariaDB、PostgreSQL、Oracle、Microsoft SQL Server 和 Amazon Aurora 数据库引擎。

在创建数据库实例时，某些数据库引擎需要指定数据库名称。一个数据库实例可承载多个数据库，或一个带有多个架构的 Oracle 数据库。数据库名称值取决于数据库引擎：

- 对于 Db2 数据库引擎，数据库名称是数据库实例中托管的数据库的名称。该字段是可选的。可以通过调用 `rdsadmin.create_database` 存储过程来创建数据库。有关更多信息，请参阅 the section called “创建数据库”。
- 对于 MySQL 和 MariaDB 数据库引擎，数据库名称是数据库实例中托管的数据库的名称。由同一个数据库实例托管的数据库在此实例内必须有一个唯一的名称。
- 对于 Oracle 数据库引擎，数据库名称用于设置 ORACLE_SID 值，连接 Oracle RDS 实例时必须提供此值。
- 对于 Microsoft SQL Server 数据库引擎，数据库名称不是一个受到支持的参数。
- 对于 PostgreSQL 数据库引擎，数据库名称是数据库实例中托管的数据库的名称。创建数据库实例时不需要数据库名称。由同一个数据库实例托管的数据库在此实例内必须有一个唯一的名称。

Amazon RDS 会为数据库实例创建一个主用户账户，这也是创建过程的一部分。此主用户具有创建数据库以及在主用户创建的表上执行创建、删除、选择、更新和插入操作的权限。创建数据库实例时，必须设置主用户密码，但可以使用 Amazon CLI、Amazon RDS API 操作或Amazon Web Services Management Console随时进行更改。还可以使用标准 SQL 命令更改主用户密码并管理用户。

> **Note**
>
> 该用户指南涵盖非 Aurora Amazon RDS 数据库引擎。有关使用 Amazon Aurora 的更多信息，请参阅 Amazon Aurora 用户指南。

数据库实例类

数据库实例类确定 Amazon RDS 数据库实例的计算和内存容量。您需要的数据库实例类取决于您的处理能力和内存要求。

数据库实例类由数据库实例类类型和大小共同组成。例如，db.r6g 是由 Amazon Graviton2 处理器提供支持的内存优化数据库实例类类型。在 db.r6g 实例类类型中，db.r6g.2xlarge 是数据库实例类。该类的大小为 2xlarge。

有关数据库实例类类型、支持的数据库引擎、支持的 Amazon Web Services 区域、更改数据库实例类、配置 RDS for Oracle 的处理器或数据库实例类的硬件规格的更多信息，请参阅以下各节。

主题

数据库实例类类型

Amazon RDS 对于以下使用案例支持数据库实例类：

- 通用型
- 内存优化
- 计算优化
- 可突增性能
- 优化型读取

有关 Amazon EC2 实例类型的更多信息，请参阅 Amazon EC2 文档中的实例类型。

通用实例类类型

以下通用数据库实例类可用：

- db.m8g –由 Amazon Graviton4 处理器提供支持的通用数据库实例类。这些实例类为各种不同的通用工作负载提供平衡的计算、内存和网络。与基于 Amazon Graviton3 的第七代 M7g 实例相比，这些新类提供了更大的实例大小，其 vCPU 和内存最多可增加 3 倍。

  您可以修改数据库实例，以使用由 Amazon Graviton4 处理器提供支持的其中一个数据库实例类。为此，请完成与修改任何其他数据库实例相同的步骤。

- db.m7i – 由第四代 Intel Xeon 可扩展处理器提供支持的通用数据库实例类。db.m7i 实例已通过 SAP 认证，非常适合支持需要更大的实例大小或较高持续 CPU 使用率的企业级应用程序。这些实例类为各种不同的通用工作负载提供平衡的计算、内存和网络。此实例类类型可提供高达 40,000 Mbps 的 EBS 带宽和高达 50 Gbps 的网络带宽。与 db.m6i 实例相比，db.m7i 实例的性价比最高可提高 15%。

  您可以修改数据库实例，以使用由 Amazon Graviton3 处理器提供支持的其中一个数据库实例类。为此，请完成与修改任何其他数据库实例相同的步骤。

- db.m7g –由 Amazon Graviton3 处理器提供支持的通用数据库实例类。这些实例类为各种不同的通用工作负载提供平衡的计算、内存和网络。

  您可以修改数据库实例，以使用由 Amazon Graviton3 处理器提供支持的其中一个数据库实例类。为此，请完成与修改任何其他数据库实例相同的步骤。

- db.m6g – 由 Amazon Graviton2 处理器提供支持的通用数据库实例类。这些实例为各种不同的通用工作负载提供平衡的计算、内存和网络。db.m6gd 实例类具有基于本地 NVMe 的 SSD 数据块级存储，适用于需要高速、低延迟本地存储的应用程序。

  您可以修改数据库实例，以使用由 Amazon Graviton2 处理器提供支持的其中一个数据库实例类。为此，请完成与修改任何其他数据库实例相同的步骤。

- db.m6i – 由第三代 Intel Xeon 可扩展处理器提供支持的通用数据库实例类。这些实例已通过 SAP 认证，非常适合如下工作负载：支持企业应用程序的后端服务器、游戏服务器、缓存实例集和应用程序开发环境等等。db.m6id 和 db.m6idn 实例类提供高达 7.6TB 的基于 NVMe 的本地 SSD 存储，而 db.m6in 提供仅 EBS 存储。db.m6in 和 db.m6idn 类提供高达 200Gbps 的网络带宽。
- db.m5 – 通用数据库实例类，它们兼顾了计算、内存和网络资源，是很多应用程序的理想之选。db.m5d 实例类提供与主机服务器物理连接的基于 NVMe 的 SSD 存储。db.m5 实例类提供比以前的 db.m4 实例类更多的计算容量。它们由 Amazon Nitro 系统（专用硬件和轻量级虚拟机监控程序的组合）提供支持。
- db.m4 – 通用数据库实例类，它们提供比以前的 db.m3 实例类更多的计算容量。

对于 RDS for Oracle 数据库引擎，Amazon RDS 不再支持 db.m4 数据库实例类。如果您之前创建了 RDS for Oracle db.m4 数据库实例类，则 Amazon RDS 会自动将这些数据库实例升级为等效的 db.m5 数据库实例类。

对于 RDS for MariaDB、RDS for MySQL、RDS for SQL Server 和 RDS for PostgreSQL 数据库引擎，Amazon RDS 已使用以下计划启动此数据库实例类的终止支持流程。对于使用此实例类的所有 RDS 数据库实例，建议您尽快升级到新一代的数据库实例类。

操作或建议	日期
从该日期开始，Amazon RDS 开始自动将使用 db.m4 的实例升级到新一代的 db.m5 实例类。不再支持使用 db.m4 实例类创建数据库实例。	2024 年 6 月 1 日
Amazon RDS 终止对 db.m4 的支持。	2024 年 12 月 31 日

- db.m3 – 通用数据库实例类，它们提供比以前的 db.m1 实例类更多的计算容量。

  对于 RDS for MariaDB、RDS for MySQL 和 RDS for PostgreSQL 数据库引擎，Amazon RDS 已使用以下计划启动 db.m3 数据库实例类的生命周期终止过程，其中包括升级建议。对于所有使用 db.m3 数据库实例类的 RDS 数据库实例，建议您尽快升级到更高代次的数据库实例类。

操作或建议	日期
您不能再创建使用 db.m3 数据库实例类的 RDS 数据库实例。	现在
Amazon RDS 启动了使用 db.m3 数据库实例类的 RDS 数据库实例的自动升级，以将其升级为等效的 db.m5 数据库实例类。	2023 年 2 月 1 日

内存优化型实例类类型

内存优化型 Z 系列支持以下实例类：

- db.z1d – 针对内存密集型应用程序进行了优化的实例类。这些实例类提供了高计算容量和高内存占用。高频率 z1d 实例可提供高达 4.0GHz 的持续所有核心频率。

内存优化型 X 系列支持以下实例类：

- db.x2g - 针对内存密集型应用程序进行优化的实例类，由 Amazon Graviton2 处理器提供支持。这些实例类可提供较低的每 GiB 内存成本。

  您可以修改数据库实例，以使用由 Amazon Graviton2 处理器提供支持的其中一个数据库实例类。为此，请完成与修改任何其他数据库实例相同的步骤。
- db.x2i – 针对内存密集型应用程序进行了优化的实例类。db.x2iedn 和 db.x2idn 实例类类型由第三代 Intel Xeon 可扩展处理器（Ice Lake）提供支持。它们包括高达 3.8TB 的本地 NVMe SSD 存储、高达 100Gbps 的网络带宽以及高达 4TiB（db.x2iden）或 2TiB（db.x2idn）的内存。db.x2iezn 类型由第二代 Intel Xeon 可扩展处理器（Cascade Lake）提供支持，全核睿频高达 4.5GHz，内存高达 1.5TiB。
- db.x1 – 针对内存密集型应用程序进行了优化的实例类。这些实例类在数据库实例类中提供了每 GiB RAM 的最低价格之一，以及多达 1,952GiB 基于 DRAM 的实例内存。db.x1e 实例类类型提供高达 3904GiB 的基于 DRAM 的实例内存。

内存优化型 R 系列支持以下实例类类型：

- db.r8g – 由 Amazon Graviton4 处理器提供支持的实例类。这些实例类非常适合在开源数据库（如 MySQL 和 PostgreSQL）中运行内存密集型工作负载。与基于 Amazon Graviton3 的第七代 db.r7g 实例相比，这些实例提供了更大的实例大小，其 vCPU 和内存最多可增加 3 倍。

  您可以修改数据库实例，以使用由 Amazon Graviton4 处理器提供支持的其中一个数据库实例类。为此，请完成与修改任何其他数据库实例相同的步骤。
- db.r7g – 由 Amazon Graviton3 处理器提供支持的实例类。这些实例类非常适合在开源数据库（如 MySQL 和 PostgreSQL）中运行内存密集型工作负载。

  您可以修改数据库实例，以使用由 Amazon Graviton3 处理器提供支持的其中一个数据库实例类。为此，请完成与修改任何其他数据库实例相同的步骤。
- db.r7i – 由第四代 Intel Xeon 可扩展处理器提供支持的实例类 这些实例类已通过 SAP 认证，非常适合在开源数据库（如 MySQL 和 PostgreSQL）中运行内存密集型工作负载。您可以修改数据库实例，以使用由第四代 Intel Xeon 可扩展处理器提供支持的其中一个数据库实例类。为此，请完成与修改任何其他数据库实例相同的步骤。
- db.r6g – 由 Amazon Graviton2 处理器提供支持的实例类。这些实例类非常适合在开源数据库（如 MySQL 和 PostgreSQL）中运行内存密集型工作负载。db.r6gd 类型提供基于本地 NVMe 的 SSD 数据块级存储，适用于需要高速、低延迟本地存储的应用程序。

您可以修改数据库实例，以使用由 Amazon Graviton2 处理器提供支持的其中一个数据库实例类。为此，请完成与修改任何其他数据库实例相同的步骤。

- db.r6i – 由第三代 Intel Xeon 可扩展处理器提供支持的实例类 这些实例类已通过 SAP 认证，非常适合在开源数据库（如 MySQL 和 PostgreSQL）中运行内存密集型工作负载。db.r6id、db.r6in 和 db.r6idn 实例类的内存与 vCPU 比率为 8:1，最大内存为 1TiB。db.r6id 和 db.r6idn 类提供高达 7.6TB 的基于 NVMe 的直接连接 SSD 存储，而 db.r6in 提供仅 EBS 存储。db.r6idn 和 db.r6in 类提供高达 200Gbps 的网络带宽。
- db.r5b – 针对吞吐量密集型应用程序进行了内存优化的实例类。db.r5b 实例由 Amazon Nitro 系统提供支持，可提供高达 60Gbps 的带宽和 260,000 IOPS 的 EBS 性能。这是 EC2 上最快的数据块存储性能。
- db.r5d – 针对低延迟、极高随机 I/O 性能和高速连续读取吞吐量进行了优化的实例类。
- db.r5 – 针对内存密集型应用程序进行了优化的实例类。这些实例类提供了改进的联网性能。它们由 Amazon Nitro 系统（专用硬件和轻量级虚拟机监控程序的组合）提供支持。
- db.r4 – 与以前的 db.r3 实例类相比，此实例类提供了改进的网络功能。

对于 RDS for Oracle 数据库引擎，Amazon RDS 已使用以下计划启动 db.r4 数据库实例类的生命周期终止过程，其中包括升级建议。对于使用 db.r4 实例类的 RDS for Oracle 数据库实例，建议您尽快升级到更高代次的实例类。

操作或建议	日期
您不再能够创建使用 db.r4 数据库实例类的 RDS for Oracle 数据库实例。	现在
Amazon RDS 启动了使用 db.r4 数据库实例类的 RDS for Oracle 数据库实例的自动升级，以将其升级为等效的 db.r5 数据库实例类。	2023 年 4 月 17 日

对于 RDS for MariaDB、RDS for MySQL、RDS for SQL Server 和 RDS for PostgreSQL 数据库引擎，Amazon RDS 已使用以下计划启动此数据库实例类的终止支持流程。对于使用此实例类的所有 RDS 数据库实例，建议您尽快升级到新一代的数据库实例类。

操作或建议	日期
从该日期开始，Amazon RDS 开始自动将使用 db.r4 的实例升级到新一代 db.r5 实例类。不再支持使用 db.m4 实例类创建数据库实例。	2024 年 6 月 1 日
Amazon RDS 终止对 db.r4 的支持。	2024 年 12 月 31 日

- db.r3 – 提供内存优化的实例类。

  对于 RDS for MariaDB、RDS for MySQL 和 RDS for PostgreSQL 数据库引擎，Amazon RDS 已使用以下计划启动 db.r3 数据库实例类的生命周期终止过程，其中包括升级建议。对于所有使用 db.r3 数据库实例类的 RDS 数据库实例，建议您尽快升级到更高代次的数据库实例类。

操作或建议	日期
您不能再创建使用 db.r3 数据库实例类的 RDS 数据库实例。	现在
Amazon RDS 启动了使用 db.r3 数据库实例类的 RDS 数据库实例的自动升级，以将其升级为等效的 db.r5 数据库实例类。	2023 年 2 月 1 日

计算优化型实例类类型

以下计算优化型实例类类型可用：

- db.c6gd – 非常适合运行高级计算密集型工作负载的实例类。这些实例类由 Amazon Graviton2 处理器提供支持，提供基于 NVMe 的本地 SSD 块级存储，适用于需要高速、低延迟本地存储的应用程序。

> **Note**
>
> 仅多可用区数据库集群部署支持 c6gd 实例类。它们是提供 `medium` 实例大小的多可用区数据库集群唯一支持的实例类。有关更多信息，请参阅 the section called “多可用区数据库集群部署”。

可突增性能实例类类型

提供了以下可突增性能数据库实例类类型：

- db.t4g - 由基于 Arm 的 Amazon Graviton2 处理器提供支持的通用实例类。与之前的可突增性能数据库实例类相比，这些实例类提供了更好的性价比，适用于广泛的可突增通用工作负载。Amazon RDS db.t4g 实例配置为无限模式。这表示实例可以在 24 小时时段内突增到基准以上，但需额外付费。

  您可以修改数据库实例，以使用由 Amazon Graviton2 处理器提供支持的其中一个数据库实例类。为此，请完成与修改任何其他数据库实例相同的步骤。
- db.t3 – 提供基准性能水平的实例类，并且可以突增到完全 CPU 使用率。db.t3 实例配置为无限模式。这些实例类提供比以前的 db.t2 实例类更多的计算容量。它们由 Amazon Nitro 系统（专用硬件和轻量级管理程序的组合）提供支持。
- db.t2 – 提供基准性能水平的实例类，并且可以突增到完全 CPU 使用率。db.t2 实例配置为无限模式。建议仅将这些实例类用于开发和测试服务器，或其他非生产服务器。

  对于 RDS for MariaDB、RDS for MySQL、RDS for SQL Server 和 RDS for PostgreSQL 数据库引擎，Amazon RDS 已使用以下计划启动此数据库实例类的终止支持流程。对于使用此实例类的所有 RDS 数据库实例，建议您尽快升级到新一代的数据库实例类。

操作或建议	日期
从该日期开始，Amazon RDS 开始自动将使用 db.t2 的实例升级到新一代 db.t3 实例类。不再支持使用 db.t2 实例类创建数据库实例。	2024 年 6 月 1 日
Amazon RDS 终止对 db.t2 的支持。	2024 年 12 月 31 日

> **Note**
>
> 对于读取和写入组合工作负载，使用 Amazon Nitro 系统的数据库实例类（db.m5、db.r5、db.t3）会受限制。

对于数据库实例类硬件规格，请参阅[数据库实例类的硬件规格](#)。

优化型读取实例类类型

以下是可用的优化型读取实例类类型：

- db.r6gd – 由 Amazon Graviton2 处理器提供支持的实例类。这些实例类非常适合运行内存密集型工作负载，并为需要高速、低延迟本地存储的应用程序提供基于本地 NVMe 的 SSD 块级存储。
- db.r6id – 由第三代 Intel Xeon 可扩展处理器提供支持的实例类。这些实例类已通过 SAP 认证，非常适合内存密集型工作负载。它们提供最多 1 TiB 的内存和高达 7.6 TB 基于 NVMe 的直连 SSD 存储。

数据库实例类支持的数据库引擎

以下是数据库实例类的数据库引擎特定注意事项：

Db2

数据库实例类支持因 Db2 的版本和版本号而异。有关不同版本和版本号支持的实例类，请参阅 Amazon RDS for Db2 实例类。

Microsoft SQL Server

数据库实例类支持因 SQL Server 的版本和版本号而异。有关不同版本和版本号支持的实例类，请参阅 Microsoft SQL Server 的数据库实例类支持。

Oracle

数据库实例类支持因 Oracle 数据库版本和版本号而异。适用于 Oracle 的 RDS 支持其他内存优化型实例类。这些类的名称格式为 db.r5.*instance_size*.tpc*threads_per_core*.mem*ratio*。有关每个优化类的 vCPU 计数和内存分配，请参阅 支持的 RDS for Oracle 数据库实例类。

RDS Custom

有关 RDS Custom 中支持的数据库实例类的信息，请参阅RDS Custom for Oracle 的数据库实例类支持和RDS Custom for SQL Server 的数据库实例类支持。

在下表中，您可以找到每个 Amazon RDS 数据库引擎支持的 Amazon RDS 数据库实例类的详细信息。每个引擎的单元格包含以下值之一：

是

数据库引擎的所有版本都支持此实例类。

否

数据库引擎不支持此实例类。

specific-versions

仅数据库引擎的指定数据库版本才支持此实例类。

Amazon RDS 会定期弃用主要和次要数据库引擎版本。并非所有 Amazon Web Services 区域都支持早期的引擎版本。有关当前支持的版本的信息，请参阅各个数据库引擎的主题：Db2 版本、MariaDB 版本、Microsoft SQL Server 版本、MySQL 版本、Oracle 版本和 PostgreSQL 版本。

主题

- 通用实例类支持的数据库引擎
- 内存优化型实例类支持的数据库引擎
- 计算优化型实例类支持的数据库引擎
- 可突增性能实例类支持的数据库引擎
- 优化型读取实例类支持的数据库引擎

通用实例类支持的数据库引擎

下表显示了通用实例类支持的数据库和数据库版本。

db.m8g – 由 Amazon Graviton4 处理器提供支持的通用实例类

实例类	Db2	MariaDB	Micros SQL Server	MySQL	Orac	PostgreSQL
db.m8g.48 xlarge	否	MariaDB 11.4.3 及更高版本、10.11.7 及更高版本、10.6.13 及更高版本、10.5.20 及更高版本和 10.4.29 及更高版本	否	MySQL 8.0.32 及更高版本	否	PostgreSQL 17.1 及更高版本、16.1 及更高版本、15.4 及更高版本、14.9 及更高版本和 13.11 及更高版本

实例类	Db2	MariaDB	Micros SQL Server	MySQL	Orac	PostgreSQL
db.m8g.24 xlarge	否	MariaDB 11.4.3 及更高版本、10.11.7 及更高版本、10.6.13 及更高版本、10.5.20 及更高版本和 10.4.29 及更高版本	否	MySQL 8.0.32 及更高版本	否	PostgreSQL 17.1 及更高版本、16.1 及更高版本、15.4 及更高版本、14.9 及更高版本和 13.11 及更高版本
db.m8g.16 xlarge	否	MariaDB 11.4.3 及更高版本、10.11.7 及更高版本、10.6.13 及更高版本、10.5.20 及更高版本和 10.4.29 及更高版本	否	MySQL 8.0.32 及更高版本	否	PostgreSQL 17.1 及更高版本、16.1 及更高版本、15.4 及更高版本、14.9 及更高版本和 13.11 及更高版本
db.m8g.12 xlarge	否	MariaDB 11.4.3 及更高版本、10.11.7 及更高版本、10.6.13 及更高版本、10.5.20 及更高版本和 10.4.29 及更高版本	否	MySQL 8.0.32 及更高版本	否	PostgreSQL 17.1 及更高版本、16.1 及更高版本、15.4 及更高版本、14.9 及更高版本和 13.11 及更高版本
db.m8g.8x large	否	MariaDB 11.4.3 及更高版本、10.11.7 及更高版本、10.6.13 及更高版本、10.5.20 及更高版本和 10.4.29 及更高版本	否	MySQL 8.0.32 及更高版本	否	PostgreSQL 17.1 及更高版本、16.1 及更高版本、15.4 及更高版本、14.9 及更高版本和 13.11 及更高版本

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.m8g.4xlarge	否	MariaDB 11.4.3 及更高版本、10.11.7 及更高版本、10.6.13 及更高版本、10.5.20 及更高版本和 10.4.29 及更高版本	否	MySQL 8.0.32 及更高版本	否	PostgreSQL 17.1 及更高版本、16.1 及更高版本、15.4 及更高版本、14.9 及更高版本和 13.11 及更高版本
db.m8g.2xlarge	否	MariaDB 11.4.3 及更高版本、10.11.7 及更高版本、10.6.13 及更高版本、10.5.20 及更高版本和 10.4.29 及更高版本	否	MySQL 8.0.32 及更高版本	否	PostgreSQL 17.1 及更高版本、16.1 及更高版本、15.4 及更高版本、14.9 及更高版本和 13.11 及更高版本
db.m8g.xlarge	否	MariaDB 11.4.3 及更高版本、10.11.7 及更高版本、10.6.13 及更高版本、10.5.20 及更高版本和 10.4.29 及更高版本	否	MySQL 8.0.32 及更高版本	否	PostgreSQL 17.1 及更高版本、16.1 及更高版本、15.4 及更高版本、14.9 及更高版本和 13.11 及更高版本
db.m8g.large	否	MariaDB 11.4.3 及更高版本、10.11.7 及更高版本、10.6.13 及更高版本、10.5.20 及更高版本和 10.4.29 及更高版本	否	MySQL 8.0.32 及更高版本	否	PostgreSQL 17.1 及更高版本、16.1 及更高版本、15.4 及更高版本、14.9 及更高版本和 13.11 及更高版本

db.m7i – 由第四代 Intel Xeon 可扩展处理器提供支持的通用实例类

实例类	Db2	MariaDB	Micros SQL Server	MySQL	Oracle	PostgreSQL
db.m7i.48xlarge	Db2 11.5	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.0.32 及更高版本	仅限 BYOL 仅限 EE	PostgreSQL 17.1 及更高版本、16.1 及更高版本、15.4 及更高版本、14.9 及更高版本和 13.11 及更高版本
db.m7i.24xlarge	Db2 11.5	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.0.32 及更高版本	仅限 BYOL 仅限 EE	PostgreSQL 17.1 及更高版本、16.1 及更高版本、15.4 及更高版本、14.9 及更高版本和 13.11 及更高版本
db.m7i.16xlarge	Db2 11.5	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.0.32 及更高版本	仅限 BYOL 仅限 EE	PostgreSQL 17.1 及更高版本、16.1 及更高版本、15.4 及更高版本、14.9 及更高版本和 13.11 及更高版本
db.m7i.12xlarge	Db2 11.5	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.0.32 及更高版本	仅限 BYOL 仅限 EE	PostgreSQL 17.1 及更高版本、16.1 及更高版本、15.4 及更高版本、14.9 及更高版本和 13.11 及更高版本
db.m7i.8xlarge	Db2 11.5	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.0.32 及更高版本	仅限 BYOL 仅限 EE	PostgreSQL 17.1 及更高版本、16.1 及更高版本、15.4 及更高版本、14.

实例类	Db2	MariaDB	Micros SQL Server	MySQL	Oracle	PostgreSQL
						9 及更高版本和 13.11 及更高版本
db.m7i.4xlarge	Db2 11.5	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.0.32 及更高版本	仅限 BYOL 所有版本	PostgreSQL 17.1 及更高版本、16.1 及更高版本、15.4 及更高版本、14.9 及更高版本和 13.11 及更高版本
db.m7i.2xlarge	Db2 11.5	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.0.32 及更高版本	仅限 BYOL 所有版本	PostgreSQL 17.1 及更高版本、16.1 及更高版本、15.4 及更高版本、14.9 及更高版本和 13.11 及更高版本
db.m7i.xlarge	Db2 11.5	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.0.32 及更高版本	仅限 BYOL 所有版本	PostgreSQL 17.1 及更高版本、16.1 及更高版本、15.4 及更高版本、14.9 及更高版本和 13.11 及更高版本
db.m7i.large	Db2 11.5	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.0.32 及更高版本	仅限 BYOL 所有版本	PostgreSQL 17.1 及更高版本、16.1 及更高版本、15.4 及更高版本、14.9 及更高版本和 13.11 及更高版本

db.m7g – 由 Amazon Graviton3 处理器提供支持的通用实例类

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.m7g.16xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.4 及更高的 13 版本
db.m7g.12xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.4 及更高的 13 版本
db.m7g.8xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.4 及更高的 13 版本
db.m7g.4xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.4 及更高的 13 版本
db.m7g.2xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
						13.4 及更高的 13 版本
db.m7g.xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.4 及更高的 13 版本
db.m7g.large	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.4 及更高的 13 版本

db.m6g – 由 Amazon Graviton2 处理器提供支持的通用实例类

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.m6g.16xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16、15、14 和 13 版本；以及 12.7 及更高的 12 版本
db.m6g.12xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16、15、14 和 13 版本；以及 12.7 及更高的 12 版本

实例类	Db2	MariaDB	Micros SQL Server	MySQL	Ora	PostgreSQL
db.m6g.8xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16、15、14 和 13 版本；以及 12.7 及更高的 12 版本
db.m6g.4xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16、15、14 和 13 版本；以及 12.7 及更高的 12 版本
db.m6g.2xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16、15、14 和 13 版本；以及 12.7 及更高的 12 版本
db.m6g.xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16、15、14 和 13 版本；以及 12.7 及更高的 12 版本
db.m6g.large	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16、15、14 和 13 版本；以及 12.7 及更高的 12 版本

db.m6gd – 由 Amazon Graviton2 处理器和 SSD 存储提供支持的通用实例类

实例类	Db2	MariaDB	Micros SQL Server	MySQL	Ora	PostgreSQL
db.m6gd.16xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16、15 和 14

实例类	Db2	MariaDB	Micros SQL Server	MySQL	Ora	PostgreSQL
						版本；13.7 及更高的 13 版本；以及 13.4 版本
db.m6gd.12xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16、15 和 14 版本；13.7 及更高的 13 版本；以及 13.4 版本
db.m6gd.8xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16、15 和 14 版本；13.7 及更高的 13 版本；以及 13.4 版本
db.m6gd.4xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16、15 和 14 版本；13.7 及更高的 13 版本；以及 13.4 版本
db.m6gd.2xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16、15 和 14 版本；13.7 及更高的 13 版本；以及 13.4 版本
db.m6gd.xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16、15 和 14 版本；13.7 及更高的 13 版本；以及 13.4 版本

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.m6gd.large	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16、15 和 14 版本；13.7 及更高的 13 版本；以及 13.4 版本

db.m6id – 由第三代 Intel Xeon 可扩展处理器和 SSD 存储提供支持的通用实例类

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.m6id.32xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	仅限 EE 和 BYOL	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本
db.m6id.24xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	仅限 EE 和 BYOL	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本
db.m6id.16xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	仅限 EE 和 BYOL	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本
db.m6id.12xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	仅限 EE 和 BYOL	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14

实例类	Db	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
						版本，以及 13.7 及更高的 13 版本
db.m6id.8xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	仅限 EE 和 BYOL	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本
db.m6id.4xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	仅限 BYOL	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本
db.m6id.2xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	仅限 BYOL	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本
db.m6id.xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	仅限 BYOL	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本
db.m6id.large	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	仅限 BYOL	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本

db.m6idn – 具有第三代 Intel Xeon 可扩展处理器、SSD 存储和网络优化的通用实例类

实例类	Db	MariaDB	Micros SQL Server	MySQL	Orac	PostgreSQL
db.m6idn.32xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本
db.m6idn.24xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本
db.m6idn.16xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本
db.m6idn.12xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本
db.m6idn.8xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本
db.m6idn.4xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14

实例类	Db	MariaDB	Micros SQL Server	MySQL	Orac	PostgreSQL
						版本，以及 13.7 及更高的 13 版本
db.m6idn.2xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本
db.m6idn.xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本
db.m6idn.large	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本

db.m6in – 由第三代 Intel Xeon 可扩展处理器和网络优化提供支持的通用实例类

实例类	Db	MariaDB	Micros SQL Server	MySQL	Ora	PostgreSQL
db.m6in.32xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.3 及更高的 14 版本、13.7 及更高的 13 版本、12.11 及更高的 12 版本和 11.16 及更高的 11 版本

实例类	Db	MariaDB	Micros SQL Server	MySQL	Ora	PostgreSQL
db.m6in.24xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.3 及更高的 14 版本、13.7 及更高的 13 版本、12.11 及更高的 12 版本和 11.16 及更高的 11 版本
db.m6in.16xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.3 及更高的 14 版本、13.7 及更高的 13 版本、12.11 及更高的 12 版本和 11.16 及更高的 11 版本
db.m6in.12xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.3 及更高的 14 版本、13.7 及更高的 13 版本、12.11 及更高的 12 版本和 11.16 及更高的 11 版本
db.m6in.8xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.3 及更高的 14 版本、13.7 及更高的 13 版本、12.11 及更高的 12 版本和 11.16 及更高的 11 版本

实例类	Db	MariaDB	Micros SQL Server	MySQL	Ora	PostgreSQL
db.m6in.4xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.3 及更高的 14 版本、13.7 及更高的 13 版本、12.11 及更高的 12 版本和 11.16 及更高的 11 版本
db.m6in.2xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.3 及更高的 14 版本、13.7 及更高的 13 版本、12.11 及更高的 12 版本和 11.16 及更高的 11 版本
db.m6in.xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.3 及更高的 14 版本、13.7 及更高的 13 版本、12.11 及更高的 12 版本和 11.16 及更高的 11 版本
db.m6in.large	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.3 及更高的 14 版本、13.7 及更高的 13 版本、12.11 及更高的 12 版本和 11.16 及更高的 11 版本

db.m6i – 由第三代 Intel Xeon 可扩展处理器提供支持的通用实例类

实例类	Db	MariaDB	Micros SQL Server	MySQL	Oracle	PostgreSQL
db.m6i.32xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	Oracle Databa 19c	所有可用版本
db.m6i.24xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	Oracle Databa 19c	所有可用版本
db.m6i.16xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	Oracle Databa 19c	所有可用版本
db.m6i.12xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	Oracle Databa 19c	所有可用版本
db.m6i.8xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	Oracle Databa 19c	所有可用版本
db.m6i.4xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	Oracle Databa 19c	所有可用版本
db.m6i.2xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	Oracle Databa 19c	所有可用版本
db.m6i.xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	Oracle Databa 19c	所有可用版本

实例类	Db	MariaDB	Micros SQL Servei	MySQL	Oracle	PostgreSQL
db.m6i.large	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	Oracle Databa 19c	所有可用版本

db.m5d – 由 Intel Xeon Platinum 处理器和 SSD 存储提供支持的通用实例类

实例类	Db:	MariaDB	Microsc SQL Server	MySQL	Orac	PostgreSQL
db.m5d.24xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	是	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本
db.m5d.16xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	是	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本
db.m5d.12xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	是	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本
db.m5d.8xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	是	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本
db.m5d.4xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	是	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
						版本、13.7 及更高的 13 版本和 13.4 版本
db.m5d.2xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	是	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本
db.m5d.xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	是	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本
db.m5d.large	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	是	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本

db.m5 – 具有 2.5GHz Intel Xeon Platinum 处理器的通用实例类

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.m5.24xlarge	否	是	是	是	是	所有 PostgreSQL 17、16、15、14、13、12 和 11 版本；10.17 及更高的 10 版本；以及 9.6.22 及更高的 9 版本
db.m5.16xlarge	否	是	是	是	是	所有 PostgreSQL 17、16、15、14、13、12 和 11 版本；10.17 及更高的 10 版本；以及 9.6.22 及更高的 9 版本

实例类	Db2	Maria	Microsoft SQL Server	MyS	Oracl	PostgreSQL
db.m5.12xlarge	否	是	是	是	是	所有 PostgreSQL 17、16、15、14、13、12 和 11 版本；10.17 及更高的 10 版本；以及 9.6.22 及更高的 9 版本
db.m5.8xlarge	否	是	是	是	是	所有 PostgreSQL 17、16、15、14、13、12 和 11 版本；10.17 及更高的 10 版本；以及 9.6.22 及更高的 9 版本
db.m5.4xlarge	否	是	是	是	是	所有 PostgreSQL 17、16、15、14、13、12 和 11 版本；10.17 及更高的 10 版本；以及 9.6.22 及更高的 9 版本
db.m5.2xlarge	否	是	是	是	是	所有 PostgreSQL 17、16、15、14、13、12 和 11 版本；10.17 及更高的 10 版本；以及 9.6.22 及更高的 9 版本
db.m5.xlarge	否	是	是	是	是	所有 PostgreSQL 17、16、15、14、13、12 和 11 版本；10.17 及更高的 10 版本；以及 9.6.22 及更高的 9 版本
db.m5.large	否	是	是	是	是	所有 PostgreSQL 17、16、15、14、13、12 和 11 版本；10.17 及更高的 10 版本；以及 9.6.22 及更高的 9 版本

db.m4 – 具有 Intel Xeon 处理器的通用实例类

实例类	Db	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.m4.16xlarge	否	已弃用	已弃用	已弃用	已弃用	已弃用

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.m4.10xlarge	否	已弃用	已弃用	已弃用	已弃用	已弃用
db.m4.4xlarge	否	已弃用	已弃用	已弃用	已弃用	已弃用
db.m4.2xlarge	否	已弃用	已弃用	已弃用	已弃用	已弃用
db.m4.xlarge	否	已弃用	已弃用	已弃用	已弃用	已弃用
db.m4.large	否	已弃用	已弃用	已弃用	已弃用	已弃用

db.m3 – 通用实例类

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.m3.2xlarge	否	否	已弃用	是	已淘汰	已弃用
db.m3.xlarge	否	否	已弃用	是	已淘汰	已弃用
db.m3.large	否	否	已弃用	是	已淘汰	已弃用

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.m3.medium	否	否	已弃用	是	已淘汰	已弃用

内存优化型实例类支持的数据库引擎

下表显示了内存优化型实例类支持的数据库和数据库版本。

db.z1d – 内存优化型实例类

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.z1d.1: xlarge	否	否	是	否	是	否
db.z1d.6: large	否	否	是	否	是	否
db.z1d.3: large	否	否	是	否	是	否
db.z1d.2: large	否	否	是	否	是	否
db.z1d.xlarge	否	否	是	否	是	否
db.z1d.large	否	否	是	否	是	否

db.x2g – 由 Amazon Graviton2 处理器提供支持的内存优化型实例类

实例类	Db	MariaDB	Micros SQL Server	MySQL	Ora	PostgreSQL
db.x2g.1 xlarge	否	MariaDB 11.4、10.1 1、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16、15、14 和 13 版本；以及 12.7 及更高的 12 版本
db.x2g.1 xlarge	否	MariaDB 11.4、10.1 1、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16、15、14 和 13 版本；以及 12.7 及更高的 12 版本
db.x2g.8 large	否	MariaDB 11.4、10.1 1、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16、15、14 和 13 版本；以及 12.7 及更高的 12 版本
db.x2g.4 large	否	MariaDB 11.4、10.1 1、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16、15、14 和 13 版本；以及 12.7 及更高的 12 版本
db.x2g.2 large	否	MariaDB 11.4、10.1 1、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16、15、14 和 13 版本；以及 12.7 及更高的 12 版本
db.x2g.x arge	否	MariaDB 11.4、10.1 1、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16、15、14 和 13 版本；以及 12.7 及更高的 12 版本
db.x2g.la rge	否	MariaDB 11.4、10.1 1、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16、15、14 和 13 版本；以及 12.7 及更高的 12 版本

db.x2idn – 由第三代 Intel Xeon 可扩展处理器提供支持的内存优化型实例类

实例类	Db	MariaDB	Micros SQL Server	MySQL	Oracle	PostgreSQL
db.x2idn.32xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	仅企业版	PostgreSQL 15 版本、14.6 和 13.9
db.x2idn.24xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	仅企业版	PostgreSQL 15 版本、14.6 和 13.9
db.x2idn.16xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	仅企业版	PostgreSQL 15 版本、14.6 和 13.9

db.x2iedn – 使用基于本地 NVMe 的 SSD 的内存优化型实例类，由第三代 Intel Xeon 可扩展处理器提供支持

实例类	DI	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.x2iedn.32xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	仅限企业版和标准版，SQL Server 2014 12.00 和更高版本	MySQL 8.4 和 8.0	仅企业版	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本
db.x2iedn.24xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	仅限企业版和标准版，SQL Server 2014 12.00 和更高版本	MySQL 8.4 和 8.0	仅企业版	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本

实例类	DI	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.x2iedn.16xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	仅限企业版和标准版，SQL Server 2014 12.00 和更高版本	MySQL 8.4 和 8.0	仅企业版	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本
db.x2iedn.8xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	仅限企业版和标准版，SQL Server 2014 12.00 和更高版本	MySQL 8.4 和 8.0	仅企业版	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本
db.x2iedn.4xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	仅限企业版和标准版，SQL Server 2014 12.00 和更高版本	MySQL 8.4 和 8.0	企业版和标准版 2（SE	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本
db.x2iedn.2xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	仅限企业版和标准版，SQL Server 2014 12.00 和更高版本	MySQL 8.4 和 8.0	企业版和标准版 2（SE	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本
db.x2iedn.xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	仅限企业版和标准版，SQL Server 2014 12.00 和更高版本	MySQL 8.4 和 8.0	企业版和标准版 2（SE	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本

db.x2iezn – 由第二代 Intel Xeon 可扩展处理器提供支持的内存优化型实例类

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.x2iezn.8xlarge	否	否	否	否	仅企业版	否
db.x2iezn.6xlarge	否	否	否	否	仅企业版	否
db.x2iezn.4xlarge	否	否	否	否	企业版和标准版 2（SE2）	否
db.x2iezn.2xlarge	否	否	否	否	企业版和标准版 2（SE2）	否

db.x1e – 内存优化型实例类

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.x1e.32xlarge	否	否	是	否	已弃用[1]	否
db.x1e.16xlarge	否	否	是	否	已弃用[1]	否
db.x1e.8xlarge	否	否	是	否	已弃用[1]	否
db.x1e.4xlarge	否	否	是	否	已弃用[1]	否
db.x1e.2xlarge	否	否	是	否	已弃用[1]	否

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.x1e.xlarge	否	否	是	否	已弃用[1]	否

[1] 您不能再使用 X1 实例类族创建 RDS for Oracle 数据库实例。如果您当前使用的是 X1 类，请尽快切换到新一代实例类。从 2025 年 1 月 22 日起，RDS 将在您定义的维护时段内开始自动升级。在升级期间，RDS 会选择等效的 X2iedn 实例类型并对其进行升级。欲了解更多信息，请参阅 re:Post 文章 [Amazon RDS for Oracle is ending support for X1 Database Instances on January 22, 2025]。

db.x1 – 内存优化型实例类

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.x1.32xlarge	否	否	是	否	已弃用[1]	否
db.x1.16xlarge	否	否	是	否	已弃用[1]	否

[1] 您不能再使用 X1 实例类族创建 RDS for Oracle 数据库实例。如果您当前使用的是 X1 类，请尽快切换到新一代实例类。从 2025 年 1 月 22 日起，RDS 将在您定义的维护时段内开始自动升级。在升级期间，RDS 会选择等效的 X2iedn 实例类型并对其进行升级。欲了解更多信息，请参阅 re:Post 文章 [Amazon RDS for Oracle is ending support for X1 Database Instances on January 22, 2025]。

db.r8g – 由 Amazon Graviton4 处理器提供支持的内存优化型实例类

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.r8g.48xlarge	否	MariaDB 11.4.3 及更高版本、10.11.7 及更高版本、10.6.13 及更高版	否	MySQL 8.0.32 及更高版本	否	PostgreSQL 17.1 及更高版本、16.1 及更高版本、15.

实例类	Db2	MariaDB	Micros SQL Server	MySQL	Orac	PostgreSQL
		本、10.5.20 及更高版本和 10.4.29 及更高版本				4 及更高版本、14. 9 及更高版本和 13.11 及更高版本
db.r8g.24 xlarge	否	MariaDB 11.4.3 及更高版本、10.11.7 及更高版本、10.6.13 及更高版本、10.5.20 及更高版本和 10.4.29 及更高版本	否	MySQL 8.0.32 及更高版本	否	PostgreSQL 17.1 及更高版本、16. 1 及更高版本、15. 4 及更高版本、14. 9 及更高版本和 13.11 及更高版本
db.r8g.16 xlarge	否	MariaDB 11.4.3 及更高版本、10.11.7 及更高版本、10.6.13 及更高版本、10.5.20 及更高版本和 10.4.29 及更高版本	否	MySQL 8.0.32 及更高版本	否	PostgreSQL 17.1 及更高版本、16. 1 及更高版本、15. 4 及更高版本、14. 9 及更高版本和 13.11 及更高版本
db.r8g.12 xlarge	否	MariaDB 11.4.3 及更高版本、10.11.7 及更高版本、10.6.13 及更高版本、10.5.20 及更高版本和 10.4.29 及更高版本	否	MySQL 8.0.32 及更高版本	否	PostgreSQL 17.1 及更高版本、16. 1 及更高版本、15. 4 及更高版本、14. 9 及更高版本和 13.11 及更高版本
db.r8g.8x large	否	MariaDB 11.4.3 及更高版本、10.11.7 及更高版本、10.6.13 及更高版本、10.5.20 及更高版本和 10.4.29 及更高版本	否	MySQL 8.0.32 及更高版本	否	PostgreSQL 17.1 及更高版本、16. 1 及更高版本、15. 4 及更高版本、14. 9 及更高版本和 13.11 及更高版本

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.r8g.4xlarge	否	MariaDB 11.4.3 及更高版本、10.11.7 及更高版本、10.6.13 及更高版本、10.5.20 及更高版本和 10.4.29 及更高版本	否	MySQL 8.0.32 及更高版本	否	PostgreSQL 17.1 及更高版本、16.1 及更高版本、15.4 及更高版本、14.9 及更高版本和 13.11 及更高版本
db.r8g.2xlarge	否	MariaDB 11.4.3 及更高版本、10.11.7 及更高版本、10.6.13 及更高版本、10.5.20 及更高版本和 10.4.29 及更高版本	否	MySQL 8.0.32 及更高版本	否	PostgreSQL 17.1 及更高版本、16.1 及更高版本、15.4 及更高版本、14.9 及更高版本和 13.11 及更高版本
db.r8g.xlarge	否	MariaDB 11.4.3 及更高版本、10.11.7 及更高版本、10.6.13 及更高版本、10.5.20 及更高版本和 10.4.29 及更高版本	否	MySQL 8.0.32 及更高版本	否	PostgreSQL 17.1 及更高版本、16.1 及更高版本、15.4 及更高版本、14.9 及更高版本和 13.11 及更高版本
db.r8g.large	否	MariaDB 11.4.3 及更高版本、10.11.7 及更高版本、10.6.13 及更高版本、10.5.20 及更高版本和 10.4.29 及更高版本	否	MySQL 8.0.32 及更高版本	否	PostgreSQL 17.1 及更高版本、16.1 及更高版本、15.4 及更高版本、14.9 及更高版本和 13.11 及更高版本

db.r7i – 由第四代 Intel Xeon 可扩展处理器提供支持的内存优化型实例类

实例类	Db	MariaDB	Microsс SQL Server	MySQL	Oracle	PostgreSQL
db.r7i.48 xlarge	Db 11	MariaDB 11.4、10.1 1、10.6、10.5 和 10.4	否	MySQL 8.0.32 及更高版本	仅限 BYOL	PostgreSQL 17.1 及更高版本、16. 1 及更高版本、15. 4 及更高版本、14. 9 及更高版本和 13.11 及更高版本
db.r7i.24 xlarge	Db 11	MariaDB 11.4、10.1 1、10.6、10.5 和 10.4	否	MySQL 8.0.32 及更高版本	仅限 BYOL	PostgreSQL 17.1 及更高版本、16. 1 及更高版本、15. 4 及更高版本、14. 9 及更高版本和 13.11 及更高版本
db.r7i.16 xlarge	Db 11	MariaDB 11.4、10.1 1、10.6、10.5 和 10.4	否	MySQL 8.0.32 及更高版本	仅限 BYOL	PostgreSQL 17.1 及更高版本、16. 1 及更高版本、15. 4 及更高版本、14. 9 及更高版本和 13.11 及更高版本
db.r7i.12 xlarge	Db 11	MariaDB 11.4、10.1 1、10.6、10.5 和 10.4	否	MySQL 8.0.32 及更高版本	仅限 BYOL	PostgreSQL 17.1 及更高版本、16. 1 及更高版本、15. 4 及更高版本、14. 9 及更高版本和 13.11 及更高版本
db.r7i.8x large	Db 11	MariaDB 11.4、10.1 1、10.6、10.5 和 10.4	否	MySQL 8.0.32 及更高版本	仅限 BYOL	PostgreSQL 17.1 及更高版本、16. 1 及更高版本、15. 4 及更高版本、14.

实例类	Db	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
						9 及更高版本和 13.11 及更高版本
db.r7i.4xlarge	Db 11	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.0.32 及更高版本	仅限 BYOL，所有版本	PostgreSQL 17.1 及更高版本、16.1 及更高版本、15.4 及更高版本、14.9 及更高版本和 13.11 及更高版本
db.r7i.2xlarge	Db 11	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.0.32 及更高版本	仅限 BYOL，所有版本	PostgreSQL 17.1 及更高版本、16.1 及更高版本、15.4 及更高版本、14.9 及更高版本和 13.11 及更高版本
db.r7i.xlarge	Db 11	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.0.32 及更高版本	仅限 BYOL，所有版本	PostgreSQL 17.1 及更高版本、16.1 及更高版本、15.4 及更高版本、14.9 及更高版本和 13.11 及更高版本
db.r7i.large	Db 11	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.0.32 及更高版本	仅限 BYOL，所有版本	PostgreSQL 17.1 及更高版本、16.1 及更高版本、15.4 及更高版本、14.9 及更高版本和 13.11 及更高版本

db.r7g – 由 Amazon Graviton3 处理器提供支持的内存优化型实例类

实例类	DI	MariaDB	Micros SQL Server	MySQL	Ora	PostgreSQL
db.r7g.1 xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.4 及更高的 13 版本
db.r7g.1 xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.4 及更高的 13 版本
db.r7g.8 large	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.4 及更高的 13 版本
db.r7g.4 large	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.4 及更高的 13 版本
db.r7g.2 large	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.4 及更高的 13 版本
db.r7g.xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
						本，以及 13.4 及更高的 13 版本
db.r7g.large	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.4 及更高的 13 版本

db.r6g – 由 Amazon Graviton2 处理器提供支持的内存优化型实例类

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.r6g.16xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16、15、14 和 13 版本；以及 12.7 及更高的 12 版本
db.r6g.12xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16、15、14 和 13 版本；以及 12.7 及更高的 12 版本
db.r6g.8xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16、15、14 和 13 版本；以及 12.7 及更高的 12 版本
db.r6g.4xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16、15、14 和 13 版

实例类	Db2	MariaDB	Micros SQL Server	MySQL	Ora	PostgreSQL
						本；以及 12.7 及更高的 12 版本
db.r6g.2xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16、15、14 和 13 版本；以及 12.7 及更高的 12 版本
db.r6g.xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16、15、14 和 13 版本；以及 12.7 及更高的 12 版本
db.r6g.large	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16、15、14 和 13 版本；以及 12.7 及更高的 12 版本

db.r6gd – 由 Amazon Graviton2 处理器提供支持的内存优化型实例类

实例类	Dl	MariaDB	Micros SQL Server	MySQL	Ora	PostgreSQL
db.r6gd.16xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本
db.r6gd.12xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本

实例类	Dl	MariaDB	Micros SQL Servei	MySQL	Ora	PostgreSQL
db.r6gd.8 xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本
db.r6gd.4 xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本
db.r6gd.2 xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本
db.r6gd.x large	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本
db.r6gd.l arge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本

db.r6id – 由第三代 Intel Xeon 可扩展处理器提供支持的内存优化型实例类

实例类	Db2	MariaDB	Microso SQL Server	MySQL	Oracle	PostgreSQL
db.r6id.32xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	仅限 EE	所有 PostgreSQL 17、16 和 15 版

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
					和 BYOL	本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本
db.r6id.24xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	仅限 EE 和 BYOL	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本
db.r6id.16xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	仅限 EE 和 BYOL	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本
db.r6id.12xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	仅限 EE 和 BYOL	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本
db.r6id.8xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	仅限 EE 和 BYOL	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本
db.r6id.4xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	仅限 BYOL	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.r6id.2xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	仅限 BYOL	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本
db.r6id.xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	仅限 BYOL	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本
db.r6id.large	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	仅限 BYOL	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本

db.r6idn – 由第三代 Intel Xeon 可扩展处理器提供支持的内存优化型实例类

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.r6idn.32xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本
db.r6idn.24xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14

实例类	Db2	MariaDB	Micros SQL Server	MySQL	Ora	PostgreSQL
						版本，以及 13.7 及更高的 13 版本
db.r6idn.16xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本
db.r6idn.12xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本
db.r6idn.8xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本
db.r6idn.4xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本
db.r6idn.2xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本

实例类	Db2	MariaDB	Micros SQL Server	MySQL	Ora	PostgreSQL
db.r6idn.xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本

db.r6in – 由第三代 Intel Xeon 可扩展处理器提供支持的内存优化型实例类

实例类	Db2	MariaDB	Micros SQL Server	MySQL	Ora	PostgreSQL
db.r6in.32xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.3 及更高的 14 版本、13.7 及更高的 13 版本、12.11 及更高的 12 版本和 11.16 及更高的 11 版本
db.r6in.24xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.3 及更高的 14 版本、13.7 及更高的 13 版本、12.11 及更高的 12 版本和 11.16 及更高的 11 版本
db.r6in.16xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.3 及更高的 14 版本、13.7 及更高的 13 版本、12.11 及更高的 12 版本和 11.16 及更高的 11 版本
db.r6in.12xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.3 及更高的 14 版本、13.7 及更高的 13 版

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
						本、12.11 及更高的 12 版本和 11.16 及更高的 11 版本
db.r6in.8xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.3 及更高的 14 版本、13.7 及更高的 13 版本、12.11 及更高的 12 版本和 11.16 及更高的 11 版本
db.r6in.4xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.3 及更高的 14 版本、13.7 及更高的 13 版本、12.11 及更高的 12 版本和 11.16 及更高的 11 版本
db.r6in.2xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.3 及更高的 14 版本、13.7 及更高的 13 版本、12.11 及更高的 12 版本和 11.16 及更高的 11 版本
db.r6in.xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.3 及更高的 14 版本、13.7 及更高的 13 版本、12.11 及更高的 12 版本和 11.16 及更高的 11 版本
db.r6in.large	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.3 及更高的 14 版本、13.7 及更高的 13 版本、12.11 及更高的 12 版本和 11.16 及更高的 11 版本

db.r6i – 针对高内存、存储和输入/输出预配置的内存优化型实例类

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.r6i.8xlarge.tpc2.mem4x	否	否	否	否	仅限 EE	否
db.r6i.8xlarge.tpc2.mem3x	否	否	否	否	仅限 EE	否
db.r6i.6xlarge.tpc2.mem4x	否	否	否	否	仅限 EE	否
db.r6i.4xlarge.tpc2.mem4x	否	否	否	否	EE 和 SE2 BYOL	否
db.r6i.4xlarge.tpc2.mem3x	否	否	否	否	EE 和 SE2 BYOL	否
db.r6i.4xlarge.tpc2.mem2x	否	否	否	否	EE 和 SE2 BYOL	否
db.r6i.2xlarge.tpc2.mem8x	否	否	否	否	EE 和 SE2 BYOL	否
db.r6i.2xlarge.tpc2.mem4x	否	否	否	否	EE 和 SE2 BYOL	否

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.r6i.2xlarge.tpc1.mem2x	否	否	否	否	EE 和 SE2 BYOL	否
db.r6i.xlarge.tpc2.mem4x	否	否	否	否	EE 和 SE2 BYOL	否
db.r6i.xlarge.tpc2.mem2x	否	否	否	否	EE 和 SE2 BYOL	否
db.r6i.large.tpc1.mem2x	否	否	否	否	EE 和 SE2 BYOL	否

db.r6i – 内存优化型实例类

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.r6i.32xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	是	所有 PostgreSQL 17、16、15 和 14 版本、13.4 及更高的 13 版本、12.8 及更高的 12 版本、11.13 及更高的 11 版本，以及 10.21 及更高的 10 版本
db.r6i.24xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	是	所有 PostgreSQL 17、16、15 和 14 版本、13.4 及更高的 13 版本

实例类	DI	MariaDB	Micros SQL Serve	MySQL	Ora	PostgreSQL
						本、12.8 及更高的 12 版本、11.13 及更高的 11 版本，以及 10.21 及更高的 10 版本
db.r6i.1(xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	是	所有 PostgreSQL 17、16、15 和 14 版本、13.4 及更高的 13 版本、12.8 及更高的 12 版本、11.13 及更高的 11 版本，以及 10.21 及更高的 10 版本
db.r6i.1: xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	是	所有 PostgreSQL 17、16、15 和 14 版本、13.4 及更高的 13 版本、12.8 及更高的 12 版本、11.13 及更高的 11 版本，以及 10.21 及更高的 10 版本
db.r6i.8: large	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	是	所有 PostgreSQL 17、16、15 和 14 版本、13.4 及更高的 13 版本、12.8 及更高的 12 版本、11.13 及更高的 11 版本，以及 10.21 及更高的 10 版本

实例类	Dl	MariaDB	Micros SQL Serve	MySQL	Ora	PostgreSQL
db.r6i.4x large	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	是	所有 PostgreSQL 17、16、15 和 14 版本、13.4 及更高的 13 版本、12.8 及更高的 12 版本、11.13 及更高的 11 版本，以及 10.21 及更高的 10 版本
db.r6i.2x large	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	是	所有 PostgreSQL 17、16、15 和 14 版本、13.4 及更高的 13 版本、12.8 及更高的 12 版本、11.13 及更高的 11 版本，以及 10.21 及更高的 10 版本
db.r6i.xlarge	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	是	所有 PostgreSQL 17、16、15 和 14 版本、13.4 及更高的 13 版本、12.8 及更高的 12 版本、11.13 及更高的 11 版本，以及 10.21 及更高的 10 版本
db.r6i.large	是	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	是	所有 PostgreSQL 17、16、15 和 14 版本、13.4 及更高的 13 版本、12.8 及更高的 12 版本、11.13 及更高的 11 版本，以及 10.21 及更高的 10 版本

db.r5d – 内存优化型实例类

实例类	DI	MariaDB	Micros SQL Serve	MySQL	Ora	PostgreSQL
db.r5d.2 xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	是	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本
db.r5d.1 xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	是	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本
db.r5d.1 xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	是	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本
db.r5d.8 large	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	是	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本
db.r5d.4 large	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	是	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本
db.r5d.2 large	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	是	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本
db.r5d.x arge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	是	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.r5d.large	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	是	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本

db.r5b – 针对高内存、存储和输入/输出进行预配置的内存优化型实例类

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.r5b.8xlarge.tpc2.mem3x	否	否	否	否	是	否
db.r5b.6xlarge.tpc2.mem4x	否	否	否	否	是	否
db.r5b.4xlarge.tpc2.mem4x	否	否	否	否	是	否
db.r5b.4xlarge.tpc2.mem3x	否	否	否	否	是	否
db.r5b.4xlarge.tpc2.mem2x	否	否	否	否	是	否

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.r5b.2xlarge.tpc2.mem8x	否	否	否	否	是	否
db.r5b.2xlarge.tpc2.mem4x	否	否	否	否	是	否
db.r5b.2xlarge.tpc1.mem2x	否	否	否	否	是	否
db.r5b.xlarge.tpc2.mem4x	否	否	否	否	是	否
db.r5b.xlarge.tpc2.mem2x	否	否	否	否	是	否
db.r5b.large.tpc1.mem2x	否	否	否	否	是	否

db.r5b – 内存优化型实例类

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.r5b.24xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	是	所有 PostgreSQL 17、16、15、14 和 13

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
						版本；以及 12.7 及更高的 12 版本
db.r5b.16xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	是	所有 PostgreSQL 17、16、15、14 和 13 版本；以及 12.7 及更高的 12 版本
db.r5b.12xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	是	所有 PostgreSQL 17、16、15、14 和 13 版本；以及 12.7 及更高的 12 版本
db.r5b.8xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	> 是	所有 PostgreSQL 17、16、15、14 和 13 版本；以及 12.7 及更高的 12 版本
db.r5b.4xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	是	所有 PostgreSQL 17、16、15、14 和 13 版本；以及 12.7 及更高的 12 版本
db.r5b.2xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	是	所有 PostgreSQL 17、16、15、14 和 13 版本；以及 12.7 及更高的 12 版本
db.r5b.xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	是	所有 PostgreSQL 17、16、15、14 和 13 版本；以及 12.7 及更高的 12 版本

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.r5b.large	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	是	MySQL 8.4 和 8.0	是	所有 PostgreSQL 17、16、15、14 和 13 版本；以及 12.7 及更高的 12 版本

db.r5 – 针对高内存、存储和输入/输出进行预配置的内存优化型实例类

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.r5.12xlarge.tpc2.mem2x	否	否	否	否	是	否
db.r5.8xlarge.tpc2.mem3x	否	否	否	否	是	否
db.r5.6xlarge.tpc2.mem4x	否	否	否	否	是	否
db.r5.4xlarge.tpc2.mem4x	否	否	否	否	是	否
db.r5.4xlarge.tpc2.mem3x	否	否	否	否	是	否
db.r5.4xlarge.tpc2.mem2x	否	否	否	否	是	否
db.r5.2xlarge.tpc2.mem8x	否	否	否	否	是	否
db.r5.2xlarge.tpc2.mem4x	否	否	否	否	是	否

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.r5.2xlarge.tpc1.mem2x	否	否	否	否	是	否
db.r5.xlarge.tpc2.mem4x	否	否	否	否	是	否
db.r5.xlarge.tpc2.mem2x	否	否	否	否	是	否
db.r5.large.tpc1.mem2x	否	否	否	否	是	否

db.r5 – 内存优化型实例类

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.r5.24xlarge	否	是	是	是	是	所有 PostgreSQL 17、16、15、14、13、12 和 11 版本；10.17 及更高的 10 版本；以及 9.6.22 及更高的 9 版本
db.r5.16xlarge	否	是	是	是	是	所有 PostgreSQL 17、16、15、14、13、12 和 11 版本；10.17 及更高的 10 版本；以及 9.6.22 及更高的 9 版本
db.r5.12xlarge	否	是	是	是	是	所有 PostgreSQL 17、16、15、14、13、12 和 11 版本；10.17 及更高的 10 版本；以及 9.6.22 及更高的 9 版本
db.r5.8xlarge	否	是	是	是	是	所有 PostgreSQL 17、16、15、14、13、12 和 11 版本；10.17 及更高的 10 版本；以及 9.6.22 及更高的 9 版本

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.r5.4xlarge	否	是	是	是	是	所有 PostgreSQL 17、16、15、14、13、12 和 11 版本；10.17 及更高的 10 版本；以及 9.6.22 及更高的 9 版本
db.r5.2xlarge	否	是	是	是	是	所有 PostgreSQL 17、16、15、14、13、12 和 11 版本；10.17 及更高的 10 版本；以及 9.6.22 及更高的 9 版本
db.r5.xlarge	否	是	是	是	是	所有 PostgreSQL 17、16、15、14、13、12 和 11 版本；10.17 及更高的 10 版本；以及 9.6.22 及更高的 9 版本
db.r5.large	否	是	是	是	是	所有 PostgreSQL 17、16、15、14、13、12 和 11 版本；10.17 及更高的 10 版本；以及 9.6.22 及更高的 9 版本

db.r4 – 内存优化型实例类

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.r4.16xlarge	否	已弃用	已弃用	已弃用	已弃用	已弃用
db.r4.8xlarge	否	已弃用	已弃用	已弃用	已弃用	已弃用
db.r4.4xlarge	否	已弃用	已弃用	已弃用	已弃用	已弃用
db.r4.2xlarge	否	已弃用	已弃用	已弃用	已弃用	已弃用

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.r4.xlarge	否	已弃用	已弃用	已弃用	已弃用	已弃用
db.r4.large	否	已弃用	已弃用	已弃用	已弃用	已弃用

db.r3 – 内存优化型实例类

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.r3.8xlarge**	否	所有 MariaDB 10.6、10.5、10.4 和 10.3 版本	已弃用	是	已淘汰	已弃用
db.r3.4xlarge	否	所有 MariaDB 10.6、10.5、10.4 和 10.3 版本	已弃用	是	已淘汰	已弃用
db.r3.2xlarge	否	所有 MariaDB 10.6、10.5、10.4 和 10.3 版本	已弃用	是	已淘汰	已弃用
db.r3.xlarge	否	所有 MariaDB 10.6、10.5、10.4 和 10.3 版本	已弃用	是	已淘汰	已弃用
db.r3.large	否	所有 MariaDB 10.6、10.5、10.4 和 10.3 版本	已弃用	是	已淘汰	已弃用

计算优化型实例类支持的数据库引擎

下表显示了计算优化型实例类支持的数据库和数据库版本。

db.c6gd – 计算优化型实例类（仅适用于多可用区数据库集群部署）

实例类	Db2	Maria	Microsoft SQL Server	MySQL	Orac	PostgreSQL
db.c6gd.16xlarge	否	否	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本；14.5 及更高的 14 版本；13.4 和 13.7 及更高的 13 版本
db.c6gd.12xlarge	否	否	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本；14.5 及更高的 14 版本；13.4 和 13.7 及更高的 13 版本
db.c6gd.8xlarge	否	否	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本；14.5 及更高的 14 版本；13.4 和 13.7 及更高的 13 版本
db.c6gd.4xlarge	否	否	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本；14.5 及更高的 14 版本；13.4 和 13.7 及更高的 13 版本
db.c6gd.2xlarge	否	否	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本；14.5 及更高的 14 版本；13.4 和 13.7 及更高的 13 版本
db.c6gd.xlarge	否	否	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本；14.5 及更高的 14 版本；13.4 和 13.7 及更高的 13 版本
db.c6gd.large	否	否	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本；14.5 及更高的 14 版本；13.4 和 13.7 及更高的 13 版本
db.c6gd.medium	否	否	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本；14.5 及更高的 14 版本；13.4 和 13.7 及更高的 13 版本

可突增性能实例类支持的数据库引擎

下表显示了可突增性能实例类支持的数据库和数据库版本。

db.t4g - 由 Amazon Graviton2 处理器提供支持的可突增性能实例类

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.t4g.2xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16、15、14 和 13 版本；以及 12.7 及更高的 12 版本
db.t4g.xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16、15、14 和 13 版本；以及 12.7 及更高的 12 版本
db.t4g.large	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16、15、14 和 13 版本；以及 12.7 及更高的 12 版本
db.t4g.medium	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16、15、14 和 13 版本；以及 12.7 及更高的 12 版本
db.t4g.small	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16、15、14 和 13 版本；以及 12.7 及更高的 12 版本
db.t4g.micro	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16、15、14 和 13 版本；以及 12.7 及更高的 12 版本

db.t3 – 具爆发能力的实例类

实例类	Db2	Maria	Microsoft SQL Server	MySC	Oracle	PostgreSQL
db.t3.2xlarge	是	是	是	是	是	所有 PostgreSQL 17、16、15、14、13、12、11 和 10 版本；以及 9.6.22 及更高的 9 版本
db.t3.xlarge	是	是	是	是	是	所有 PostgreSQL 17、16、15、14、13、12、11 和 10 版本；以及 9.6.22 及更高的 9 版本
db.t3.large	是	是	是	是	是	所有 PostgreSQL 17、16、15、14、13、12、11 和 10 版本；以及 9.6.22 及更高的 9 版本
db.t3.medium	是	是	是	是	是	所有 PostgreSQL 17、16、15、14、13、12、11 和 10 版本；以及 9.6.22 及更高的 9 版本
db.t3.small	是	是	是	是	是	所有 PostgreSQL 17、16、15、14、13、12、11 和 10 版本；以及 9.6.22 及更高的 9 版本
db.t3.micro	否	是	是	是	仅在 Oracle Database 12c 版本 1（12.1.0.2）上，该版本已弃用	所有 PostgreSQL 17、16、15、14、13、12、11 和 10 版本；以及 9.6.22 及更高的 9 版本

db.t2 – 具爆发能力的实例类

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.t2.2xlarge	否	已弃用	否	已弃用	已弃用	已弃用
db.t2.xlarge	否	已弃用	否	已弃用	已弃用	已弃用
db.t2.large	否	已弃用	已弃用	已弃用	已弃用	已弃用
db.t2.medium	否	已弃用	已弃用	已弃用	已弃用	已弃用
db.t2.small	否	已弃用	已弃用	已弃用	已弃用	已弃用
db.t2.micro	否	已弃用	已弃用	已弃用	已弃用	已弃用

优化型读取实例类支持的数据库引擎

下表显示了优化型读取实例类支持的数据库和数据库版本。

db.r6gd – 支持优化型读取功能并由 Amazon Graviton2 处理器提供支持的内存优化型实例类

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
db.r6gd.16xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本

实例类	DI	MariaDB	Micros SQL Server	MySQL	Ora	PostgreSQL
db.r6gd.12xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本
db.r6gd.8xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本
db.r6gd.4xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本
db.r6gd.2xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本
db.r6gd.xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本
db.r6gd.large	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	否	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本、13.7 及更高的 13 版本和 13.4 版本

db.r6id – 支持优化型读取功能并由第三代 Intel Xeon 可扩展处理器提供支持的内存优化型实例类

实例类	Db2	MariaDB	Microso SQL Server	MySQL	Oracle	PostgreSQL
db.r6id.32xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	仅限 EE 和 BYOL	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本
db.r6id.24xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	仅限 EE 和 BYOL	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本
db.r6id.16xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	仅限 EE 和 BYOL	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本
db.r6id.12xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	仅限 EE 和 BYOL	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本
db.r6id.8xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	仅限 EE 和 BYOL	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本
db.r6id.4xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	仅限 BYOL	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14

实例类	Db2	MariaDB	Microsoft SQL Server	MySQL	Oracle	PostgreSQL
						版本，以及 13.7 及更高的 13 版本
db.r6id.2xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	仅限 BYOL	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本
db.r6id.xlarge	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	仅限 BYOL	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本
db.r6id.large	否	MariaDB 11.4、10.11、10.6、10.5 和 10.4	否	MySQL 8.4 和 8.0	仅限 BYOL	所有 PostgreSQL 17、16 和 15 版本、14.5 及更高的 14 版本，以及 13.7 及更高的 13 版本

确定 Amazon Web Services 区域 中的数据库实例类支持

要确定特定 Amazon Web Services 区域 中每个数据库引擎支持的数据库实例类，您可以采用以下几种方法之一。您可以使用 Amazon Web Services Management Console、Amazon RDS 定价页面或 Amazon Command Line Interface（Amazon CLI）的 describe-orderable-db-instance-options 命令。

> **Note**
>
> 当您使用 Amazon Web Services Management Console 执行操作时，它会自动显示特定数据库引擎、数据库引擎版本和 Amazon Web Services 区域 支持的数据库实例类。您可以执行的操作示例包括创建和修改数据库实例。

目录

使用 Amazon RDS 定价页面确定 Amazon Web Services 区域 中的数据库实例类支持

您可以使用 Amazon RDS 定价页面确定特定 Amazon Web Services 区域 中每个数据库引擎支持的数据库实例类。

使用定价页面确定区域中每个引擎支持的数据库实例类

1. 转到 Amazon RDS 定价。
2. 在 Amazon RDS 的 Amazon 定价计算器部分，选择立即创建您的自定义估算值。
3. 在选择区域中，选择一个 Amazon Web Services 区域。
4. 在查找服务中，输入 **`Amazon RDS`**。
5. 针对您的配置选项和数据库引擎选择配置。
6. 使用兼容实例部分查看支持的数据库实例类。
7. （可选）在计算器中选择其它选项，然后选择保存并查看摘要或保存并添加服务。

使用 Amazon CLI 确定 Amazon Web Services 区域 中的数据库实例类支持

您可以使用 Amazon CLI 来确定 Amazon Web Services 区域 中的特定数据库引擎和数据库引擎版本支持哪些数据库实例类。下表显示了有效的数据库引擎值。

引擎名称	CLI 命令中的引擎值	有关版本的更多信息
Db2	`db2-ae` `db2-se`	Amazon RDS 上的 Db2 版本
MariaDB	`mariadb`	Amazon RDS 上的 MariaDB 版本

引擎名称	CLI 命令中的引擎值	有关版本的更多信息

Microsoft SQL Server	`sqlserver-ee`
	`sqlserver-se`
	`sqlserver-ex`
	`sqlserver-web`
		Amazon RDS 上的 Microsoft SQL Server 版本

MySQL	`mysql`	Amazon RDS 上的 MySQL 版本
Oracle	`oracle-ee` `oracle-se2`	Amazon RDS for Oracle 发布说明
PostgreSQL	`postgres`	可用的 PostgreSQL 数据库版本

有关 Amazon Web Services 区域 名称的信息，请参阅 Amazon 地区。

以下示例演示了如何使用 describe-orderable-db-instance-options Amazon CLI 命令确定 Amazon Web Services 区域 中的数据库实例类支持。

> **Note**
> 为了限制输出，以下示例仅显示通用型 SSD (`gp2`) 存储类型的结果。如有必要，您可以在命令中将存储类型更改为通用型 SSD 存储 (`gp3`)、预调配 IOPS (`io1`)、预调配 IOPS Block Express (`io2`) 或磁介质 (`standard`)。

主题

列出 Amazon Web Services 区域 中特定数据库引擎版本支持的数据库实例类

要列出 Amazon Web Services 区域 中特定数据库引擎版本支持的数据库实例类，请运行以下命令。

对于 Linux、macOS 或 Unix：

```
aws rds describe-orderable-db-instance-options --engine engine --engine-version version
 \
    --query "*[].{DBInstanceClass:DBInstanceClass,StorageType:StorageType}|[?
StorageType=='gp2']|[].{DBInstanceClass:DBInstanceClass}" \
    --output text \
    --region region
```

对于 Windows：

```
aws rds describe-orderable-db-instance-options --engine engine --engine-version version
 ^
    --query "*[].{DBInstanceClass:DBInstanceClass,StorageType:StorageType}|[?
StorageType=='gp2']|[].{DBInstanceClass:DBInstanceClass}" ^
    --output text ^
    --region region
```

例如，以下命令列出了美国东部（弗吉尼亚州北部）的 RDS for PostgreSQL 数据库引擎 13.6 版支持的数据库实例类。

对于 Linux、macOS 或 Unix：

```
aws rds describe-orderable-db-instance-options --engine postgres --engine-version 15.4
 \
    --query "*[].{DBInstanceClass:DBInstanceClass,StorageType:StorageType}|[?
StorageType=='gp2']|[].{DBInstanceClass:DBInstanceClass}" \
    --output text \
    --region us-east-1
```

对于 Windows：

```
aws rds describe-orderable-db-instance-options --engine postgres --engine-version 15.4
 ^
    --query "*[].{DBInstanceClass:DBInstanceClass,StorageType:StorageType}|[?
StorageType=='gp2']|[].{DBInstanceClass:DBInstanceClass}" ^
    --output text ^
    --region us-east-1
```

列出支持 Amazon Web Services 区域 中特定数据库实例类的数据库引擎版本

要列出支持 Amazon Web Services 区域 中特定数据库实例类的数据库引擎版本，请运行以下命令。

对于 Linux、macOS 或 Unix：

```
aws rds describe-orderable-db-instance-options --engine engine --db-instance-
class DB_instance_class \
    --query "*[].{EngineVersion:EngineVersion,StorageType:StorageType}|[?
StorageType=='gp2']|[].{EngineVersion:EngineVersion}" \
    --output text \
    --region region
```

对于 Windows：

```
aws rds describe-orderable-db-instance-options --engine engine --db-instance-
class DB_instance_class ^
    --query "*[].{EngineVersion:EngineVersion,StorageType:StorageType}|[?
StorageType=='gp2']|[].{EngineVersion:EngineVersion}" ^
    --output text ^
    --region region
```

例如，以下命令列出了支持 US East (N. Virginia) 中 db.r5.large 数据库实例类的 RDS for PostgreSQL 数据库引擎的数据库引擎版本。

对于 Linux、macOS 或 Unix：

```
aws rds describe-orderable-db-instance-options --engine postgres --db-instance-class
 db.m7g.large \
    --query "*[].{EngineVersion:EngineVersion,StorageType:StorageType}|[?
StorageType=='gp2']|[].{EngineVersion:EngineVersion}" \
    --output text \
    --region us-east-1
```

对于 Windows：

```
aws rds describe-orderable-db-instance-options --engine postgres --db-instance-class
 db.m7g.large ^
    --query "*[].{EngineVersion:EngineVersion,StorageType:StorageType}|[?
StorageType=='gp2']|[].{EngineVersion:EngineVersion}" ^
    --output text ^
    --region us-east-1
```

列出支持特定数据库引擎和实例类的 Amazon Web Services 区域

以下 bash 脚本列出了所有支持指定数据库引擎和实例类组合的 Amazon Web Services 区域。

```
#!/usr/bin/env bash
# Usage: check_region_support.sh <db-engine> <db-instance-class>

if [ $# -ne 2 ]; then
  echo "Usage: $0 <db-engine> <db-instance-class>"
  exit 1
fi
ENGINE="$1"
INSTANCE_CLASS="$2"
REGIONS=$(aws ec2 describe-regions --query "Regions[].RegionName" --output text)
for region in $REGIONS; do
  supported_count=$(aws rds describe-orderable-db-instance-options \
    --region "$region" \
    --engine "$ENGINE" \
    --db-instance-class "$INSTANCE_CLASS" \
    --query 'length(OrderableDBInstanceOptions)' \
    --output text 2>/dev/null || echo "0")
  if [ "$supported_count" -gt 0 ]; then
    echo "$region supports $INSTANCE_CLASS for $ENGINE."
  else
    echo "$region doesn't support $INSTANCE_CLASS for $ENGINE."
  fi
done
```

以下示例输出使用 db.r8g.large 实例类检查了 RDS for MySQL 的区域支持。

```
./check_region_support.sh mysql db.r8g.large
ap-south-1 doesn't support db.r8g.large for mysql.
eu-north-1 doesn't support db.r8g.large for mysql.
eu-west-3 doesn't support db.r8g.large for mysql.
eu-west-2 doesn't support db.r8g.large for mysql.
eu-west-1 doesn't support db.r8g.large for mysql.
ap-northeast-3 doesn't support db.r8g.large for mysql.
ap-northeast-2 doesn't support db.r8g.large for mysql.
ap-northeast-1 doesn't support db.r8g.large for mysql.
ca-central-1 doesn't support db.r8g.large for mysql.
sa-east-1 doesn't support db.r8g.large for mysql.
ap-southeast-1 doesn't support db.r8g.large for mysql.
ap-southeast-2 doesn't support db.r8g.large for mysql.
eu-central-1 supports db.r8g.large for mysql.
us-east-1 supports db.r8g.large for mysql.
us-east-2 supports db.r8g.large for mysql.
us-west-1 doesn't support db.r8g.large for mysql.
```

```
us-west-2 supports db.r8g.large for mysql.
```

更改数据库实例类

通过更改数据库实例类，您可以更改数据库实例的可用 CPU 和内存。要更改数据库实例类，请按照 修改 Amazon RDS 数据库实例 中的说明修改数据库实例。

在 RDS for Oracle 中为数据库实例类配置处理器

Amazon RDS 数据库实例类支持 Intel 超线程技术，该技术可使多个线程在一个 Intel Xeon CPU 核心上并发运行。每个线程都表示为数据库实例上的一个虚拟 CPU (vCPU)。数据库实例具有默认数量的 CPU 核心，该数量因数据库实例类而异。例如，默认情况下，db.m4.xlarge DB 实例类有两个 CPU 核心，每个核心有两个线程 – 共四个 vCPU。

> **Note**
>
> 每个 vCPU 是 Intel Xeon CPU 核心的一个超线程。

主题

- 适用于 RDS for Oracle 的处理器配置概述
- 支持处理器配置的数据库实例类
- 为数据库实例类设置 CPU 核心数和每 CPU 核心线程数

适用于 RDS for Oracle 的处理器配置概述

当您使用 RDS for Oracle 时，您通常可以找到一个数据库实例类，它具有适合您工作负载的内存和 vCPU 数量的组合。但是，您也可以指定以下处理器特征，来针对特定工作负载或业务需求优化 RDS for Oracle 数据库实例：

- CPU 核心数 – 您可以自定义数据库实例的 CPU 核心数。也许可以通过这种方式让数据库实例拥有适合内存密集型工作负载的充足 RAM，同时减少 CPU 核心数，从而优化您的软件的许可成本。
- 每核心线程数 – 您可以通过为每个 CPU 核心指定一个线程来禁用 Intel 超线程技术。也许可以为特定工作负载（例如高性能计算 (HPC) 工作负载）执行该操作。

您可以单独控制 CPU 核心和每个核心的线程数量。您可以在请求中设置其中的一项或两项。在某一设置与数据库实例关联后，该设置将一直保留，直到对其进行更改。

数据库实例的处理器设置与数据库实例的快照关联。还原快照时，其还原的数据库实例将使用拍摄快照时使用的处理器特征设置。

如果用非默认的处理器设置来修改数据库实例的数据库实例类，则应指定默认处理器设置或在修改时明确指定处理器设置。此要求确保您在修改数据库实例时了解可能产生的第三方许可成本。

在 RDS for Oracle 数据库实例中指定处理器特征时，不会增加或减少费用。所收取的费用与使用默认 CPU 配置启动的数据库实例相同。

支持处理器配置的数据库实例类

只有在满足以下条件时，才能配置 CPU 核心数和每个核心的线程数：

- 您正在配置 RDS for Oracle 数据库实例。有关不同 Oracle 数据库版本支持的数据库实例类的信息，请参阅RDS for Oracle 数据库实例类。
- 您的数据库实例正在使用 RDS for Oracle 的自带许可（BYOL）许可选项。有关 Oracle 许可选项的更多信息，请参阅RDS for Oracle 许可选项。
- 您的数据库实例不属于具有预定义处理器配置的 db.r5 或 db.r5b 实例类之一。这些实例类的名称格式如下：db.r5.*instance_size*.tpc*threads_per_core*.mem*ratio* 或 db.r5b.*instance_size*.tpc*threads_per_core*.mem*ratio*。例如，db.r5b.xlarge.tpc2.mem4x 预配置为每个内核 2 个线程（tpc2）和标准 db.r5b.xlarge 实例类的 4 倍内存。您无法配置这些优化实例类的处理器特征。有关更多信息，请参阅 支持的 RDS for Oracle 数据库实例类。

您可以使用以下 Amazon CLI 命令来显示实例类的默认 vCPU、核心、每核心线程数和有效核心数。在示例命令中使用您的实例类的名称替换 *r7i.48xlarge*。

```
aws ec2 describe-instance-types \
    --instance-types r7i.48xlarge \
    --query '{
        DefaultVCPUs: InstanceTypes[0].VCpuInfo.DefaultVCpus,
        DefaultCores: InstanceTypes[0].VCpuInfo.DefaultCores,
        DefaultThreadsPerCore: InstanceTypes[0].VCpuInfo.DefaultThreadsPerCore,
        ValidCores: InstanceTypes[0].VCpuInfo.ValidCores
    }' \
    --output json
```

在下表中，您可以找到支持设置许多 CPU 核心和每核心 CPU 线程的数据库实例类。您还可以找到每个数据库实例类的 CPU 核心数和每核心 CPU 线程数的默认值和有效值。

数据库实例类	默认 vCPU	默认 CPU 核心数	默认每核心线程数	有效的 CPU 核心数	有效的每核心线程数
db.m6i – 内存优化型实例类					
db.m6i.large	2	1	2	1	1、2
db.m6i.xlarge	4	2	2	2	1、2
db.m6i.2xlarge	8	4	2	2、4	1、2
db.m6i.4xlarge	16	8	2	2、4、6、8	1、2
db.m6i.4xlarge	16	8	2	2、4、6、8	1、2
db.m6i.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
db.m6i.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
db.m6i.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
db.m6i.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
db.m6i.32xlarge	128	64	2	2、4、6、8、10、12、14、16、18、20、2	1、2

数据库实例类	默认 vCPU	默认 CPU 核心数	默认每核心线程数	有效的 CPU 核心数	有效的每核心线程数
				2、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	
db.m5 – 通用实例类					
db.m5.large	2	1	2	1	1、2
db.m5.xlarge	4	2	2	2	1、2
db.m5.2xlarge	8	4	2	2、4	1、2
db.m5.4xlarge	16	8	2	2、4、6、8	1、2
db.m5.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
db.m5.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
db.m5.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2

数据库实例类	默认 vCPU	默认 CPU 核心数	默认每核心线程数	有效的 CPU 核心数	有效的每核心线程数
db.m5.24xlarge	96	48	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
db.m5d – 通用实例类					
db.m5d.large	2	1	2	1	1、2
db.m5d.xlarge	4	2	2	2	1、2
db.m5d.2xlarge	8	4	2	2、4	1、2
db.m5d.4xlarge	16	8	2	2、4、6、8	1、2
db.m5d.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
db.m5d.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
db.m5d.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2

数据库实例类	默认 vCPU	默认 CPU 核心数	默认每核心线程数	有效的 CPU 核心数	有效的每核心线程数
db.m5d.24xlarge	96	48	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
db.m4 – 通用实例类					
db.m4.10xlarge	40	20	2	2、4、6、8、10、12、14、16、18、20	1、2
db.m4.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
db.r7i – 内存优化型实例类					
db.r7i.large	2	1	2	1	1、2
db.r7i.xlarge	4	2	2	1、2	1、2
db.r7i.2xlarge	8	4	2	1、2、3、4	1、2
db.r7i.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
db.r7i.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2

数据库实例类	默认 vCPU	默认 CPU 核心数	默认每核心线程数	有效的 CPU 核心数	有效的每核心线程数
db.r7i.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
db.r7i.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2
db.r7i.24xlarge	96	48	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1、2

数据库实例类	默认 vCPU	默认 CPU 核心数	默认每核心线程数	有效的 CPU 核心数	有效的每核心线程数
db.r7i.48xlarge	192	96	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96	1、2
db.r6i – 内存优化型实例类					
db.r6i.large	2	1	2	1	1、2
db.r6i.xlarge	4	2	2	1、2	1、2
db.r6i.2xlarge	8	4	2	2、4	1、2
db.r6i.4xlarge	16	8	2	2、4、6、8	1、2
db.r6i.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
db.r6i.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2

数据库实例类	默认 vCPU	默认 CPU 核心数	默认每核心线程数	有效的 CPU 核心数	有效的每核心线程数
db.r6i.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
db.r6i.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
db.r6i.32xlarge	128	64	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
db.r5 – 内存优化型实例类					
db.r5.large	2	1	2	1	1、2
db.r5.xlarge	4	2	2	2	1、2
db.r5.2xlarge	8	4	2	2、4	1、2
db.r5.4xlarge	16	8	2	2、4、6、8	1、2

数据库实例类	默认 vCPU	默认 CPU 核心数	默认每核心线程数	有效的 CPU 核心数	有效的每核心线程数
db.r5.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
db.r5.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
db.r5.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
db.r5.24xlarge	96	48	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
db.r5 – 内存优化型实例类					
db.r5b.large	2	1	2	1	1、2
db.r5b.xlarge	4	2	2	2	1、2
db.r5b.2xlarge	8	4	2	2、4	1、2
db.r5b.4xlarge	16	8	2	2、4、6、8	1、2
db.r5b.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2

数据库实例类	默认 vCPU	默认 CPU 核心数	默认每核心线程数	有效的 CPU 核心数	有效的每核心线程数
db.r5b.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
db.r5b.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
db.r5b.24xlarge	96	48	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
db.r5d – 内存优化型实例类					
db.r5d.large	2	1	2	1	1、2
db.r5d.xlarge	4	2	2	2	1、2
db.r5d.2xlarge	8	4	2	2、4	1、2
db.r5d.4xlarge	16	8	2	2、4、6、8	1、2
db.r5d.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
db.r5d.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2

数据库实例类	默认 vCPU	默认 CPU 核心数	默认每核心线程数	有效的 CPU 核心数	有效的每核心线程数
db.r5d.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
db.r5d.24xlarge	96	48	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
db.r4 – 内存优化型实例类					
db.r4.large	2	1	2	1	1、2
db.r4.xlarge	4	2	2	1、2	1、2
db.r4.2xlarge	8	4	2	1、2、3、4	1、2
db.r4.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
db.r4.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
db.r4.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
db.r3 – 内存优化型实例类					

数据库实例类	默认 vCPU	默认 CPU 核心数	默认每核心线程数	有效的 CPU 核心数	有效的每核心线程数
db.r3.large	2	1	2	1	1、2
db.r3.xlarge	4	2	2	1、2	1、2
db.r3.2xlarge	8	4	2	1、2、3、4	1、2
db.r3.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
db.r3.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
db.x2idn – 内存优化型实例类					
db.x2idn.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
db.x2idn.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2

数据库实例类	默认 vCPU	默认 CPU 核心数	默认每核心线程数	有效的 CPU 核心数	有效的每核心线程数
db.x2idn.32xlarge	128	64	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
db.x2iedn – 内存优化型实例类					
db.x2iedn.xlarge	4	2	2	1、2	1、2
db.x2iedn.2xlarge	8	4	2	2、4	1、2
db.x2iedn.4xlarge	16	8	2	2、4、6、8	1、2
db.x2iedn.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
db.x2iedn.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2

数据库实例类	默认 vCPU	默认 CPU 核心数	默认每核心线程数	有效的 CPU 核心数	有效的每核心线程数
db.x2iedn.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
db.x2iedn.32xlarge	128	64	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
db.x2iezn – 内存优化型实例类					
db.x2iezn.2xlarge	8	4	2	2、4	1、2
db.x2iezn.4xlarge	16	8	2	2、4、6、8	1、2
db.x2iezn.6xlarge	24	12	2	2、4、6、8、10、12	1、2
db.x2iezn.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2

数据库实例类	默认 vCPU	默认 CPU 核心数	默认每核心线程数	有效的 CPU 核心数	有效的每核心线程数
db.x2iezn.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
db.x1 – 内存优化型实例类					
db.x1.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
db.x1.32xlarge	128	64	2	4、8、12、16、20、24、28、32、36、40、44、48、52、56、60、64	1、2
db.x1e – 内存优化型实例类					
db.x1e.xlarge	4	2	2	1、2	1、2
db.x1e.2xlarge	8	4	2	1、2、3、4	1、2
db.x1e.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
db.x1e.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
db.x1e.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2

数据库实例类	默认 vCPU	默认 CPU 核心数	默认每核心线程数	有效的 CPU 核心数	有效的每核心线程数
db.x1e.32xlarge	128	64	2	4、8、12、16、20、24、28、32、36、40、44、48、52、56、60、64	1、2
db.z1d – 内存优化型实例类					
db.z1d.large	2	1	2	1	1、2
db.z1d.xlarge	4	2	2	2	1、2
db.z1d.2xlarge	8	4	2	2、4	1、2
db.z1d.3xlarge	12	6	2	2、4、6	1、2
db.z1d.6xlarge	24	12	2	2、4、6、8、10、12	1、2
db.z1d.12xlarge	48	24	2	4、6、8、10、12、14、16、18、20、22、24	1、2

> **Note**
>
> 您可以使用 Amazon CloudTrail 监控和审核对 Oracle 数据库实例的 Amazon RDS for Oracle 的流程配置进行的更改。有关使用 CloudTrail 的更多信息，请参阅监控 Amazon CloudTrail 中的 Amazon RDS API 调用。

为数据库实例类设置 CPU 核心数和每 CPU 核心线程数

可在执行以下操作时为数据库实例类配置 CPU 核心数和每核心线程数：

- 创建 Amazon RDS 数据库实例

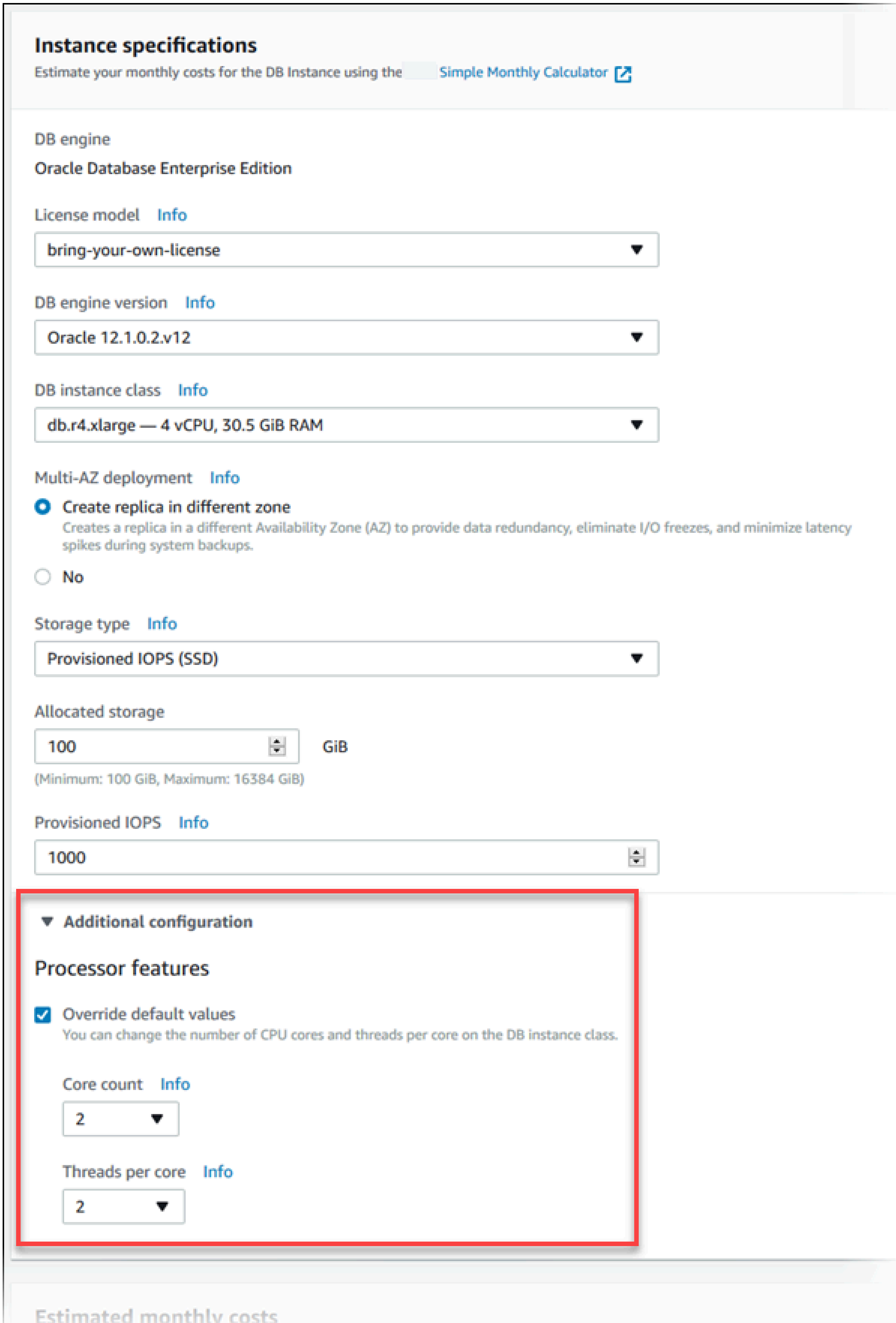
- 修改 Amazon RDS 数据库实例
- 还原到数据库实例
- 将 Amazon RDS 的数据库实例还原到指定时间

> **Note**
> 在修改数据库实例以配置 CPU 核心数或每核心线程数时，会出现短暂的数据库实例中断。

可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 设置数据库实例类的 CPU 核心数和每 CPU 核心线程数。

控制台

在创建、修改或还原数据库实例时，可在 Amazon Web Services Management Console 中设置数据库实例类。Instance specifications（实例规格）部分显示用于处理器的选项。下图显示处理器特征选项。

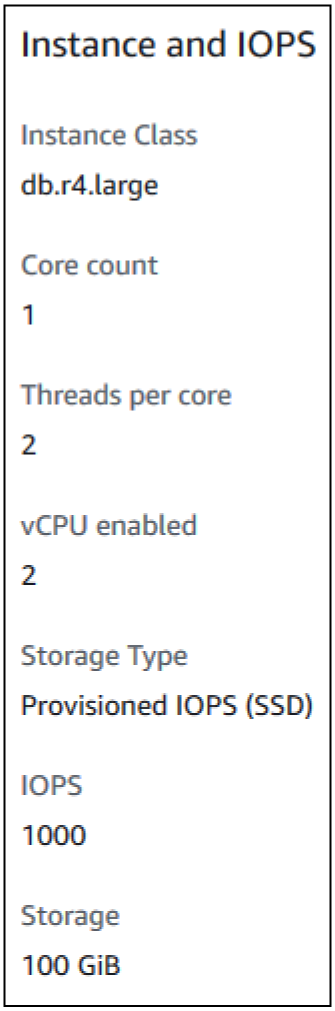

Instance specifications
Estimate your monthly costs for the DB Instance using the Simple Monthly Calculator
DB engine
Oracle Database Enterprise Edition
License model Info
bring-your-own-license
DB engine version Info
Oracle 12.1.0.2.v12
DB instance class Info
db.r4.xlarge — 4 vCPU, 30.5 GiB RAM
Multi-AZ deployment Info
Create replica in different zone
Creates a replica in a different Availability Zone (AZ) to provide data redundancy, eliminate I/O freezes, and minimize latency spikes during system backups.
No
Storage type Info
Provisioned IOPS (SSD)
Allocated storage
100
GiB
(Minimum: 100 GiB, Maximum: 16384 GiB)
Provisioned IOPS Info
1000
Additional configuration
Processor features
Override default values
You can change the number of CPU cores and threads per core on the DB instance class.
Core count Info
2
Threads per core Info
2
Estimated monthly costs

在 Processor features (处理器特征) 下将数据库实例类的以下选项设置为合适的值：

- Core count (核心计数) – 使用此选项设置 CPU 核心数。该值必须等于或小于数据库实例类的最大 CPU 核心数。
- Threads per core (每核心线程数) – 指定 2 以启用每核心多个线程，或指定 1 以禁用每核心多个线程。

在修改或还原数据库实例时，还可以将 CPU 核心数和每核心线程数设置为实例类的默认设置。

在控制台中查看数据库实例的详细信息时，您可以在 Configuration (配置) 选项卡上查看其数据库实例类的处理器信息。下图显示启用了一个 CPU 核心和每核心多个实例的数据库实例类。



对于 Oracle 数据库实例，只有自带许可 (BYOL) 数据库实例显示处理器信息。

Amazon CLI

在运行以下 Amazon CLI 命令之一时，可为数据库实例设置处理器特征：

- [create-db-instance]
- [modify-db-instance]
- [restore-db-instance-from-db-snapshot]
- [restore-db-instance-from-s3]
- [restore-db-instance-to-point-in-time]

要使用 Amazon CLI 为数据库实例的数据库实例类配置处理器，请在命令中包含 `--processor-features` 选项。使用 `coreCount` 特征名称指定 CPU 核心数，并使用 `threadsPerCore` 特征名称指定是否启用每核心多个线程。

该选项具有以下语法。

```
--processor-features "Name=coreCount,Value=<value>" "Name=threadsPerCore,Value=<value>"
```

下面是配置处理器的示例：

示例

- 为数据库实例设置 CPU 核心数
- 为数据库实例设置 CPU 核心数并禁用多个线程
- 查看数据库实例类的有效处理器值
- 返回数据库实例的默认处理器设置
- 返回数据库实例的默认 CPU 核心数
- 返回数据库实例的默认每核心线程数

为数据库实例设置 CPU 核心数

Example

以下示例通过将 CPU 核心数设置为 4 来修改 `mydbinstance`。使用 `--apply-immediately` 可立即应用更改。如果要在下一个计划的维护时段应用更改，请省略 `--apply-immediately` 选项。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier mydbinstance \
    --processor-features "Name=coreCount,Value=4" \
```

```
    --apply-immediately
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier mydbinstance ^
    --processor-features "Name=coreCount,Value=4" ^
    --apply-immediately
```

为数据库实例设置 CPU 核心数并禁用多个线程

Example

以下示例通过将 CPU 核心数设置为 mydbinstance 并禁用每核心多个线程来修改 4。使用 `--apply-immediately` 可立即应用更改。如果要在下一个计划的维护时段应用更改，请省略 `--apply-immediately` 选项。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier mydbinstance \
    --processor-features "Name=coreCount,Value=4" "Name=threadsPerCore,Value=1" \
    --apply-immediately
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier mydbinstance ^
    --processor-features "Name=coreCount,Value=4" "Name=threadsPerCore,Value=1" ^
    --apply-immediately
```

查看数据库实例类的有效处理器值

Example

可以通过运行 [describe-orderable-db-instance-options](https://docs.aws.amazon.com/cli/latest/reference/rds/describe-orderable-db-instance-options.html) 命令并为 `--db-instance-class` 选项指定实例类来查看特定数据库实例类的有效处理器值。例如，以下命令的输出显示用于 db.r3.large 实例类的处理器选项。

```
aws rds describe-orderable-db-instance-options --engine oracle-ee --db-instance-class
 db.r3.large
```

以下为 JSON 格式的命令示例输出。


```
{
            "SupportsIops": true,
            "MaxIopsPerGib": 50.0,
            "LicenseModel": "bring-your-own-license",
            "DBInstanceClass": "db.r3.large",
            "SupportsIAMDatabaseAuthentication": false,
            "MinStorageSize": 100,
            "AvailabilityZones": [
                {
                    "Name": "us-west-2a"
                },
                {
                    "Name": "us-west-2b"
                },
                {
                    "Name": "us-west-2c"
                }
            ],
            "EngineVersion": "12.1.0.2.v2",
            "MaxStorageSize": 32768,
            "MinIopsPerGib": 1.0,
            "MaxIopsPerDbInstance": 40000,
            "ReadReplicaCapable": false,
            "AvailableProcessorFeatures": [
                {
                    "Name": "coreCount",
                    "DefaultValue": "1",
                    "AllowedValues": "1"
                },
                {
                    "Name": "threadsPerCore",
                    "DefaultValue": "2",
                    "AllowedValues": "1,2"
                }
            ],
            "SupportsEnhancedMonitoring": true,
            "SupportsPerformanceInsights": false,
            "MinIopsPerDbInstance": 1000,
            "StorageType": "io1",
            "Vpc": false,
            "SupportsStorageEncryption": true,
            "Engine": "oracle-ee",
```

```
                "MultiAZCapable": true
    }
```

此外，可以运行以下命令来获取数据库实例类处理器信息：

- describe-db-instances – 显示指定数据库实例的处理器信息。
- describe-db-snapshots – 显示指定数据库快照的处理器信息。
- describe-valid-db-instance-modifications – 显示对指定数据库实例的处理器的有效修改。

在上述命令的输出中，只有在满足以下条件时，处理器特征的值才不为 Null：

- 您正在使用 RDS for Oracle 数据库实例。
- 您的 RDS for Oracle 数据库实例支持更改处理器值。
- 当前 CPU 核心和线程设置设置为非默认值。

如果不满足上述条件，您可以使用 describe-db-instances 获取实例类型。您可以通过运行 EC2 操作 describe-instance-types 来获取此实例类型的处理器信息。

返回数据库实例的默认处理器设置

Example

以下示例通过将数据库实例的数据库实例类返回给默认处理器值来修改 `mydbinstance`。使用 `--apply-immediately` 可立即应用更改。如果要在下一个计划的维护时段应用更改，请省略 `--apply-immediately` 选项。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier mydbinstance \
    --use-default-processor-features \
    --apply-immediately
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier mydbinstance ^
    --use-default-processor-features ^
    --apply-immediately
```

返回数据库实例的默认 CPU 核心数

Example

以下示例通过将数据库实例的数据库实例类返回给默认 CPU 核心数来修改 mydbinstance。每核心线程数未更改。使用 --apply-immediately 可立即应用更改。如果要在下一个计划的维护时段应用更改，请省略 --apply-immediately 选项。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier mydbinstance \
    --processor-features "Name=coreCount,Value=DEFAULT" \
    --apply-immediately
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier mydbinstance ^
    --processor-features "Name=coreCount,Value=DEFAULT" ^
    --apply-immediately
```

返回数据库实例的默认每核心线程数

Example

以下示例通过将数据库实例的数据库实例类返回给默认每核心线程数来修改 mydbinstance。CPU 核心数设置未更改。使用 --apply-immediately 可立即应用更改。如果要在下一个计划的维护时段应用更改，请省略 --apply-immediately 选项。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier mydbinstance \
    --processor-features "Name=threadsPerCore,Value=DEFAULT" \
    --apply-immediately
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier mydbinstance ^
    --processor-features "Name=threadsPerCore,Value=DEFAULT" ^
```

```
    --apply-immediately
```

RDS API

在调用以下 Amazon RDS API 操作之一时，可为数据库实例设置处理器特征：

- CreateDBInstance
- ModifyDBInstance
- RestoreDBInstanceFromDBSnapshot
- RestoreDBInstanceFromS3
- RestoreDBInstanceToPointInTime

要使用 Amazon RDS API 为数据库实例的数据库实例类配置处理器特征，请在该调用中包含 `ProcessFeatures` 参数。

该参数具有以下语法。

```
ProcessFeatures "Name=coreCount,Value=<value>" "Name=threadsPerCore,Value=<value>"
```

使用 `coreCount` 特征名称指定 CPU 核心数，并使用 `threadsPerCore` 特征名称指定是否启用每核心多个线程。

可以通过运行 DescribeOrderableDBInstanceOptions 操作并为 `DBInstanceClass` 参数指定实例类来查看特定数据库实例类的有效处理器值。您还可以使用以下操作：

- DescribeDBInstances – 显示指定数据库实例的处理器信息。
- DescribeDBSnapshots – 显示指定数据库快照的处理器信息。
- DescribeValidDBInstanceModifications – 显示对指定数据库实例的处理器的有效修改。

在上述操作的输出中，只有在满足以下条件时，处理器特征的值才不为 Null：

- 您正在使用 RDS for Oracle 数据库实例。
- 您的 RDS for Oracle 数据库实例支持更改处理器值。
- 当前 CPU 核心和线程设置设置为非默认值。

如果不满足上述条件，您可以使用 DescribeDBInstances 获取实例类型。您可以通过运行 EC2 操作 DescribeInstanceTypes 来获取此实例类型的处理器信息。

数据库实例类的硬件规格

在本节的表中，可以找到有关 Amazon RDS 数据库实例类的硬件详细信息。

有关每个数据库实例类的 Amazon RDS 数据库引擎支持的信息，请参阅数据库实例类支持的数据库引擎。

主题

数据库实例类的硬件术语

以下术语用于描述数据库实例类的硬件规格。

vCPU

虚拟中央处理器（CPU）的数量。虚拟 CPU 是可用于比较数据库实例类的容量单位。您不再购买或租用特定的处理器并用上数月或数年，而是以小时为单位租用容量。我们的目标是在实际基础硬件的限制内提供特定数量的一致 CPU 容量。

ECU

Amazon EC2 实例的整数处理能力的相对度量。为了便于开发人员比较不同实例类的 CPU 容量，我们定义了一个 Amazon EC2 计算单位。分配给特定实例的 CPU 量是以这些 EC2 计算单位来表示的。一个 ECU 目前提供的 CPU 容量相当于 1.0–1.2 GHz 2007 Opteron 或 2007 Xeon 处理器。

内存 (GiB)

为数据库实例分配的 RAM (GiB)。内存与 vCPU 之间通常具有一致的比率。以 db.r4 实例类为例，其具有类似于 db.r5 实例类的内存 vCPU 比。但是，对于大多数使用案例，db.r5 实例类会提供比 db.r4 实例类更好、更一致的性能。

EBS 优化

数据库实例使用经过优化的配置堆栈，并为输入/输出提供额外的专用容量。这种优化通过最小化输入/输出与来自您实例的其他流量之间的争用，为您提供最佳性能。有关 Amazon EBS 优化型实例的更多信息，请参阅《Amazon EC2 用户指南》中的 Amazon EBS 优化型实例。

EBS 优化的实例具有基准和最大 IOPS 速率。最大 IOPS 速率是在数据库实例级别强制执行的。一组 EBS 卷加起来的 IOPS 速率高于最大值，但不能超过实例级阈值。例如，如果特定数据库实例类的最大 IOPS 为 40000，而您连接了四个 64000 IOPS EBS 卷，则最大 IOPS 为 40000 而不是 256000。有关特定于每种 EC2 实例类型的 IOPS 最大值，请参阅《Amazon EC2 用户指南（适用于 Linux 实例）》中的支持的实例类型。

最大 EBS 带宽（Mbps）

以 MB/s 为单位的最大 EBS 带宽。除以 8 可获得预期吞吐量（MB/s）。

> **Important**
>
> 在大多数情况下，Amazon RDS 数据库实例的通用型 SSD (gp2) 卷的吞吐量限制为 250 MiB/s。但是，吞吐量限制可能会因卷大小而异。有关更多信息，请参阅《Amazon EC2 用户指南》中的 Amazon EBS 卷类型。

网络带宽

与其他数据库实例类有关的网络速度。

通用实例类的硬件规格

下表显示了通用实例类的计算、内存、存储和带宽规格。

db.m8g – 由 Amazon Graviton4 处理器提供支持的通用实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.m8g.48xlarge	192	—	768	仅 EBS 优化	40000	50
db.m8g.24xlarge	96	—	512	仅 EBS 优化	30000	40
db.m8g.16xlarge	64	—	384	仅 EBS 优化	20000	30
db.m8g.12xlarge	48	—	256	仅 EBS 优化	15000	22.5
db.m8g.8xlarge	32	—	128	仅 EBS 优化	10000	15
db.m8g.4xlarge*	16	—	64	仅 EBS 优化	最高 10,000	最多 15

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.m8g.2xlarge*	8	—	32	仅 EBS 优化	最高 10,000	最多 15
db.m8g.xlarge*	4	—	16	仅 EBS 优化	最高 10,000	最多 12.5
db.m8g.large*	2	—	8	仅 EBS 优化	最高 10,000	最多 12.5

db.m7i – 由第四代 Intel Xeon 可扩展处理器提供支持的通用实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.m7i.48xlarge	192	—	768	仅 EBS 优化	40000	50
db.m7i.24xlarge	96	—	384	仅 EBS 优化	30000	37.5
db.m7i.16xlarge	64	—	256	仅 EBS 优化	20000	25
db.m7i.12xlarge	48	—	192	仅 EBS 优化	15000	18.75
db.m7i.8xlarge	32	—	128	仅 EBS 优化	10000	12.5
db.m7i.4xlarge	16	—	64	仅 EBS 优化	最高 10,000	最多 12.5
db.m7i.2xlarge	8	—	32	仅 EBS 优化	最高 10,000	最多 12.5
db.m7i.xlarge	4	—	16	仅 EBS 优化	最高 10,000	最多 12.5
db.m7i.large	2	—	8	仅 EBS 优化	最高 10,000	最多 12.5

db.m7g – 由 Amazon Graviton3 处理器提供支持的通用实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.m7g.16xlarge	64	—	256	仅 EBS 优化	20000	30

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.m7g.12xlarge	48	—	192	仅 EBS 优化	15000	22.5
db.m7g.8xlarge	32	—	128	仅 EBS 优化	10000	15
db.m7g.4xlarge	16	—	64	仅 EBS 优化	最高 10,000	最多 15
db.m7g.2xlarge*	8	—	32	仅 EBS 优化	最高 10,000	最多 15
db.m7g.xlarge*	4	—	16	仅 EBS 优化	最高 10,000	最多 12.5
db.m7g.large*	2	—	8	仅 EBS 优化	最高 10,000	最多 12.5

db.m6g – 由 Amazon Graviton2 处理器提供支持的通用实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.m6g.16xlarge	64	—	256	仅 EBS 优化	19000	25
db.m6g.12xlarge	48	—	192	仅 EBS 优化	13,500	20
db.m6g.8xlarge	32	—	128	仅 EBS 优化	9,000	12
db.m6g.4xlarge	16	—	64	仅 EBS 优化	4750	最多 10
db.m6g.2xlarge*	8	—	32	仅 EBS 优化	最多 4,750	最多 10
db.m6g.xlarge*	4	—	16	仅 EBS 优化	最多 4,750	最多 10
db.m6g.large*	2	—	8	仅 EBS 优化	最多 4,750	最多 10

db.m6gd – 由 Amazon Graviton2 处理器和 SSD 存储提供支持的通用实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.m6gd.16xlarge	64	—	256	2 x 1,900 NVMe SSD	19000	25
db.m6gd.12xlarge	48	—	192	2 x 1,425 NVMe SSD	13,500	20
db.m6gd.8xlarge	32	—	128	1 x 1,900 NVMe SSD	9,000	12
db.m6gd.4xlarge*	16	—	64	1 x 950 NVMe SSD	4750	最多 10
db.m6gd.2xlarge*	8	—	32	1 x 474 NVMe SSD	最多 4,750	最多 10
db.m6gd.xlarge*	4	—	16	1 x 237 NVMe SSD	最多 4,750	最多 10
db.m6gd.large*	2	—	8	1 x 118 NVMe SSD	最多 4,750	最多 10

db.m6id – 由第三代 Intel Xeon 可扩展处理器和 SSD 存储提供支持的通用实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.m6id.32xlarge	128	—	512	4 x 1,900 NVMe SSD	40000	50
db.m6id.24xlarge	96	—	384	4 x 1,425 NVMe SSD	30000	37.5
db.m6id.16xlarge	64	—	256	2 x 1900 NVMe SSD	20000	25

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.m6id.12xlarge	48	—	192	2 x 1425 NVMe SSD	15000	18.75
db.m6id.8xlarge	32	—	128	1 x 1900 NVMe SSD	10000	12.5
db.m6id.4xlarge*	16	—	64	1 x 950 NVMe SSD	最高 10,000	最多 12.5
db.m6id.2xlarge*	8	—	32	1 x 474 NVMe SSD	最高 10,000	最多 12.5
db.m6id.xlarge*	4	—	16	1 x 237 NVMe SSD	最高 10,000	最多 12.5
db.m6id.large*	2	—	8	1 x 118 NVMe SSD	最高 10,000	最多 12.5

db.m6idn – 具有第三代 Intel Xeon 可扩展处理器、SSD 存储和网络优化的通用实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.m6idn.32xlarge	128	—	512	4 x 1,900 NVMe SSD	80,000	200
db.m6idn.24xlarge	96	—	384	4 x 1,425 NVMe SSD	60000	150
db.m6idn.16xlarge	64	—	256	2 x 1,900 NVMe SSD	40000	100
db.m6idn.12xlarge	48	—	192	2 x 1,425 NVMe SSD	30000	75

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.m6idn.8xlarge	32	—	128	1 x 1,900 NVMe SSD	20000	50
db.m6idn.4xlarge*	16	—	64	1 x 950 NVMe SSD	最多 20,000	最多 50
db.m6idn.2xlarge*	8	—	32	1 x 474 NVMe SSD	最多 20,000	最多 40
db.m6idn.xlarge*	4	—	16	1 x 237 NVMe SSD	最多 20,000	最多 30
db.m6idn.large*	2	—	8	1 x 118 NVMe SSD	最多 20,000	最多 25

db.m6in – 由第三代 Intel Xeon 可扩展处理器和网络优化提供支持的通用实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.m6in.32xlarge	128	—	512	仅 EBS 优化	80,000	200
db.m6in.24xlarge	96	—	384	仅 EBS 优化	60000	150
db.m6in.16xlarge	64	—	256	仅 EBS 优化	40000	100
db.m6in.12xlarge	48	—	192	仅 EBS 优化	30000	75
db.m6in.8xlarge	32	—	128	仅 EBS 优化	20000	50
db.m6in.4xlarge*	16	—	64	仅 EBS 优化	最多 20,000	最多 50
db.m6in.2xlarge*	8	—	32	仅 EBS 优化	最多 20,000	最多 40
db.m6in.xlarge*	4	—	16	仅 EBS 优化	最多 20,000	最多 30

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.m6in.large*	2	—	8	仅 EBS 优化	最多 20,000	最多 25

db.m6i – 由第三代 Intel Xeon 可扩展处理器提供支持的通用实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.m6i.32xlarge	128	—	512	仅 EBS 优化	40000	50
db.m6i.24xlarge	96	—	384	仅 EBS 优化	30000	37.5
db.m6i.16xlarge	64	—	256	仅 EBS 优化	20000	25
db.m6i.12xlarge	48	—	192	仅 EBS 优化	15000	18.75
db.m6i.8xlarge	32	—	128	仅 EBS 优化	10000	12.5
db.m6i.4xlarge*	16	—	64	仅 EBS 优化	最高 10,000	最多 12.5
db.m6i.2xlarge*	8	—	32	仅 EBS 优化	最高 10,000	最多 12.5
db.m6i.xlarge*	4	—	16	仅 EBS 优化	最高 10,000	最多 12.5
db.m6i.large*	2	—	8	仅 EBS 优化	最高 10,000	最多 12.5

db.m5d – 由 Intel Xeon Platinum 处理器和 SSD 存储提供支持的通用实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.m5d.24xlarge	96	345	384	4 x 900 NVMe SSD	19000	25
db.m5d.16xlarge	64	262	256	4 x 600 NVMe SSD	13600	20

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.m5d.12xlarge	48	173	192	2 x 900 NVMe SSD	9500	10
db.m5d.8xlarge	32	131	128	2 x 600 NVMe SSD	6800	10
db.m5d.4xlarge	16	61	64	2 x 300 NVMe SSD	4750	最多 10
db.m5d.2xlarge*	8	31	32	1 x 300 NVMe SSD	最多 4,750	最多 10
db.m5d.xlarge*	4	15	16	1 x 150 NVMe SSD	最多 4,750	最多 10
db.m5d.large*	2	10	8	1 x 75 NVMe SSD	最多 4,750	最多 10

db.m5 – 具有 Intel Xeon Platinum 处理器的通用实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.m5.24xlarge	96	345	384	仅 EBS 优化	19000	25
db.m5.16xlarge	64	262	256	仅 EBS 优化	13600	20
db.m5.12xlarge	48	173	192	仅 EBS 优化	9500	10
db.m5.8xlarge	32	131	128	仅 EBS 优化	6800	10
db.m5.4xlarge	16	61	64	仅 EBS 优化	4750	最多 10
db.m5.2xlarge*	8	31	32	仅 EBS 优化	最多 4,750	最多 10
db.m5.xlarge*	4	15	16	仅 EBS 优化	最多 4,750	最多 10

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.m5.large*	2	10	8	仅 EBS 优化	最多 4,750	最多 10

db.m4 – 具有 Intel Xeon 可扩展处理器的通用实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.m4.16xlarge	64	188	256	仅 EBS 优化	10000	25
db.m4.10xlarge	40	124.5	160	仅 EBS 优化	4,000	10
db.m4.4xlarge	16	53.5	64	仅 EBS 优化	2000	高
db.m4.2xlarge	8	25.5	32	仅 EBS 优化	1000	高
db.m4.xlarge	4	13	16	仅 EBS 优化	750	高
db.m4.large	2	6.5	8	仅 EBS 优化	450	中

db.m3 – 通用实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.m3.2xlarge	8	26	30	仅 EBS 优化	1000	高
db.m3.xlarge	4	13	15	仅 EBS 优化	500	高
db.m3.large	2	6.5	7.5	仅限于 EBS	—	中
db.m3.medium	1	3	3.75	仅限于 EBS	—	中

* 这些数据库实例类可以至少每 24 小时支持一次 30 分钟的最高性能。有关基础 EC2 实例类型的基准性能的更多信息，请参阅《Amazon EC2 用户指南》中的 Amazon EBS 优化的实例。

内存优化型实例类的硬件规格

下表显示了内存优化型实例类的计算、内存、存储和带宽规格。

db.z1d – 内存优化型实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.z1d.12xlarge	48	271	384	2 x 900 NVMe SSD	14,000	25
db.z1d.6xlarge	24	134	192	1 x 900 NVMe SSD	7,000	10
db.z1d.3xlarge	12	75	96	1 x 450 NVMe SSD	3,500	最多 10
db.z1d.2xlarge	8	53	64	1 x 300 NVMe SSD	2,333	最多 10
db.z1d.xlarge*	4	28	32	1 x 150 NVMe SSD	最高 2,333	最多 10
db.z1d.large*	2	15	16	1 x 75 NVMe SSD	最高 2,333	最多 10

db.x2g – 具有 Amazon Graviton2 处理器的内存优化型实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.x2g.16xlarge	64	—	1024	仅 EBS 优化	19000	25
db.x2g.12xlarge	48	—	768	仅 EBS 优化	14,250	20
db.x2g.8xlarge	32	—	512	仅 EBS 优化	9500	12
db.x2g.4xlarge	16	—	256	仅 EBS 优化	4750	最多 10

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.x2g.2xlarge	8	—	128	仅 EBS 优化	最多 4,750	最多 10
db.x2g.xlarge	4	—	64	仅 EBS 优化	最多 4,750	最多 10
db.x2g.large	2	—	32	仅 EBS 优化	最多 4,750	最多 10

db.x2idn – 具有第三代 Intel Xeon 可扩展处理器的内存优化型实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.x2idn.32xlarge	128	—	2,048	2 x 1900 NVMe SSD	80,000	100
db.x2idn.24xlarge	96	—	1,536	2 x 1425 NVMe SSD	60000	75
db.x2idn.16xlarge	64	—	1024	1 x 1900 NVMe SSD	40000	50

db.x2iedn – 使用基于 NVMe 的本地 SSD 且具有第三代 Intel Xeon 可扩展处理器的内存优化型实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.x2iedn.32xlarge	128	—	4,096	2 x 1900 NVMe SSD	80,000	100
db.x2iedn.24xlarge	96	—	3072	2 x 1425 NVMe SSD	60000	75
db.x2iedn.16xlarge	64	—	2,048	1 x 1900 NVMe SSD	40000	50

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.x2iedn.8xlarge	32	—	1024	1 x 950 NVMe SSD	20000	25
db.x2iedn.4xlarge	16	—	512	1 x 475 NVMe SSD	最多 20,000	最多 25
db.x2iedn.2xlarge	8	—	256	1 x 237 NVMe SSD	最多 20,000	最多 25
db.x2iedn.xlarge	4	—	128	1 x 118 NVMe SSD	最多 20,000	最多 25

db.x2iezn – 具有第二代 Intel Xeon 可扩展处理器的内存优化型实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.x2iezn.12xlarge	>48	—	1,536	仅 EBS 优化	19000	100
db.x2iezn.8xlarge	32	—	1024	仅 EBS 优化	12000	75
db.x2iezn.6xlarge	24	—	768	仅 EBS 优化	最多 9,500	50
db.x2iezn.4xlarge	16	—	512	仅 EBS 优化	最多 4,750	最多 25
db.x2iezn.2xlarge	8	—	256	仅 EBS 优化	最多 3,170	最多 25

db.x1e – 内存优化型实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.x1e.32xlarge	128	340	3,904	仅 EBS 优化	14,000	25

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.x1e.16xlarge	64	179	1,952	仅 EBS 优化	7,000	10
db.x1e.8xlarge	32	91	976	仅 EBS 优化	3,500	最多 10
db.x1e.4xlarge	16	47	488	仅 EBS 优化	1,750	最多 10
db.x1e.2xlarge	8	23	244	仅 EBS 优化	1000	最多 10
db.x1e.xlarge	4	12	122	仅 EBS 优化	500	最多 10

db.x1 – 内存优化型实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.x1.32xlarge	128	349	1,952	仅 EBS 优化	14,000	25
db.x1.16xlarge	64	174.5	976	仅 EBS 优化	7,000	10

db.r8g – 具有 Amazon Graviton4 处理器的内存优化型实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.r8g.48xlarge	192	—	1536	仅 EBS 优化	40000	50
db.r8g.24xlarge	96	—	768	仅 EBS 优化	30000	40
db.r8g.16xlarge	64	—	512	仅 EBS 优化	20000	30
db.r8g.12xlarge	48	—	384	仅 EBS 优化	15000	22.5
db.r8g.8xlarge	32	—	256	仅 EBS 优化	10000	15
db.r8g.4xlarge*	16	—	128	仅 EBS 优化	最高 10,000	最多 15

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.r8g.2xlarge*	8	—	64	仅 EBS 优化	最高 10,000	最多 15
db.r8g.xlarge*	4	—	32	仅 EBS 优化	最高 10,000	最多 12.5
db.r8g.large*	2	—	16	仅 EBS 优化	最高 10,000	最多 12.5

db.r7i – 由第四代 Intel Xeon 可扩展处理器提供支持的内存优化型实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.r7i.48xlarge	192	—	1536	仅 EBS 优化	40000	50
db.r7i.24xlarge	96	—	768	仅 EBS 优化	30000	37.5
db.r7i.16xlarge	64	—	512	仅 EBS 优化	20000	25
db.r7i.12xlarge	48	—	384	仅 EBS 优化	15000	18.75
db.r7i.8xlarge	32	—	256	仅 EBS 优化	10000	12.5
db.r7i.4xlarge	16	—	128	仅 EBS 优化	最高 10,000	最多 12.5
db.r7i.2xlarge	8	—	64	仅 EBS 优化	最高 10,000	最多 12.5
db.r7i.xlarge	4	—	32	仅 EBS 优化	最高 10,000	最多 12.5
db.r7i.large	2	—	16	仅 EBS 优化	最高 10,000	最多 12.5

db.r7g – 具有 Amazon Graviton3 处理器的内存优化型实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.r7g.16xlarge	64	—	512	仅 EBS 优化	20000	30

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.r7g.12xlarge	48	—	384	仅 EBS 优化	15000	22.5
db.r7g.8xlarge	32	—	256	仅 EBS 优化	10000	15
db.r7g.4xlarge	16	—	128	仅 EBS 优化	最高 10,000	最多 15
db.r7g.2xlarge*	8	—	64	仅 EBS 优化	最高 10,000	最多 15
db.r7g.xlarge*	4	—	32	仅 EBS 优化	最高 10,000	最多 12.5
db.r7g.large*	2	—	16	仅 EBS 优化	最高 10,000	最多 12.5

db.r6g – 具有 Amazon Graviton2 处理器的内存优化型实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.r6g.16xlarge	64	—	512	仅 EBS 优化	19000	25
db.r6g.12xlarge	48	—	384	仅 EBS 优化	13,500	20
db.r6g.8xlarge	32	—	256	仅 EBS 优化	9,000	12
db.r6g.4xlarge	16	—	128	仅 EBS 优化	4750	最多 10
db.r6g.2xlarge*	8	—	64	仅 EBS 优化	最多 4,750	最多 10
db.r6g.xlarge*	4	—	32	仅 EBS 优化	最多 4,750	最多 10
db.r6g.large*	2	—	16	仅 EBS 优化	最多 4,750	最多 10

db.r6gd – 具有 Amazon Graviton2 处理器和 SSD 存储的内存优化型实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.r6gd.16xlarge	64	—	512	2 x 1900 NVMe SSD	19000	25
db.r6gd.12xlarge	48	—	384	2 x 1425 NVMe SSD	13,500	20
db.r6gd.8xlarge	32	—	256	1 x 1900 NVMe SSD	9,000	12
db.r6gd.4xlarge	16	—	128	1 x 950 NVMe SSD	4750	最多 10
db.r6gd.2xlarge	8	—	64	1 x 474 NVMe SSD	最多 4,750	最多 10
db.r6gd.xlarge	4	—	32	1 x 237 NVMe SSD	最多 4,750	最多 10
db.r6gd.large	2	—	16	1 x 118 NVMe SSD	最多 4,750	最多 10

db.r6id – 具有第三代 Intel Xeon 可扩展处理器和 SSD 存储的内存优化型实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.r6id.32xlarge	128	—	1024	4x1900 NVMe SSD	40000	50
db.r6id.24xlarge	96	—	768	4x1425 NVMe SSD	30000	37.5
db.r6id.16xlarge	64	—	512	2x1900 NVMe SSD	20000	25

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.r6id.12xlarge	48	—	384	2x1425 NVMe SSD	15000	18.75
db.r6id.8xlarge	32	—	256	1x1900 NVMe SSD	10000	12.5
db.r6id.4xlarge*	16	—	128	1x950 NVMe SSD	最高 10,000	最多 12.5
db.r6id.2xlarge*	8	—	64	1x474 NVMe SSD	最高 10,000	最多 12.5
db.r6id.xlarge*	4	—	32	1x237 NVMe SSD	最高 10,000	最多 12.5
db.r6id.large*	2	—	16	1x118 NVMe SSD	最高 10,000	最多 12.5

db.r6idn – 具有第三代 Intel Xeon 可扩展处理器、SSD 存储和网络优化的内存优化型实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.r6idn.32xlarge	128	—	1024	4x1900 NVMe SSD	80,000	200
db.r6idn.24xlarge	96	—	768	4x1425 NVMe SSD	60000	150
db.r6idn.16xlarge	64	—	512	2x1900 NVMe SSD	40000	100
db.r6idn.12xlarge	48	—	384	2x1425 NVMe SSD	30000	75

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.r6idn.8xlarge	32	—	256	1x1900 NVMe SSD	20000	50
db.r6idn.4xlarge*	16	—	128	1x950 NVMe SSD	最多 20,000	最多 50
db.r6idn.2xlarge*	8	—	64	1x474 NVMe SSD	最多 20,000	最多 40
db.r6idn.xlarge*	4	—	32	1x237 NVMe SSD	最多 20,000	最多 30
db.r6idn.large*	2	—	16	1x118 NVMe SSD	最多 20,000	最多 25

db.r6in – 具有第三代 Intel Xeon 可扩展处理器和网络优化的内存优化型实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.r6in.32xlarge	128	—	1024	仅 EBS 优化	80,000	200
db.r6in.24xlarge	96	—	768	仅 EBS 优化	60000	150
db.r6in.16xlarge	64	—	512	仅 EBS 优化	40000	100
db.r6in.12xlarge	48	—	384	仅 EBS 优化	30000	75
db.r6in.8xlarge	32	—	256	仅 EBS 优化	20000	50
db.r6in.4xlarge*	16	—	128	仅 EBS 优化	最多 20,000	最多 50
db.r6in.2xlarge*	8	—	64	仅 EBS 优化	最多 20,000	最多 40
db.r6in.xlarge*	4	—	32	仅 EBS 优化	最多 20,000	最多 30

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.r6in.large*	2	—	16	仅 EBS 优化	最多 20,000	最多 25

db.r6i – 针对高内存、存储和输入/输出进行预配置的 Oracle 内存优化型实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.r6i.8xlarge.tpc2.mem4x	32	—	1024	仅 EBS 优化	40000	50
db.r6i.8xlarge.tpc2.mem3x	32	—	768	仅 EBS 优化	30000	37.5
db.r6i.6xlarge.tpc2.mem4x	24	—	768	仅 EBS 优化	30000	37.5
db.r6i.4xlarge.tpc2.mem4x	16	—	512	仅 EBS 优化	20000	25
db.r6i.4xlarge.tpc2.mem3x	16	—	384	仅 EBS 优化	15000	18.75
db.r6i.4xlarge.tpc2.mem2x	16	—	256	仅 EBS 优化	10000	12.5
db.r6i.2xlarge.tpc2.mem8x	8	—	512	仅 EBS 优化	20000	12.5
db.r6i.2xlarge.tpc2.mem4x	8	—	256	仅 EBS 优化	10000	12.5
db.r6i.2xlarge.tpc1.mem2x	8	—	128	仅 EBS 优化	最高 10,000	12.5

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.r6i.xlarge.tpc2.mem4x	4	—	128	仅 EBS 优化	最高 10,000	12.5
db.r6i.xlarge.tpc2.mem2x	4	—	64	仅 EBS 优化	最高 10,000	12.5
db.r6i.large.tpc1.mem2x	2	—	32	仅 EBS 优化	最高 10,000	12.5

db.r6i – 具有第三代 Intel Xeon 可扩展处理器的内存优化型实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.r6i.32xlarge	128	—	1024	仅 EBS 优化	40000	50
db.r6i.24xlarge	96	—	768	仅 EBS 优化	30000	37.5
db.r6i.16xlarge	64	—	512	仅 EBS 优化	20000	25
db.r6i.12xlarge	48	—	384	仅 EBS 优化	15000	18.75
db.r6i.8xlarge	32	—	256	仅 EBS 优化	10000	12.5
db.r6i.4xlarge*	16	—	128	仅 EBS 优化	最高 10,000	最多 12.5
db.r6i.2xlarge*	8	—	64	仅 EBS 优化	最高 10,000	最多 12.5
db.r6i.xlarge*	4	—	32	仅 EBS 优化	最高 10,000	最多 12.5
db.r6i.large*	2	—	16	仅 EBS 优化	最高 10,000	最多 12.5

db.r5d – 具有 Intel Xeon Platinum 处理器和 SSD 存储的内存优化型实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.r5d.24xlarge	96	347	768	4 x 900 NVMe SSD	19000	25
db.r5d.16xlarge	64	264	512	4 x 600 NVMe SSD	13600	20
db.r5d.12xlarge	48	173	384	2 x 900 NVMe SSD	9500	10
db.r5d.8xlarge	32	132	256	2 x 600 NVMe SSD	6800	10
db.r5d.4xlarge	16	71	128	2 x 300 NVMe SSD	4750	最多 10
db.r5d.2xlarge*	8	38	64	1 x 300 NVMe SSD	最多 4,750	最多 10
db.r5d.xlarge*	4	19	32	1 x 150 NVMe SSD	最多 4,750	最多 10
db.r5d.large*	2	10	16	1 x 75 NVMe SSD	最多 4,750	最多 10

db.r5b – 针对高内存、存储和输入/输出进行预配置的 Oracle 内存优化型实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.r5b.8xlarge.tpc2.mem3x	32	—	768	仅 EBS 优化	60000	25
db.r5b.6xlarge.tpc2.mem4x	24	—	768	仅 EBS 优化	60000	25

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.r5b.4xlarge.tpc2.mem4x	16	—	512	仅 EBS 优化	40000	20
db.r5b.4xlarge.tpc2.mem3x	16	—	384	仅 EBS 优化	30000	10
db.r5b.4xlarge.tpc2.mem2x	16	—	256	仅 EBS 优化	20000	10
db.r5b.2xlarge.tpc2.mem8x	8	—	512	仅 EBS 优化	40000	20
db.r5b.2xlarge.tpc2.mem4x	8	—	256	仅 EBS 优化	20000	10
db.r5b.2xlarge.tpc1.mem2x	8	—	128	仅 EBS 优化	10000	最多 10
db.r5b.xlarge.tpc2.mem4x	4	—	128	仅 EBS 优化	10000	最多 10
db.r5b.xlarge.tpc2.mem2x	4	—	64	仅 EBS 优化	最高 10,000	最多 10
db.r5b.large.tpc1.mem2x	2	—	32	仅 EBS 优化	最高 10,000	最多 10

db.r5b – 具有 Intel Xeon Platinum 处理器和 EBS 优化的内存优化型实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.r5b.24xlarge	96	347	768	仅 EBS 优化	60000	25
db.r5b.16xlarge	64	264	512	仅 EBS 优化	40000	20

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.r5b.12xlarge	48	173	384	仅 EBS 优化	30000	10
db.r5b.8xlarge	32	132	256	仅 EBS 优化	20000	10
db.r5b.4xlarge	16	71	128	仅 EBS 优化	10000	最多 10
db.r5b.2xlarge*	8	38	64	仅 EBS 优化	最高 10,000	最多 10
db.r5b.xlarge*	4	19	32	仅 EBS 优化	最高 10,000	最多 10
db.r5b.large*	2	10	16	仅 EBS 优化	最高 10,000	最多 10

db.r5 – 针对高内存、存储和输入/输出进行预配置的 Oracle 内存优化型实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.r5.12xlarge.tpc2.mem2x	48	—	768	仅 EBS 优化	19000	25
db.r5.8xlarge.tpc2.mem3x	32	—	768	仅 EBS 优化	19000	25
db.r5.6xlarge.tpc2.mem4x	24	—	768	仅 EBS 优化	19000	25
db.r5.4xlarge.tpc2.mem4x	16	—	512	仅 EBS 优化	13600	20
db.r5.4xlarge.tpc2.mem3x	16	—	384	仅 EBS 优化	9500	10
db.r5.4xlarge.tpc2.mem2x	16	—	256	仅 EBS 优化	6800	10

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.r5.2xlarge.tpc2.mem8x	8	—	512	仅 EBS 优化	13600	20
db.r5.2xlarge.tpc2.mem4x	8	—	256	仅 EBS 优化	6800	10
db.r5.2xlarge.tpc1.mem2x	8	—	128	仅 EBS 优化	4750	最多 10
db.r5.xlarge.tpc2.mem4x	4	—	128	仅 EBS 优化	4750	最多 10
db.r5.xlarge.tpc2.mem2x	4	—	64	仅 EBS 优化	最多 4,750	最多 10
db.r5.large.tpc1.mem2x	2	—	32	仅 EBS 优化	最多 4,750	最多 10

db.r5 – 内存优化型实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.r5.24xlarge	96	347	768	仅 EBS 优化	19000	25
db.r5.16xlarge	64	264	512	仅 EBS 优化	13600	20
db.r5.12xlarge	48	173	384	仅 EBS 优化	9500	12
db.r5.8xlarge	32	132	256	仅 EBS 优化	6800	10
db.r5.4xlarge	16	71	128	仅 EBS 优化	4750	最多 10
db.r5.2xlarge*	8	38	64	仅 EBS 优化	最多 4,750	最多 10
db.r5.xlarge*	4	19	32	仅 EBS 优化	最多 4,750	最多 10

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.r5.large*	2	10	16	仅 EBS 优化	最多 4,750	最多 10

db.r4 – 具有 Intel Xeon 可扩展处理器的内存优化型实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.r4.16xlarge	64	195	488	仅 EBS 优化	14,000	25
db.r4.8xlarge	32	99	244	仅 EBS 优化	7,000	10
db.r4.4xlarge	16	53	122	仅 EBS 优化	3,500	最多 10
db.r4.2xlarge	8	27	61	仅 EBS 优化	1,700	最多 10
db.r4.xlarge	4	13.5	30.5	仅 EBS 优化	850	最多 10
db.r4.large	2	7	15.25	仅 EBS 优化	425	最多 10

db.r3 – 内存优化型实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.r3.8xlarge**	32	104	244	仅限于 EBS	—	10
db.r3.4xlarge	16	52	122	仅 EBS 优化	2000	高
db.r3.2xlarge	8	26	61	仅 EBS 优化	1000	高
db.r3.xlarge	4	13	30.5	仅 EBS 优化	500	中
db.r3.large	2	6.5	15.25	仅 EBS 优化	—	中

* 这些数据库实例类可以至少每 24 小时支持一次 30 分钟的最高性能。有关基础 EC2 实例类型的基准性能的更多信息，请参阅《Amazon EC2 用户指南》中的 Amazon EBS 优化的实例。

** r3.8xlarge 数据库实例类没有专用的 EBS 带宽，因此不提供 EBS 优化。对于此实例类，网络流量和 Amazon EBS 流量将共用同一 10Gb 网络接口。

计算优化型实例类的硬件规格

下表显示了计算优化型实例类的计算、内存、存储和带宽规格。

db.c6gd – 计算优化型实例类（仅适用于多可用区数据库集群部署）

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.c6gd.16xlarge	64	—	128	2 x 1,900 NVMe SSD	19000	25
db.c6gd.12xlarge	48	—	96	2 x 1,425 NVMe SSD	13,500	20
db.c6gd.8xlarge	32	—	64	1 x 1,900 NVMe SSD	9,000	12
db.c6gd.4xlarge	16	—	32	1 x 950 NVMe SSD	4750	最多 10
db.c6gd.2xlarge	8	—	16	1 x 474 NVMe SSD	最多 4,750	最多 10
db.c6gd.xlarge	4	—	8	1 x 237 NVMe SSD	最多 4,750	最多 10
db.c6gd.large	2	—	4	1 x 118 NVMe SSD	最多 4,750	最多 10
db.c6gd.medium	1	—	2	1 x 59 NVMe SSD	最多 4,750	最多 10

可突增性能实例类的硬件规格

下表显示了可突增性能实例类的计算、内存、存储和带宽规格。

db.t4g - 由 Amazon Graviton2 处理器提供支持的可突增性能实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.t4g.2xlarge*	8	—	32	仅 EBS 优化	最多 2,780	最多 5
db.t4g.xlarge*	4	—	16	仅 EBS 优化	最多 2,780	最多 5
db.t4g.large*	2	—	8	仅 EBS 优化	最多 2,780	最多 5
db.t4g.medium*	2	—	4	仅 EBS 优化	最多 2,085	最多 5
db.t4g.small*	2	—	2	仅 EBS 优化	最多 2,085	最多 5
db.t4g.micro*	2	—	1	仅 EBS 优化	最多 2,085	最多 5

db.t3 – 具爆发能力的实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.t3.2xlarge*	8	变量	32	仅 EBS 优化	最多 2,048	最多 5
db.t3.xlarge*	4	变量	16	仅 EBS 优化	最多 2,048	最多 5
db.t3.large*	2	变量	8	仅 EBS 优化	最多 2,048	最多 5
db.t3.medium*	2	变量	4	仅 EBS 优化	最多 1,536	最多 5
db.t3.small*	2	变量	2	仅 EBS 优化	最多 1,536	最多 5
db.t3.micro*	2	变量	1	仅 EBS 优化	最多 1,536	最多 5

db.t2 – 具爆发能力的实例类

实例类	vCPU	ECU	内存（GiB）	实例存储（GiB）	最大 EBS 带宽（Mbps）	网络带宽（Gbps)
db.t2.2xlarge	8	变量	32	仅限于 EBS	—	中
db.t2.xlarge	4	变量	16	仅限于 EBS	—	中
db.t2.large	2	变量	8	仅限于 EBS	—	中
db.t2.medium	2	变量	4	仅限于 EBS	—	中
db.t2.small	1	变量	2	仅限于 EBS	—	低
db.t2.micro	1	变量	1	仅限于 EBS	—	低

* 这些数据库实例类可以至少每 24 小时支持一次 30 分钟的最高性能。有关基础 EC2 实例类型的基准性能的更多信息，请参阅《Amazon EC2 用户指南》中的 Amazon EBS 优化的实例。

Amazon RDS 数据库实例存储

Amazon RDS for Db2、MariaDB、MySQL、PostgreSQL、Oracle 和 Microsoft SQL Server 的数据库实例使用 Amazon Elastic Block Store（Amazon EBS）卷进行数据库和日志存储。

在某些情况下，您的数据库工作负载可能无法 100% 实现您预调配的 IOPS。有关更多信息，请参阅影响数据库性能的因素。

主题

Amazon RDS 存储类型

Amazon RDS 提供三种存储类型：预调配 IOPS SSD（也称为 io1 和 io2 Block Express）、通用型 SSD（也称为 gp2 和 gp3）和磁性存储（也称为标准存储）。它们的性能特性和价格不同，这意味着您可以根据数据库工作负载需求定制存储性能和成本。您可以创建具有多达 64 TiB 存储的 Db2、MySQL、MariaDB、Oracle、SQL Server 和 PostgreSQL RDS 数据库实例。RDS for Db2 不支持 gp2 和磁性存储类型。

下面的列表简要介绍这三个存储类型：

- 预调配 IOPS SSD – 预调配 IOPS 存储符合输入/输出密集型工作负载（尤其是数据库工作负载）的需求，此类工作负载需要低输入/输出延迟和一致的输入/输出吞吐量。预调配 IOPS 存储最适合生产环境。

  有关预调配 IOPS 存储的更多信息（包括存储大小范围），请参阅预置 IOPS SSD 存储。
- 通用型 SSD – 通用型 SSD 卷提供了经济高效的存储，适用于在中等规模的数据库实例上运行的各种工作负载。通用型存储最适合开发和测试环境。

  有关通用型 SSD 存储的更多信息（包括存储大小范围），请参阅通用型 SSD 存储。

- 磁性 – Amazon RDS 还支持磁性存储以实现向后兼容。建议您采用通用型 SSD 或预调配 IOPS SSD 来满足所有新存储需求。磁性存储上的数据库实例允许的最大存储量为 3 TiB。有关更多信息，请参阅 磁性存储（已弃用）。

预置 IOPS SSD 存储

对于需要快速且一致的 I/O 性能的生产应用程序，我们建议使用预调配 IOPS 存储。预调配 IOPS 存储是一种存储类型，它提供了可预测的性能以及一致的低延迟。预调配 IOPS 存储针对要求性能一致的联机事务处理（OLTP）工作负载进行了优化。预调配 IOPS 可帮助这些工作负载的性能优化。

Amazon RDS 提供了两种类型的预调配 IOPS SSD 存储：io2 和 io1。在创建数据库实例时，您可以指定 IOPS 速率和卷的大小。Amazon RDS 为数据库实例提供该 IOPS 速率，直到您对其进行更改。

主题

io2 Block Express 存储（推荐）

对于 I/O 密集型和延迟敏感型工作负载，我们建议您使用预调配 IOPS SSD io2 Block Express 存储，以实现高达每秒 256000 次 I/O 操作（IOPS）。io2 Block Express 卷的吞吐量因每个卷的预调配 IOPS 量和正在运行的 I/O 操作的规模而异。

所有基于 Amazon Nitro 系统的 RDS io2 卷都是 io2 Block Express 卷，平均延迟为亚毫秒。不基于 Amazon Nitro 系统的数据库实例是 io2 卷。

下表显示了每个数据库引擎的预调配 IOPS 范围和最大吞吐量以及存储大小范围。

数据库引擎	存储大小的范围	预调配 IOPS 的范围	最大吞吐量
Db2、Maria DB、MySQL 和 PostgreSQL	100–65536GiB	1000–256000 IOPS	16000 MiB/s

数据库引擎	存储大小的范围	预调配 IOPS 的范围	最大吞吐量
Oracle	100–199GiB	1000–199000 IOPS	4,000 MiB/s
Oracle	200–65536GiB	1000–256000 IOPS	16000 MiB/s
SQL Server	20 – 65536 GiB	1000–256000 IOPS	4,000 MiB/s

IOPS 和存储大小范围具有以下约束：

- IOPS 与已分配存储（以 GiB 为单位）的比率必须介于 0.5 到 1,000 之间。对于不基于 Amazon Nitro 系统的数据库实例，该比率必须介于 0.5 到 500 之间。
- 可以为 256GiB 和更大的卷预置最多 IOPS（1000IOPS x 256GiB = 256000IOPS）。对于不基于 Amazon Nitro 系统的数据库实例，最大 IOPS 可达到 512GiB（500IOPS x 512GiB = 256000 IOPS）。
- 吞吐量按比例扩展至每个预置 IOPS 0.256 Mib/s。在 256000 IOPS 下，当 I/O 大小为 16KiB 时，最大吞吐量可以达到 4000MiB/s；在 I/O 大小为 256KiB 时，可以实现 16000 IOPS 或更高的吞吐量。对于不基于 Amazon Nitro 系统的数据库实例，在 128000 IOPS 下，当 I/O 大小为 16KiB 时，最大吞吐量可达到 2000MiB/s。
- 如果您使用存储自动扩展，IOPS 和最大存储阈值（以 GiB 为单位）之间的比率也同样适用。有关存储自动扩展的更多信息，请参阅 使用 Amazon RDS 存储自动扩展功能自动管理容量。

Amazon RDS io2 Block Express 卷可用于以下 Amazon Web Services 区域中：

- 亚太地区（香港）
- 亚太地区（孟买）
- 亚太地区（首尔）
- 亚太地区（新加坡）
- 亚太地区（悉尼）
- 亚太地区（东京）
- 加拿大（中部）
- 欧洲地区（法兰克福）
- 欧洲地区（爱尔兰）
- Europe (London)

- 欧洲（斯德哥尔摩）
- 中东（巴林）
- 美国东部（俄亥俄州）
- 美国东部（弗吉尼亚州北部）
- 美国西部（加利福尼亚北部）
- 美国西部（俄勒冈州）

io1 存储（上一代）

对于 I/O 密集型工作负载，您可以使用预调配 IOPS SSD io1 存储，并实现每秒高达 256,000 次的 I/O 操作（IOPS）。io1 卷的吞吐量因每个卷的预调配 IOPS 量和正在执行的 I/O 操作的规模而异。建议在可用的情况下使用 io2 Block Express 存储。

下表显示了每个数据库引擎的预调配 IOPS 范围和最大吞吐量以及存储大小范围。

数据库引擎	存储大小的范围	预调配 IOPS 的范围	最大吞吐量
Db2、Maria DB、MySQL 和 PostgreSQL	100–399GiB	1000 – 19950 IOPS	500 MiB/s
Db2、Maria DB、MySQL 和 PostgreSQL	400–65536GiB	1000–256000 IOPS	4,000 MiB/s
Oracle	100–199GiB	1000 – 9950 IOPS	500 MiB/s
Oracle	200–65536GiB	1000–256000 IOPS[1]	4,000 MiB/s
SQL Server	20–16384GiB	1000–64000 IOPS[2]	1,000 MiB/s

Note

[1] 对于 Oracle，您只能在 r5b 实例类型上预调配最高 256000 IOPS 的性能。

[2] 对于 SQL Server，仅可在基于 Nitro 的实例（m5*、m6i、r5*、r6i 和 z1d 实例类型）上确保最高 64000 IOPS 的性能。其他实例类型可确保最高 32000 IOPS 的性能。

IOPS 和存储大小范围具有以下约束：

- 在 RDS for SQL Server 上，IOPS 与已分配存储（以 GiB 为单位）的比率必须为 1–50，而在其他 RDS 数据库引擎上必须为 0.5–50。
- 如果您使用存储自动扩展，IOPS 和最大存储阈值（以 GiB 为单位）之间的比率也同样适用。

  有关存储自动扩展的更多信息，请参阅 使用 Amazon RDS 存储自动扩展功能自动管理容量。

将预置 IOPS 存储与多可用区部署或只读副本组合

对于生产 OLTP 使用案例，建议您使用多可用区部署提高容错能力和预调配 IOPS 存储，以便实现快速可预测性能。

您还可以将预调配 IOPS 存储与 MySQL、MariaDB 或 PostgreSQL 的只读副本一起使用。只读副本的存储类型与主数据库实例上的存储类型无关。例如，您可能需要对具有使用预调配 IOPS SSD 存储的主数据库实例的只读副本使用通用型 SSD 来降低成本。不过，在这种情况下，与主数据库实例和只读副本均使用预调配 IOPS 存储的配置相比，只读副本的性能可能会不同。

预置 IOPS 存储成本

通过预调配 IOPS 存储，您需要为预配置的资源付费，因此在规定月份中无论是否使用这些资源，您都需为此支付费用。

利用 Amazon RDS 预调配 IOPS 存储获得最佳性能

如果您的工作负载受 I/O 限制，则使用预调配 IOPS 存储可以增加系统能够并发处理的 I/O 请求数。并发度提高可降低延迟，因为 I/O 请求在队列中占用的时间变少。降低延迟可加快数据库提交，从而缩短响应时间并提高数据库吞吐量。

预调配 IOPS 存储提供了一种通过指定 IOPS 来预留 I/O 容量的方法。不过，与任何其他的系统容量属性一样，负载下的最大吞吐量会受到系统首先消耗的资源的限制。该资源可能是网络带宽、CPU、内存或数据库内部资源。

通用型 SSD 存储

通用存储提供了经济高效的存储性能，可用于大多数对延迟或性能不敏感的数据库工作负载。

> **Note**
>
> 与使用预调配 IOPS 存储的实例相比，使用通用型存储的数据库实例会经历更久的延迟。如果您需要数据库实例在执行这些操作之后保持最低延迟，建议您使用预置 IOPS SSD 存储。

Amazon RDS 提供两种类型的通用型存储：gp3 存储（推荐）和 gp2 存储（上一代）。

gp3 存储（推荐）

通过使用通用型 gp3 存储卷，您可以独立于存储容量自定义存储性能。存储性能是每秒输入/输出操作数（IOPS）与存储卷可执行读取和写入的速度（存储吞吐量）的组合。在 gp3 存储卷上，Amazon RDS 提供的基准存储性能为 3000 IOPS 和 125MiB/s。

对于除 RDS For SQL Server 之外的每个 RDS 数据库引擎，当 gp3 卷的存储大小达到某个阈值时，基准存储性能将得到提高。这是因为卷条带化，其中存储使用四个卷而不是一个卷。RDS for SQL Server 不支持卷条带化，因此没有阈值。对于带区卷，Amazon RDS 提供的基准存储性能为 12,000 IOPS 和 500 MiB/s。

下表显示了 Amazon RDS 数据库引擎上 gp3 卷的存储性能，包括阈值。

数据库引擎	存储大小	基准存储性能	预调配 IOPS 的范围	预调配存储吞吐量的范围
Db2、MariaDB、MySQL 和 PostgreSQL	20–399 GiB	3000 IOPS/125MiB/s	不适用	不适用
Db2、MariaDB、MySQL 和 PostgreSQL	400–65536GiB	12000 IOPS/500MiB/s	12,000–64,000IOPS	500 – 4000MiB/s
Oracle	20–199GiB	3000 IOPS/125MiB/s	不适用	不适用
Oracle	200–65536GiB	12000 IOPS/500MiB/s	12,000–64,000IOPS	500 – 4000MiB/s

数据库引擎	存储大小	基准存储性能	预调配 IOPS 的范围	预调配存储吞吐量的范围
SQL Server	20–16384GiB	3000 IOPS/125MiB/s	3,000–16,000IOPS	125 – 1000MiB/s

对于除 RDS for SQL Server 之外的每个数据库引擎，当存储大小等于或高于阈值时，您可以预调配额外的 IOPS 和存储吞吐量。对于 RDS for SQL Server，您可以为任何可用存储大小预调配额外的 IOPS 和存储吞吐量。对于所有数据库引擎，您只需为额外的预调配存储性能付费。有关更多信息，请参阅 Amazon RDS 定价。

尽管增加的预调配 IOPS 和存储吞吐量不依赖于存储大小，但它们是相互关联的。当您将 MariaDB 和 MySQL 的 IOPS 提高到 32,000 以上时，存储吞吐量值会自动从 500MiBps 增加。例如，当您在 RDS for MySQL 上将 IOPS 设置为 40,000 时，存储吞吐量必须至少为 625MiBps。Db2、Oracle、PostgreSQL 和 SQL Server 数据库实例不发生自动增加的情况。

对于多可用区数据库集群，Amazon RDS 会根据您预调配的 IOPS 自动设置吞吐量值。您不能修改吞吐量值。

RDS 上 gp3 卷的存储性能值具有以下限制：

- 对于所有支持的数据库引擎，存储吞吐量与 IOPS 的最大比率为 0.25。
- 在 RDS for SQL Server 上，IOPS 与已分配存储（以 GiB 为单位）的最小比率为 0.5。其他支持的数据库引擎没有最低比率。
- 对于所有支持的数据库引擎，IOPS 与已分配存储的最大比率为 500。
- 如果您使用存储自动扩展，IOPS 和最大存储阈值（以 GiB 为单位）之间的比率也同样适用。

  有关存储自动扩展的更多信息，请参阅 使用 Amazon RDS 存储自动扩展功能自动管理容量。

gp2 存储（上一代）

当您的应用程序不需要高存储性能时，您可以使用通用型 SSD gp2 存储。gp2 存储的基准输入/输出性能为 3IOPS/GiB，最少具有 100IOPS。这种关系意味着较大的卷具有更好的性能。例如，一个 100 GiB 卷的基准性能为 300 IOPS。一个 1000GiB 卷的基准性能为 3000 IOPS。

大小低于 1000GiB 的单独 gp2 卷也有能力在延长的时间内突增至 3000 IOPS。卷 I/O 积分余额决定突增性能。有关基准性能和输入/输出积分余额如何影响性能的更详细说明，请参阅 Amazon 数据库博客

上的博文 Understanding burst vs. baseline performance with Amazon RDS and gp2（了解 Amazon RDS 和 gp2 的突增与基线性能）。

许多工作负载从不会耗尽突增余额。不过，某些工作负载可能会用完 3000 IOPS 突增存储积分余额，因此，请规划存储容量以满足您的工作负载需求。

对于大于 4,000 GiB 的 gp2 卷，基准性能大于突增性能。对于这些卷，因为基准性能优于 3000 IOPS 突发性能，所以突发性能并不重要。但是，对于某些引擎和大小的数据库实例，存储跨四个卷进行条带化，可提供单个卷的四倍基准吞吐量和四倍突增 IOPS。

下表显示了 Amazon RDS 数据库引擎上各种存储大小的 gp2 卷的存储性能。

数据库引擎	RDS 存储大小	基准 IOPS 的范围	基准吞吐量的范围	突增 IOPS
MariaDB、MySQL 和 PostgreSQL	5–399GiB[1]	100 - 1197 IOPS	128 - 250MiB/s	3000
MariaDB、MySQL 和 PostgreSQL	400–1335GiB	1200 - 4005 IOPS	512 - 1000 MiB/s	12000
MariaDB、MySQL 和 PostgreSQL	1336–3999GiB	4008 - 11997 IOPS	1,000 MiB/s	12000
MariaDB、MySQL 和 PostgreSQL	4000–65536GiB	12000 - 64000 IOPS	1,000 MiB/s	不适用[2]
Oracle	20–199GiB	100 - 597 IOPS	128 - 250MiB/s	3000
Oracle	200–1335GiB	600 - 4005 IOPS	512 - 1000 MiB/s	12000
Oracle	1336–3999GiB	4008 - 11997 IOPS	1,000 MiB/s	12000
Oracle	4000–65536GiB	12000 - 64000 IOPS	1,000 MiB/s	不适用[2]

数据库引擎	RDS 存储大小	基准 IOPS 的范围	基准吞吐量的范围	突增 IOPS
SQL Server	20–333GiB	100 - 999IOPS	128 - 250MiB/s	3000
SQL Server	334–999GiB	1002 - 2997 IOPS	250 MiB/s	3000
SQL Server	1000–16384GiB	3000 - 16000 IOPS	250 MiB/s	不适用[2]

> **Note**
>
> [1] 使用 Amazon Web Services Management Console，您可以在 db.t3.micro 和 db.t4g.micro 数据库实例类的免费套餐中，创建存储大小最小为 5GiB 的数据库实例。否则，最小存储大小为 20GiB。此限制不适用于 Amazon CLI 和 RDS API。
>
> [2] 卷的基准性能超过了最大突发性能。

固态驱动器（SSD）存储类型的性能特征

下表介绍了 Amazon RDS 使用的 SSD 存储卷的用例和每个卷的性能特征。

特征	预调配 IOPS（io2 Block Express）	预调配 IOPS（io1）	通用型（gp3）	通用型（gp2）
描述	RDS 存储组合中性能最高（IOPS、吞吐量、延迟） 专为延迟敏感型、事务工作负载而设计	一致的存储性能（IOPS、吞吐量、延迟） 专为延迟敏感型、事务工作负载而设计	可灵活地独立预调配存储、IOPS 和吞吐量 平衡各种事务性工作负载的性价比	提供可突增的 IOPS 平衡各种事务性工作负载的性价比
使用案例	需要亚毫秒延迟和高达 256000 IOPS	需要最多 256000IOPS 的	在开发/测试环境中，在中型关系	在开发/测试环境中，在中型关系

特征	预调配 IOPS（io2 Block Express）	预调配 IOPS（io1）	通用型（gp3）	通用型（gp2）
	的持续 IOPS 性能的业务关键型事务性工作负载	持续 IOPS 性能的事务性工作负载	数据库上运行的广泛工作负载	数据库上运行的广泛工作负载
延迟	亚毫秒，在 99.9% 的时间内始终提供	一位数毫秒，在 99.9% 的时间内始终提供	一位数毫秒，在 99% 的时间内始终提供	一位数毫秒，在 99% 的时间内始终提供
卷大小	100–65536GiB	100–65536 GiB（在 RDS for SQL Server 上为 20–16384GiB）	20–65536GiB（在 RDS for SQL Server 上为 16384GiB）	20–65536GiB（在 RDS for SQL Server 上为 16384GiB）
最大 IOPS	256,000	256,000（在 RDS for SQL Server 上为 64,000）	64,000（在 RDS for SQL Server 上为 16,000）	64,000（在 RDS for SQL Server 上为 16,000） **Note** 您无法直接在 gp2 存储上预调配 IOPS。IOPS 因分配的存储大小而异。

特征	预调配 IOPS（io2 Block Express）	预调配 IOPS（io1）	通用型（gp3）	通用型（gp2）
最大吞吐量	根据预调配 IOPS 卷扩展至最多 4,000MB/s 吞吐量按比例扩展至每个预置 IOPS 0.256 Mib/s。在 256000 IOPS 下，当 I/O 大小为 16KiB 时，最大吞吐量可以达到 4000MiB/s；在 I/O 大小为 256KiB 时，可以实现 16000 IOPS 或更高的吞吐量。 对于不基于 Amazon Nitro 系统的实例，在 128000 IOPS 下，当 I/O 大小为 16KiB 时，最大吞吐量可达到 2000MiB/s。	根据预调配 IOPS 卷扩展至最多 4,000MB/s	预调配多达 4000 MB/s 的额外吞吐量（对于 RDS for SQL Server，为 1000 MB/s）	1000MB/s（RDS for SQL Server 上为 250MB/s）
Amazon CLI 及 RDS API 名称	io2	io1	gp3	gp2

在 SSD 卷上自动进行条带化

当您选择通用型 SSD 或预调配 IOPS SSD 时，根据所选引擎和所请求的存储空间量，Amazon RDS 会自动跨多个卷进行条带化以提高性能，如下表所示。

数据库引擎	Amazon RDS 存储大小	预调配的卷数
Db2	小于 400GiB	1
Db2	400–65536GiB	4
MariaDB、MySQL 和 PostgreSQL	小于 400GiB	1
MariaDB、MySQL 和 PostgreSQL	400–65536GiB	4
Oracle	小于 200GiB	1
Oracle	200–65536GiB	4
SQL Server	任何	1

修改 SSD 卷时对性能的影响

当您修改通用型 SSD 或预调配 IOPS SSD 卷时，它会经历一系列状态。当卷处于 `optimizing` 状态时，卷性能介于源配置规范和目标配置规范之间。过渡卷的性能将不低于这两种规格中的最低者。

当您修改实例的存储以使其从一个卷变为四个卷时，或者当您使用磁性存储修改实例时，Amazon RDS 不使用"弹性卷"功能。相反，Amazon RDS 会预调配新卷，并将数据从旧卷透明地移到新卷。此操作会同时消耗新旧卷的大量 IOPS 和吞吐量。根据卷的大小和修改期间存在的数据库工作负载量，此操作可能会消耗大量 IOPS，显著增加 I/O 延迟，需要几个小时才能完成，而 RDS 实例保持为 `Modifying` 状态。

EBS 优化型实例的基准和最大 IOPS 速率

EBS 优化的实例具有基准和最大 IOPS 速率。最大 IOPS 速率是在数据库实例级别强制执行的。一组 EBS 卷加起来的 IOPS 速率高于最大值，但不能超过实例级阈值。例如，如果特定数据库实例类的最大 IOPS 为 40000，而您连接了四个 64000 IOPS EBS 卷，则最大 IOPS 为 40000 而不是 256000。有关特定于每种 EC2 实例类型的 IOPS 最大值，请参阅《Amazon EC2 用户指南（适用于 Linux 实例）》中的[支持的实例类型]。

磁性存储（已弃用）

Amazon RDS 不再支持磁性存储。建议您采用通用型 SSD 或预调配 IOPS SSD 来满足存储需求。

专用日志卷（DLV）

您可以利用 Amazon RDS 控制台、Amazon CLI 或 Amazon RDS API，将专用日志卷（DLV）用于使用预调配 IOPS（PIOPS）存储的数据库实例。DLV 将 PostgreSQL 数据库事务日志、MySQL/MariaDB 重做日志和二进制日志移动到与包含数据库表的卷不同的存储卷中。DLV 可以提高事务写入日志的效率和一致性。DLV 非常适合分配了较大存储空间、每秒 I/O（IOPS）要求较高或工作负载对延迟敏感的数据库。

DLV 支持 PIOPS 存储（io1 和 io2 Block Express），创建它们时的固定大小为 1,024 GiB，预调配 IOPS 为 3,000。

> **Note**
>
> DLV 不支持通用存储（gp2 和 gp3）。

对于以下版本，Amazon RDS 在所有 Amazon Web Services 区域都支持 DLV：

- MariaDB 10.6.7 及更高的 10 版本
- MySQL 8.0.28 及更高的 8.0 版本，MySQL 8.4.3 及更高的 8.4 版本
- PostgreSQL 13.10 及更高的 13 版本、14.7 及更高的 14 版本、15.2 及更高的 15 版本，以及 16.1 及更高的 16 版本

RDS 对多可用区部署支持 DLV。修改或创建多可用区实例时，会同时为主实例和辅助实例创建 DLV。

RDS 对于只读副本支持 DLV。如果主数据库实例启用了 DLV，则启用 DLV 后创建的所有只读副本也将具有 DLV。除非经过明确修改，否则在切换到 DLV 之前创建的任何只读副本都不会启用 DLV。我们建议在启用 DLV 之前连接到主实例的所有只读副本也要手动修改为具有 DLV。

在修改数据库实例的 DLV 设置后，数据库实例必须重新启动。

有关启用 DLV 的信息，请参阅使用专用日志卷（DLV）。

监控数据库性能

Amazon RDS 提供了可供您用来确定如何执行数据库实例的多种指标。您可以在 Amazon RDS 管理控制台中的实例的摘要页上查看这些指标。您还可使用 Amazon CloudWatch 监控这些指标。有关更多信息，请参阅“在 Amazon RDS 控制台中查看指标”。增强监测提供了更详细的 I/O 指标；有关更多信息，请参阅使用增强监控来监控操作系统指标。

以下指标对于监控数据库实例的性能很有帮助：

- `DiskQueueDepth`：队列中等待处理的 I/O 请求数。这些是由应用程序提交但由于设备忙于处理其他 I/O 请求而尚未发送到设备的 I/O 请求。在队列中等待所花的时间是延迟和处理时间的一部分（不以指标形式提供）。该指标作为给定时间间隔内队列深度平均值进行报告。Amazon RDS 每隔 1 分钟报告队列深度。队列深度典型值在零至数百之间。
- `EBSByteBalance%`：RDS 数据库的突增存储桶中剩余的吞吐量积分的百分比。此指标仅对基本监控可用。该指标值基于包括根卷在内的所有卷的吞吐量，而不是仅基于那些包含数据库文件的卷。

  当该指标接近零时，意味着数据库实例即将耗尽计算容量。如果这种情况经常发生，请考虑升级到更大的实例类大小，例如从 db.r6g.large 升级到 db.r6g.xlarge。有关更多信息，请参阅 数据库实例类。
- `ReadIOPS` 和 `WriteIOPS`：每秒完成的 I/O 操作数。该指标作为给定时间间隔内 IOPS 平均值进行报告。Amazon RDS 每隔 1 分钟分别报告读取和写入 IOPS。`TotalIOPS` 是读取和写入 IOPS 的总和。IOPS 的典型值在零到每秒数万次之间。

  如果 `TotalIOPS` 值经常接近您为数据库实例设置的预调配 IOPS 值，请考虑增加预调配 IOPS（io1、io2 Block Express 和 gp3 存储类型）。

  测量的 IOPS 值与单个 I/O 操作的大小无关。这意味着，在您度量输入/输出性能时，确保查看实例的吞吐量，而不是输入/输出操作数量。
- `ReadLatency` 和 `WriteLatency`：从提交 I/O 请求到完成请求之间的已用时间。该指标作为给定时间间隔内延迟平均值进行报告。Amazon RDS 每分钟分别报告一次读取和写入延迟。典型的延迟值以毫秒 (ms) 为单位。
- `ReadThroughput` 和 `WriteThroughput`：每秒传输到磁盘或从磁盘中传出的字节数。该指标作为给定时间间隔内吞吐量平均值进行报告。Amazon RDS 每隔 1 分钟分别报告读取和写入吞吐量，所用单位为每秒字节数（B/s）。吞吐量的典型值在零到 I/O 通道的最大带宽之间。

  如果吞吐量值经常接近数据库实例的最大吞吐量，则当您使用 gp3 存储类型时，可以考虑预调配更多的存储吞吐量。

影响数据库性能的因素

系统活动、数据库工作负载和数据库实例类可能影响数据库性能。

主题

- 系统活动
- 数据库工作负载
- 数据库实例类

系统活动

以下系统相关活动均使用 I/O 容量，并且可能会在执行这些活动时降低数据库实例性能：

- 多可用区备用创建
- 创建只读副本
- 创建存储类型

数据库工作负载

在某些情况下，数据库或应用程序设计导致并发问题、锁定或其他形式的数据库争用。在这类情况下，您可能无法直接使用所有预配置的带宽。此外，您可能遇到以下与工作负载相关的情况：

- 已到达基础实例类型的吞吐量限制。
- 因为应用程序没有驱动足够的输入/输出操作，所以队列深度始终小于 1。
- 您在数据库中遇到查询争用，即使部分 I/O 容量未使用也是如此。

在某些情况下，系统资源均未达到或接近限值且添加线程不会增加数据库事务速率。在这种情况下，瓶颈很可能是数据库中的争用。最常见的形式为行锁和索引页锁争用，但也有很多其他的可能性。如果您遇到的是这种情况，请征询数据库性能优化专家的意见。

数据库实例类

要让 Amazon RDS 数据库实例发挥最大性能，请选择具有足够带宽的最新一代实例类型来支持您的存储类型。例如，您可以选择 Amazon EBS 优化实例和具有 10GB 网络连接的实例。

Important

您可能会看到 IOPS 性能低于您可以使用 RDS 预置的最大值，具体取决于所使用的实例类。有关数据库实例类的 IOPS 性能的特定信息，请参阅《Amazon EC2 用户指南》中的 Amazon EBS 优化实例。建议您在为数据库实例设置预置 IOPS 值之前，先确定实例类的最大 IOPS。

建议您使用最新一代的实例以获得最佳性能。上一代的数据库实例还具有较低的存储上限。

一些较旧的 32 位文件系统的存储容量可能较低。要确定数据库实例的存储容量，可以使用 describe-valid-db-instance-modifications Amazon CLI 命令。

以下列表显示了大多数数据库实例类可以为每个数据库引擎扩展到的最大存储：

- Db2 – 64TiB
- MariaDB – 64TiB
- Microsoft SQL Server – 64TiB
- MySQL – 64TiB
- Oracle – 64TiB
- PostgreSQL – 64TiB

下表显示了最大存储空间的一些例外情况（以 TiB 为单位）。所有独立于 io2 Block Express 存储的 RDS for Microsoft SQL Server 数据库实例的最大存储空间为 16 TiB，因此没有 SQL Server 的条目。

实例类	Db2	MariaDB	MySQL	Oracle	PostgreSQL
db.m3 – 标准实例类					
db.t4g – 具爆发能力的实例类					
db.t4g.medium	不适用	16	16	不适用	32
db.t4g.small	不适用	16	16	不适用	16
db.t4g.micro	不适用	6	6	不适用	6
db.t3 – 具爆发能力的实例类					

实例类	Db2	MariaDB	MySQL	Oracle	PostgreSQL
db.t3.medium	32	16	16	32	32
db.t3.small	32	16	16	32	16
db.t3.micro	不适用	6	6	32	6
db.t2 – 具爆发能力的实例类					

有关支持的所有实例类的更多详情，请参阅上一代数据库实例。

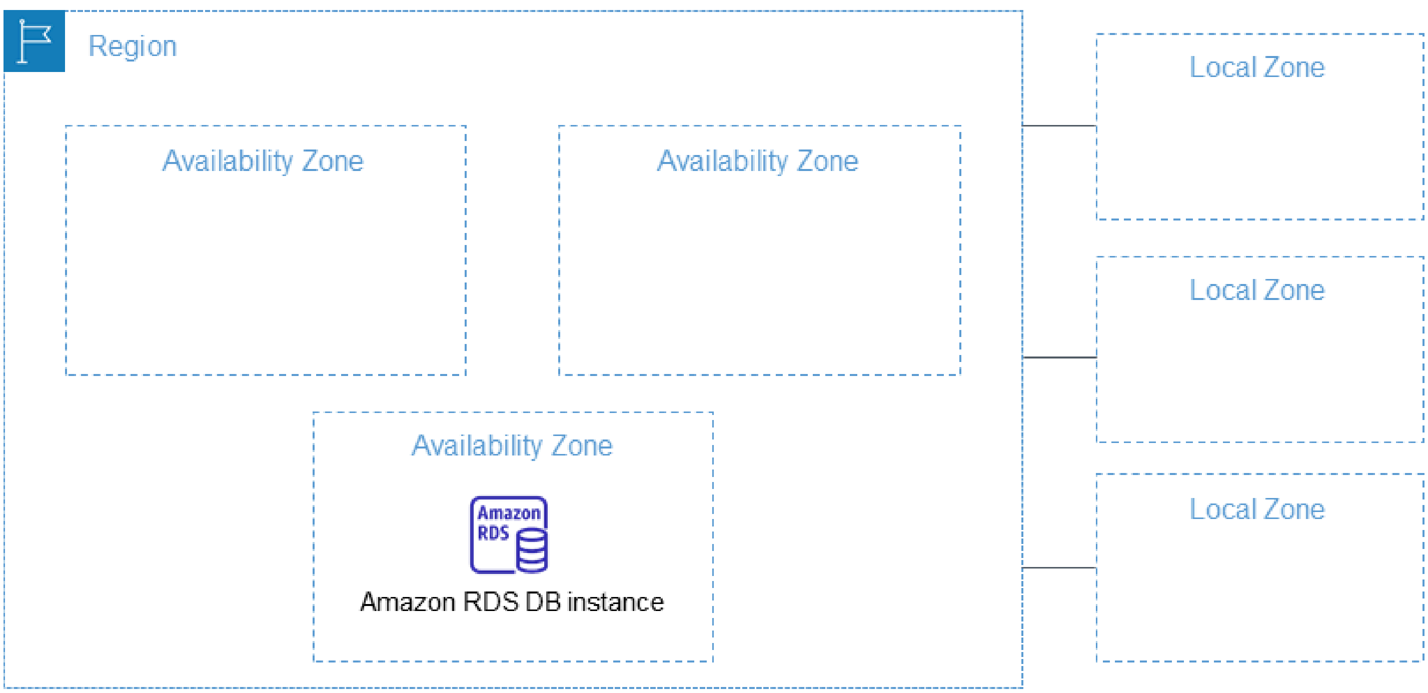
区域、可用区和 Local Zones

Amazon 云计算资源在全球多个位置托管。这些位置由 Amazon 区域、可用区和 Local Zones 构成。每个 Amazon 区域 都是一个单独的地理区域。每个 Amazon 区域都有多个相互隔离的位置，称为可用区。

> Note
>
> 有关查找 Amazon 区域的可用区的信息，请参阅 Amazon EC2 文档中的描述可用区。

通过使用 Local Zones，您在多个离用户较近的位置放置资源（如计算和存储）。借助 Amazon RDS，您可以将资源（如数据库实例）和数据放置在多个位置。除非您特意这样做，否则资源不会跨 Amazon 区域复制。

Amazon 运行着具有高可用性的先进数据中心。数据中心有时会发生影响托管于同一位置的所有数据库实例的可用性的故障，虽然这种故障极少发生。如果您将所有数据库实例都托管在受此类故障影响的同一个位置，则您的所有数据库实例都将不可用。



需要牢记的是，每个 Amazon 区域都是完全独立的，这一点非常重要。您启动的任何 Amazon RDS 活动（如创建数据库实例或列出可用的数据库实例）都只会在您当前的默认Amazon区域运行。可以在控制台中或通过设置 `AWS_DEFAULT_REGION` 环境变量更改原定设置 Amazon 区域。或者，可以通过

Amazon Command Line Interface（Amazon CLI）使用 `--region` 参数来覆盖它。有关更多信息，请参阅配置 Amazon Command Line Interface，特别是有关环境变量和命令行选项的章节。

Amazon RDS 支持名为 Amazon GovCloud (US) 的特殊 Amazon 区域。此类区域旨在使美国政府机构和客户可以将较为敏感的工作负载移至云中。Amazon GovCloud (US) 区域满足美国政府的特定法规和合规性要求。有关更多信息，请参阅什么是 Amazon GovCloud (US)？

要在特定Amazon区域中创建或使用 Amazon RDS 数据库实例，请使用相应的区域服务端点。

Amazon 地区

从设计而言，每个 Amazon 区域都与其他 Amazon 区域隔离。此设计可实现最大程度的容错能力和稳定性。

当您查看资源时，只会看到与您指定的 Amazon 区域关联的资源。这是因为 Amazon 区域间彼此隔离，而且我们不会自动跨 Amazon 区域复制资源。

区域可用性

下表显示了当前可以使用 Amazon RDS 的Amazon区域以及每个区域的端点。

区域名称	区域	端点	协议
美国东部（俄亥俄州）	us-east-2	rds.us-east-2.amazonaws.com	HTTPS
		rds-fips.us-east-2.api.aws	HTTPS
		rds.us-east-2.api.aws	HTTPS
		rds-fips.us-east-2.amazonaws.com	HTTPS
美国东部（弗吉尼亚州北部）	us-east-1	rds.us-east-1.amazonaws.com	HTTPS
		rds-fips.us-east-1.api.aws	HTTPS
		rds-fips.us-east-1.amazonaws.com	HTTPS
		rds.us-east-1.api.aws	HTTPS
美国西部（加利福	us-west-1	rds.us-west-1.amazonaws.com	HTTPS
		rds.us-west-1.api.aws	HTTPS

区域名称	区域	端点	协议
尼亚北部）		rds-fips.us-west-1.amazonaws.com	HTTPS
		rds-fips.us-west-1.api.aws	HTTPS
美国西部（俄勒冈州）	us-west-2	rds.us-west-2.amazonaws.com	HTTPS
		rds-fips.us-west-2.amazonaws.com	HTTPS
		rds.us-west-2.api.aws	HTTPS
		rds-fips.us-west-2.api.aws	HTTPS
非洲（开普敦）	af-south-1	rds.af-south-1.amazonaws.com	HTTPS
		rds.af-south-1.api.aws	HTTPS
亚太地区（香港）	ap-east-1	rds.ap-east-1.amazonaws.com	HTTPS
		rds.ap-east-1.api.aws	HTTPS
亚太地区（海得拉巴）	ap-south-2	rds.ap-south-2.amazonaws.com	HTTPS
		rds.ap-south-2.api.aws	HTTPS
亚太地区（雅加达）	ap-southeast-3	rds.ap-southeast-3.amazonaws.com	HTTPS
		rds.ap-southeast-3.api.aws	HTTPS
亚太地区（马来西亚）	ap-southeast-5	rds.ap-southeast-5.amazonaws.com	HTTPS
亚太地区（墨尔本）	ap-southeast-4	rds.ap-southeast-4.amazonaws.com	HTTPS
		rds.ap-southeast-4.api.aws	HTTPS
亚太地区（孟买）	ap-south-1	rds.ap-south-1.amazonaws.com	HTTPS
		rds.ap-south-1.api.aws	HTTPS

区域名称	区域	端点	协议
亚太地区（大阪）	ap-northeast-3	rds.ap-northeast-3.amazonaws.com rds.ap-northeast-3.api.aws	HTTPS HTTPS
亚太地区（首尔）	ap-northeast-2	rds.ap-northeast-2.amazonaws.com rds.ap-northeast-2.api.aws	HTTPS HTTPS
亚太地区（新加坡）	ap-southeast-1	rds.ap-southeast-1.amazonaws.com rds.ap-southeast-1.api.aws	HTTPS HTTPS
亚太地区（悉尼）	ap-southeast-2	rds.ap-southeast-2.amazonaws.com rds.ap-southeast-2.api.aws	HTTPS HTTPS
亚太地区（台北）	ap-east-2	rds.ap-east-2.amazonaws.com	HTTPS
亚太地区（泰国）	ap-southeast-7	rds.ap-southeast-7.amazonaws.com	HTTPS
亚太地区（东京）	ap-northeast-1	rds.ap-northeast-1.amazonaws.com rds.ap-northeast-1.api.aws	HTTPS HTTPS
加拿大（中部）	ca-central-1	rds.ca-central-1.amazonaws.com rds.ca-central-1.api.aws rds-fips.ca-central-1.api.aws rds-fips.ca-central-1.amazonaws.com	HTTPS HTTPS HTTPS HTTPS
加拿大西部（卡尔加里）	ca-west-1	rds.ca-west-1.amazonaws.com rds-fips.ca-west-1.amazonaws.com	HTTPS HTTPS

区域名称	区域	端点	协议
中国（北京）	cn-north-1	rds.cn-north-1.amazonaws.com.cn rds.cn-north-1.api.amazonwebservices.com.cn	HTTPS HTTPS
中国（宁夏）	cn-northwest-1	rds.cn-northwest-1.amazonaws.com.cn rds.cn-northwest-1.api.amazonwebservices.com.cn	HTTPS HTTPS
欧洲地区（法兰克福）	eu-central-1	rds.eu-central-1.amazonaws.com rds.eu-central-1.api.aws	HTTPS HTTPS
欧洲地区（爱尔兰）	eu-west-1	rds.eu-west-1.amazonaws.com rds.eu-west-1.api.aws	HTTPS HTTPS
欧洲地区（伦敦）	eu-west-2	rds.eu-west-2.amazonaws.com rds.eu-west-2.api.aws	HTTPS HTTPS
欧洲地区（米兰）	eu-south-1	rds.eu-south-1.amazonaws.com rds.eu-south-1.api.aws	HTTPS HTTPS
欧洲地区（巴黎）	eu-west-3	rds.eu-west-3.amazonaws.com rds.eu-west-3.api.aws	HTTPS HTTPS
欧洲（西班牙）	eu-south-2	rds.eu-south-2.amazonaws.com rds.eu-south-2.api.aws	HTTPS HTTPS
欧洲地区（斯德哥尔摩）	eu-north-1	rds.eu-north-1.amazonaws.com rds.eu-north-1.api.aws	HTTPS HTTPS

区域名称	区域	端点	协议
欧洲（苏黎世）	eu-central-2	rds.eu-central-2.amazonaws.com rds.eu-central-2.api.aws	HTTPS HTTPS
以色列（特拉维夫）	il-central-1	rds.il-central-1.amazonaws.com rds.il-central-1.api.aws	HTTPS HTTPS
墨西哥（中部）	mx-central-1	rds.mx-central-1.amazonaws.com	HTTPS
中东（巴林）	me-south-1	rds.me-south-1.amazonaws.com rds.me-south-1.api.aws	HTTPS HTTPS
中东（阿联酋）	me-central-1	rds.me-central-1.amazonaws.com rds.me-central-1.api.aws	HTTPS HTTPS
南美洲（圣保罗）	sa-east-1	rds.sa-east-1.amazonaws.com rds.sa-east-1.api.aws	HTTPS HTTPS

如果不明确指定端点，则美国西部（俄勒冈州）为默认端点。

使用 Amazon CLI 或 API 操作处理数据库实例时，请确保您指定了其区域端点。

可用区

创建数据库实例时，您可以选择可用区，也可以让 Amazon RDS 为您随机选择。可用区由 Amazon 区域代码后跟一个字母标识符表示；例如，`us-east-1a`。

使用如下的 describe-availability-zones Amazon EC2 命令描述在指定区域内已为您的账户启用的可用区。

```
aws ec2 describe-availability-zones --region region-name
```

例如，要描述在美国东部（弗吉尼亚州北部）区域（us-east-1）为您的账户启用的可用区域，请运行以下命令：

```
aws ec2 describe-availability-zones --region us-east-1
```

您不能为多可用区数据库部署中的主数据库实例和辅助数据库实例选择可用区。Amazon RDS 会随机为您选择。有关多可用区部署的更多信息，请参阅配置和管理 Amazon RDS 的多可用区部署。

> **Note**
>
> RDS 随机选择可用区并不能保证数据库实例在单个账户或数据库子网组内的可用区之间均匀分布。您可以在创建或修改单可用区实例时请求特定可用区，也可以将更具体的数据库子网组用于多可用区实例。有关更多信息，请参阅创建 Amazon RDS 数据库实例和修改 Amazon RDS 数据库实例。

Local Zones

本地扩展区 是在地理位置上靠近用户的 Amazon 区域的扩展。您可以将任何 VPC 从父 Amazon 区域扩展到本地区域。为此，请创建一个新子网并将其分配给 Amazon 本地区域。当您在本地扩展区中创建子网时，VPC 也会扩展到该本地扩展区。本地扩展区中的子网与 VPC 中其他子网的运行相同。

创建数据库实例时，您可以选择本地扩展区中的子网。Local Zones 有自己的 Internet 连接并支持 Amazon Direct Connect。因此，在本地扩展区中创建的资源可以通过非常低延迟的通信为本地用户提供服务。有关更多信息，请参阅 AmazonLocal Zones。

本地扩展区由 Amazon 区域代码后跟一个指示位置的标识符表示，例如 `us-west-2-lax-1a`。

> **Note**
>
> 本地扩展区不能包含在多可用区部署中。

使用本地扩展区

1. 在 Amazon EC2 控制台中启用本地扩展区。

   有关更多信息，请参阅 Amazon EC2 用户指南（适用于 Linux 实例） 中的启用 Local Zones。
2. 在本地扩展区中创建子网。

有关更多信息，请参阅 Amazon VPC 用户指南 中的在 VPC 中创建子网。

3. 在本地扩展区中创建数据库子网组。

   创建数据库子网组时，请为本地扩展区选择可用区组。

   有关更多信息，请参阅"在 VPC 中创建数据库实例"。

4. 创建使用本地扩展区中数据库子网组的数据库实例。

   有关更多信息，请参阅创建 Amazon RDS 数据库实例。

> **Important**
>
> 目前，唯一可用 Amazon RDS 的 Amazon 本地区域为美国西部（俄勒冈州）区域的洛杉矶。

Amazon RDS 中受 Amazon Web Services 区域和数据库引擎支持的功能

对 Amazon RDS 功能和选项的支持因 Amazon Web Services 区域和每个数据库引擎的特定版本而异。要确定给定 Amazon Web Services 区域中的 RDS 数据库引擎版本支持和可用性，可以使用以下部分。

Amazon RDS 功能不同于引擎原生功能和选项。有关引擎原生功能和选项的更多信息，请参阅引擎原生功能。

支持的区域和数据库引擎

表约定

这些功能部分中的表使用以下这些模式来指定版本号和可用性级别：

- 版本 x.y – 仅特定版本可用。
- 版本 x.y 及更高版本 – 支持指定的版本及其主要版本的所有更高次要版本。例如，“版本 10.11 及更高版本”表示 10.11、10.11.1 和 10.12 版本可用。

功能快速参考

以下快速参考表列出了每项功能和可用的 RDS 数据库引擎。区域和特定版本的可用性显示在后面的功能部分中。

功能	RDS for Db2	RDS for MariaDB	RDS for MySQL	RDS for Oracle	RDS for PostgreSQL	RDS for SQL Server
蓝绿部署	不可用	可用	可用	不可用	可用	不可用
跨区域自动备份	可用	可用	可用	可用	可用	可用
跨区域只读副本	不可用	可用	可用	可用	可用	可用
数据库活动流	不可用	不可用	不可用	可用	不可用	可用
双堆栈模式	不可用	可用	可用	可用	可用	可用
将快照导出到	不可用	可用	可用	不可用	可用	不可用

功能	RDS for Db2	RDS for MariaDB	RDS for MySQL	RDS for Oracle	RDS for PostgreSQL	RDS for SQL Server
Amazo S3						
Amazo Identit and Acces Manag t (IAM) 数据库身份验证	不可用	可用	可用	不可用	可用	不可用
Kerbe 身份验证	可用	不可用	可用	可用	可用	可用
多可用区数据库集群	不可用	不可用	可用	不可用	可用	不可用
性能详情	不可用	可用	可用	可用	可用	可用
RDS Custo	不可用	不可用	不可用	可用	不可用	可用
RDS 代理	不可用	可用	可用	不可用	可用	可用

功能	RDS for Db2	RDS for MariaDB	RDS for MySQL	RDS for Oracle	RDS for PostgreSQL	RDS for SQL Server
Secret Manag 集成	可用	可用	可用	可用	可用	可用

支持 Amazon RDS 蓝绿部署的区域和数据库引擎

蓝绿部署将生产数据库环境复制到单独的同步暂存环境中。通过使用 Amazon RDS 蓝绿部署，您可以在不影响生产环境的情况下对暂存环境中的数据库进行更改。例如，您可以升级主要或次要数据库引擎版本、更改数据库参数或在暂存环境中更改模式。准备就绪后，可以将暂存环境升级为新的生产数据库环境。有关更多信息，请参阅 使用 Amazon RDS 蓝绿部署进行数据库更新。

所有 Amazon Web Services 区域都支持蓝绿部署。

以下引擎不支持蓝绿部署：

- RDS for Db2
- RDS for SQL Server
- RDS for Oracle

主题

- 使用 RDS for MariaDB 进行蓝绿部署
- 使用 RDS for MySQL 进行蓝绿部署
- 使用 RDS for PostgreSQL 进行蓝绿部署

使用 RDS for MariaDB 进行蓝绿部署

对于 RDS for MariaDB，以下版本支持蓝绿部署：

- RDS for MariaDB 11.4（所有可用版本）
- RDS for MariaDB 10.2 及更高的 10 版本

使用 RDS for MySQL 进行蓝绿部署

对于 RDS for MySQL，以下版本支持蓝绿部署：

- RDS for MySQL 8.4（所有可用版本）
- RDS for MySQL 8.0（所有可用版本）
- RDS for MySQL 5.7（所有可用版本）

使用 RDS for PostgreSQL 进行蓝绿部署

对于 RDS for PostgreSQL，版本 11.1 以及所有更高的大版本和小版本都支持蓝绿部署。

> **Note**
>
> 在某些情况下，RDS for PostgreSQL 使用逻辑复制而不是物理复制来保持绿色环境与蓝色环境同步。有关更多信息，请参阅 the section called “PostgreSQL 复制方法”。

支持 Amazon RDS 中跨区域自动备份的区域和数据库引擎

通过在 Amazon RDS 中使用备份复制，您可以将 RDS 数据库实例配置为将快照和事务日志复制到目标区域。为数据库实例配置备份复制时，RDS 会在所有快照和事务日志准备就绪后对它们启动跨区域复制。有关更多信息，请参阅 将自动备份复制到其他 Amazon Web Services 区域。

有关备份复制可用的 Amazon Web Services 区域的信息，请参阅将自动备份复制到其他 Amazon Web Services 区域。

主题

适用于 RDS for Db2 的备份复制

Amazon RDS 对于所有当前可用的 RDS for Db2 版本都支持备份复制。

适用于 RDS for MariaDB 的备份复制

Amazon RDS 对于所有当前可用的 RDS for MariaDB 版本都支持备份复制。

适用于 RDS for MySQL 的备份复制

Amazon RDS 对于所有当前可用的 RDS for MySQL 版本都支持备份复制。

适用于 RDS for Oracle 的备份复制

Amazon RDS 对于所有当前可用的 RDS for Oracle 版本都支持备份复制。

适用于 RDS for PostgreSQL 的备份复制

Amazon RDS 对于所有当前可用的 RDS for PostgreSQL 版本都支持备份复制。

适用于 RDS for SQL Server 的备份复制

Amazon RDS 对于所有当前可用的 RDS for SQL Server 版本都支持备份复制。

支持 Amazon RDS 中跨区域只读副本的区域和数据库引擎

通过在 Amazon RDS 中使用跨区域只读副本，您可以在与源数据库实例不同的区域中创建 MariaDB、MySQL、Oracle、PostgreSQL 或 SQL Server 只读副本。有关跨区域只读副本的更多信息，包括源区域和目标区域注意事项，请参阅在不同的 Amazon Web Services 区域中创建只读副本。

跨区域只读副本不可用于以下引擎：

- RDS for Db2

主题

适用于 RDS for MariaDB 的跨区域只读副本

适用于 RDS for MariaDB 的跨区域只读副本可用于所有区域中的以下版本：

- RDS for MariaDB 11.4（所有可用版本）
- RDS for MariaDB 10.11（所有可用版本）
- RDS for MariaDB 10.6（所有可用版本）
- RDS for MariaDB 10.5（所有可用版本）
- RDS for MariaDB 10.4（所有可用版本）

适用于 RDS for MySQL 的跨区域只读副本

适用于 RDS for MySQL 的跨区域只读副本可用于所有区域中的以下版本：

- RDS for MySQL 8.4（所有可用版本）
- RDS for MySQL 8.0（所有可用版本）
- RDS for MySQL 5.7（所有可用版本）

适用于 RDS for Oracle 的跨区域只读副本

对于所有使用 Enterprise Edition 的受支持的数据库版本，适用于 RDS for Oracle 的跨区域只读副本在所有 Amazon Web Services 区域可用。仅非 CDB 和 CDB 架构的单租户配置支持副本。CDB 架构的多租户配置不支持跨区域只读副本。

有关使用 RDS for Oracle 的跨区域只读副本的其他要求的更多信息，请参阅RDS for Oracle 副本的要求和注意事项。

适用于 RDS for PostgreSQL 的跨区域只读副本

适用于 RDS for PostgreSQL 的跨区域只读副本可用于所有区域中的以下版本：

- RDS for PostgreSQL 17（所有可用版本）
- RDS for PostgreSQL 16（所有可用版本）
- RDS for PostgreSQL 15（所有可用版本）
- RDS for PostgreSQL 14（所有可用版本）
- RDS for PostgreSQL 13（所有可用版本）

- RDS for PostgreSQL 12（所有可用版本）
- RDS for PostgreSQL 11（所有可用版本）
- RDS for PostgreSQL 10（所有可用版本）

适用于 RDS for SQL Server 的跨区域只读副本

适用于 RDS for SQL Server 的跨区域只读副本在除以下区域外的所有区域均可用：

- 非洲（开普敦）
- 亚太地区（香港）
- 亚太地区（海得拉巴）
- 亚太地区（雅加达）
- 亚太地区（墨尔本）
- 加拿大西部（卡尔加里）
- 欧洲地区（米兰）
- 欧洲（西班牙）
- 欧洲（苏黎世）
- 以色列（特拉维夫）
- 中东（巴林）
- 中东（阿联酋）

适用于 RDS for SQL Server 的跨区域只读副本适用于以下使用 Microsoft SQL Server 企业版的版本：

- RDS for SQL Server 2022
- RDS for SQL Server 2019（版本 15.00.4073.23 及更高版本）
- RDS for SQL Server 2017（版本 14.00.3281.6 及更高版本）
- RDS for SQL Server 2016（版本 13.00.6300.2 及更高版本）

支持 Amazon RDS 中数据库活动流的区域和数据库引擎

通过利用 Amazon RDS 中的数据库活动流，您可以在 Oracle 数据库和 SQL Server 数据库中监控审计活动并为其设置警报。有关更多信息，请参阅 数据库活动流概览。

数据库活动流不可用于以下引擎：

- RDS for Db2
- RDS for MariaDB
- RDS for MySQL
- RDS for PostgreSQL

主题

适用于 RDS for Oracle 的数据库活动流

以下区域和引擎可用性可用于面向 RDS for Oracle 的数据库活动流。

有关 RDS For Oracle 的数据库活动流的其他要求的更多信息，请参阅数据库活动流概览。

区域	RDS for Oracle 21c	RDS for Oracle 19c
美国东部（弗吉尼亚州北部）	不可用	Oracle Database 19.0.0.0.ru-2019-07.rur-2019-07.r1 及更高版本 [使用企业版（EE）或标准版 2（SE2）]。
美国东部（俄亥俄州）	不可用	Oracle Database 19.0.0.0.ru-2019-07.rur-2019-07.r1 及更高版本 [使用企业版（EE）或标准版 2（SE2）]。
美国西部（加利福尼亚北部）	不可用	Oracle Database 19.0.0.0.ru-2019-07.rur-2019-07.r1 及更高版本 [使用企业版（EE）或标准版 2（SE2）]。
美国西部（俄勒冈州）	不可用	Oracle Database 19.0.0.0.ru-2019-07.rur-2019-07.r1 及更高版本 [使用企业版（EE）或标准版 2（SE2）]。
非洲（开普敦）	不可用	Oracle Database 19.0.0.0.ru-2019-07.rur-2019-07.r1 及更高版本 [使用企业版（EE）或标准版 2（SE2）]。

区域	RDS for Oracle 21c	RDS for Oracle 19c
亚太地区（香港）	不可用	Oracle Database 19.0.0.0.ru-2019-07.rur-2019-07.r1 及更高版本 [使用企业版（EE）或标准版 2（SE2）]。
亚太地区（海得拉巴）	不可用	Oracle Database 19.0.0.0.ru-2019-07.rur-2019-07.r1 及更高版本 [使用企业版（EE）或标准版 2（SE2）]。
亚太地区（雅加达）	不可用	Oracle Database 19.0.0.0.ru-2019-07.rur-2019-07.r1 及更高版本 [使用企业版（EE）或标准版 2（SE2）]。
亚太地区（马来西亚）	不可用	不可用
亚太地区（墨尔本）	不可用	不可用
亚太地区（孟买）	不可用	Oracle Database 19.0.0.0.ru-2019-07.rur-2019-07.r1 及更高版本 [使用企业版（EE）或标准版 2（SE2）]。
亚太地区（大阪）	不可用	Oracle Database 19.0.0.0.ru-2019-07.rur-2019-07.r1 及更高版本 [使用企业版（EE）或标准版 2（SE2）]。
亚太地区（首尔）	不可用	Oracle Database 19.0.0.0.ru-2019-07.rur-2019-07.r1 及更高版本 [使用企业版（EE）或标准版 2（SE2）]。
亚太地区（新加坡）	不可用	Oracle Database 19.0.0.0.ru-2019-07.rur-2019-07.r1 及更高版本 [使用企业版（EE）或标准版 2（SE2）]。
亚太地区（悉尼）	不可用	Oracle Database 19.0.0.0.ru-2019-07.rur-2019-07.r1 及更高版本 [使用企业版（EE）或标准版 2（SE2）]。

区域	RDS for Oracle 21c	RDS for Oracle 19c
Asia Pacific (Tokyo)	不可用	Oracle Database 19.0.0.0.ru-2019-07.rur-2019-07.r1 及更高版本 [使用企业版（EE）或标准版 2（SE2）]。
加拿大（中部）	不可用	Oracle Database 19.0.0.0.ru-2019-07.rur-2019-07.r1 及更高版本 [使用企业版（EE）或标准版 2（SE2）]。
加拿大西部（卡尔加里）	不可用	Oracle Database 19.0.0.0.ru-2019-07.rur-2019-07.r1 及更高版本 [使用企业版（EE）或标准版 2（SE2）]。
中国（北京）	不可用	不可用
中国（宁夏）	不可用	不可用
欧洲地区（法兰克福）	不可用	Oracle Database 19.0.0.0.ru-2019-07.rur-2019-07.r1 及更高版本 [使用企业版（EE）或标准版 2（SE2）]。
欧洲地区（爱尔兰）	不可用	Oracle Database 19.0.0.0.ru-2019-07.rur-2019-07.r1 及更高版本 [使用企业版（EE）或标准版 2（SE2）]。
欧洲地区（伦敦）	不可用	Oracle Database 19.0.0.0.ru-2019-07.rur-2019-07.r1 及更高版本 [使用企业版（EE）或标准版 2（SE2）]。
欧洲地区（米兰）	不可用	Oracle Database 19.0.0.0.ru-2019-07.rur-2019-07.r1 及更高版本 [使用企业版（EE）或标准版 2（SE2）]。
欧洲地区（巴黎）	不可用	Oracle Database 19.0.0.0.ru-2019-07.rur-2019-07.r1 及更高版本 [使用企业版（EE）或标准版 2（SE2）]。

区域	RDS for Oracle 21c	RDS for Oracle 19c
欧洲（西班牙）	不可用	Oracle Database 19.0.0.0.ru-2019-07.rur-2019-07.r1 及更高版本 [使用企业版（EE）或标准版 2（SE2）]。
欧洲地区（斯德哥尔摩）	不可用	Oracle Database 19.0.0.0.ru-2019-07.rur-2019-07.r1 及更高版本 [使用企业版（EE）或标准版 2（SE2）]。
欧洲（苏黎世）	不可用	不可用
亚太地区（墨尔本）	不可用	不可用
中东（巴林）	不可用	Oracle Database 19.0.0.0.ru-2019-07.rur-2019-07.r1 及更高版本 [使用企业版（EE）或标准版 2（SE2）]。
中东（阿联酋）	不可用	Oracle Database 19.0.0.0.ru-2019-07.rur-2019-07.r1 及更高版本 [使用企业版（EE）或标准版 2（SE2）]。
南美洲（圣保罗）	不可用	Oracle Database 19.0.0.0.ru-2019-07.rur-2019-07.r1 及更高版本 [使用企业版（EE）或标准版 2（SE2）]。
Amazon GovCloud（美国东部）	不可用	不可用
Amazon GovCloud（美国西部）	不可用	不可用

适用于 RDS for SQL Server 的数据库活动流

以下区域和引擎可用性可用于面向 RDS for SQL Server 的数据库活动流。

有关 RDS for SQL Server 的数据库活动流的其他要求的更多信息，请参阅数据库活动流概览。

区域	RDS for SQL Server 2019	RDS for SQL Server 2017	RDS for SQL Server 2016
美国东部（弗吉尼亚州北部）	所有可用版本	所有可用版本	所有可用版本
美国东部（俄亥俄州）	所有可用版本	所有可用版本	所有可用版本
美国西部（加利福尼亚北部）	所有可用版本	所有可用版本	所有可用版本
美国西部（俄勒冈州）	所有可用版本	所有可用版本	所有可用版本
非洲（开普敦）	所有可用版本	所有可用版本	所有可用版本
亚太地区（香港）	所有可用版本	所有可用版本	所有可用版本
亚太地区（海得拉巴）	所有可用版本	所有可用版本	所有可用版本
亚太地区（雅加达）	所有可用版本	所有可用版本	所有可用版本
亚太地区（马来西亚）	不可用	不可用	不可用
亚太地区（墨尔本）	不可用	不可用	不可用
亚太地区（孟买）	所有可用版本	所有可用版本	所有可用版本
亚太地区（大阪）	所有可用版本	所有可用版本	所有可用版本
亚太地区（首尔）	所有可用版本	所有可用版本	所有可用版本
亚太地区（新加坡）	所有可用版本	所有可用版本	所有可用版本
亚太地区（悉尼）	所有可用版本	所有可用版本	所有可用版本
Asia Pacific (Tokyo)	所有可用版本	所有可用版本	所有可用版本

区域	RDS for SQL Server 2019	RDS for SQL Server 2017	RDS for SQL Server 2016
加拿大（中部）	所有可用版本	所有可用版本	所有可用版本
加拿大西部（卡尔加里）	所有可用版本	所有可用版本	所有可用版本
中国（北京）	不可用	不可用	不可用
中国（宁夏）	不可用	不可用	不可用
欧洲地区（法兰克福）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（爱尔兰）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（伦敦）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（米兰）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（巴黎）	所有可用版本	所有可用版本	所有可用版本
欧洲（西班牙）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（斯德哥尔摩）	所有可用版本	所有可用版本	所有可用版本
欧洲（苏黎世）	不可用	不可用	不可用
以色列（特拉维夫）	不可用	不可用	不可用
中东（巴林）	所有可用版本	所有可用版本	所有可用版本
中东（阿联酋）	所有可用版本	所有可用版本	所有可用版本
南美洲（圣保罗）	所有可用版本	所有可用版本	所有可用版本
Amazon GovCloud（美国东部）	不可用	不可用	不可用

区域	RDS for SQL Server 2019	RDS for SQL Server 2017	RDS for SQL Server 2016
Amazon GovCloud (美国西部)	不可用	不可用	不可用

支持 Amazon RDS 中双堆栈模式的区域和数据库引擎

通过在 RDS 中使用双堆栈模式，资源可通过互联网协议版本 4（IPv4）和/或互联网协议版本 6（IPv6）与数据库实例进行通信。有关更多信息，请参阅 双堆栈模式。

主题

- 适用于 RDS for Db2 的双堆栈模式
- 适用于 RDS for MariaDB 的双堆栈模式
- 适用于 RDS for MySQL 的双堆栈模式
- 适用于 RDS for Oracle 的双堆栈模式
- 适用于 RDS for PostgreSQL 的双堆栈模式
- 适用于 RDS for SQL Server 的双堆栈模式

以下 Amazon Web Services 区域中的任何 Amazon RDS 数据库引擎均不支持 IPv6：

- 亚太地区（海得拉巴）
- 中东（阿联酋）
- 亚太地区（马来西亚）
- 亚太地区（墨尔本）
- 加拿大西部（卡尔加里）
- 欧洲（西班牙）
- 欧洲（苏黎世）
- 以色列（特拉维夫）

适用于 RDS for Db2 的双堆栈模式

以下区域和引擎版本可用于面向 RDS for Db2 的双堆栈模式。

区域	RDS for Db2 11.5				
美国东部（弗吉尼亚州北部）	所有可用版本				
美国东部（俄亥俄州）	所有可用版本				
美国西部（加利福尼亚北部）	所有可用版本				
美国西部（俄勒冈州）	所有可用版本				
非洲（开普敦）	所有可用版本				
亚太地区（香港）	所有可用版本				
亚太地区（海得拉巴）	不可用				
亚太地区（雅加达）	所有可用版本				
亚太地区（马来西亚）	不可用				
亚太地区（墨尔本）	不可用				
亚太地区（孟买）	所有可用版本				

区域	RDS for Db2 11.5
亚太地区（大阪）	所有可用版本
亚太地区（首尔）	所有可用版本
亚太地区（新加坡）	所有可用版本
亚太地区（悉尼）	所有可用版本
亚太地区（东京)	所有可用版本
加拿大（中部）	所有可用版本
加拿大西部（卡尔加里）	不可用
中国（北京）	不可用
中国（宁夏）	不可用
欧洲地区（法兰克福）	所有可用版本
欧洲地区（爱尔兰）	所有可用版本
欧洲地区（伦敦）	所有可用版本
欧洲地区（米兰）	所有可用版本

区域	RDS for Db2 11.5				
欧洲地区（巴黎）	所有可用版本				
欧洲地区（西班牙）	不可用				
欧洲地区（斯德哥尔摩）	所有可用版本				
欧洲（苏黎世）	不可用				
以色列（特拉维夫）	不可用				
中东（巴林）	所有可用版本				
中东（阿联酋）	不可用				
南美洲（圣保罗）	所有可用版本				
Amazon GovCloud（美国东部）	不可用				
Amazon GovCloud（美国西部）	不可用				

适用于 RDS for MariaDB 的双堆栈模式

以下区域和引擎版本可用于面向 RDS for MariaDB 的双堆栈模式。

区域	RDS for MariaDB 11.4	RDS for MariaDB 10.11	RDS for MariaDB 10.6	RDS for MariaDB 10.5	RDS for MariaDB 10.4
美国东部（弗吉尼亚州北部）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
美国东部（俄亥俄州）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
美国西部（加利福尼亚北部）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
美国西部（俄勒冈州）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
非洲（开普敦）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（香港）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（海得拉巴）	不可用	不可用	不可用	不可用	不可用
亚太地区（雅加达）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（马来西亚）	不可用	不可用	不可用	不可用	不可用
亚太地区（墨尔本）	不可用	不可用	不可用	不可用	不可用
亚太地区（孟买）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本

区域	RDS for MariaDB 11.4	RDS for MariaDB 10.11	RDS for MariaDB 10.6	RDS for MariaDB 10.5	RDS for MariaDB 10.4
亚太地区（大阪）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（首尔）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（新加坡）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（悉尼）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（东京)	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
加拿大（中部）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
加拿大西部（卡尔加里）	不可用	不可用	不可用	不可用	不可用
中国（北京）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
中国（宁夏）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（法兰克福）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（爱尔兰）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（伦敦）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（米兰）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本

区域	RDS for MariaDB 11.4	RDS for MariaDB 10.11	RDS for MariaDB 10.6	RDS for MariaDB 10.5	RDS for MariaDB 10.4
欧洲地区（巴黎）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（西班牙）	不可用	不可用	不可用	不可用	不可用
欧洲地区（斯德哥尔摩）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲（苏黎世）	不可用	不可用	不可用	不可用	不可用
以色列（特拉维夫）	不可用	不可用	不可用	不可用	不可用
中东（巴林）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
中东（阿联酋）	不可用	不可用	不可用	不可用	不可用
南美洲（圣保罗）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
Amazon GovCloud（美国东部）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
Amazon GovCloud（美国西部）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本

适用于 RDS for MySQL 的双堆栈模式

以下区域和引擎版本可用于面向 RDS for MySQL 的双堆栈模式。

区域	RDS for MySQL 8.4	RDS for MySQL 8.0	RDS for MySQL 5.7（在 RDS 扩展支持下）
美国东部（弗吉尼亚州北部）	所有可用版本	所有可用版本	所有可用版本
美国东部（俄亥俄州）	所有可用版本	所有可用版本	所有可用版本
美国西部（加利福尼亚北部）	所有可用版本	所有可用版本	所有可用版本
美国西部（俄勒冈州）	所有可用版本	所有可用版本	所有可用版本
非洲（开普敦）	所有可用版本	所有可用版本	所有可用版本
亚太地区（香港）	所有可用版本	所有可用版本	所有可用版本
亚太地区（海得拉巴）	不可用	不可用	不可用
亚太地区（雅加达）	所有可用版本	所有可用版本	所有可用版本
亚太地区（马来西亚）	不可用	不可用	不可用
亚太地区（墨尔本）	不可用	不可用	不可用
亚太地区（孟买）	所有可用版本	所有可用版本	所有可用版本
亚太地区（大阪）	所有可用版本	所有可用版本	所有可用版本
亚太地区（首尔）	所有可用版本	所有可用版本	所有可用版本
亚太地区（新加坡）	所有可用版本	所有可用版本	所有可用版本
亚太地区（悉尼）	所有可用版本	所有可用版本	所有可用版本

区域	RDS for MySQL 8.4	RDS for MySQL 8.0	RDS for MySQL 5.7（在 RDS 扩展支持下）
亚太地区（东京)	所有可用版本	所有可用版本	所有可用版本
加拿大（中部）	所有可用版本	所有可用版本	所有可用版本
加拿大西部（卡尔加里）	不可用	不可用	不可用
中国（北京）	所有可用版本	所有可用版本	所有可用版本
中国（宁夏）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（法兰克福）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（爱尔兰）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（伦敦）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（米兰）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（巴黎）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（西班牙）	不可用	不可用	不可用
欧洲地区（斯德哥尔摩）	所有可用版本	所有可用版本	所有可用版本
欧洲（苏黎世）	不可用	不可用	不可用
以色列（特拉维夫）	不可用	不可用	不可用
中东（巴林）	所有可用版本	所有可用版本	所有可用版本
中东（阿联酋）	不可用	不可用	不可用
南美洲（圣保罗）	所有可用版本	所有可用版本	所有可用版本

区域	RDS for MySQL 8.4	RDS for MySQL 8.0	RDS for MySQL 5.7（在 RDS 扩展支持下）
Amazon GovCloud (美国东部）	所有可用版本	所有可用版本	所有可用版本
Amazon GovCloud (美国西部）	所有可用版本	所有可用版本	所有可用版本

适用于 RDS for Oracle 的双堆栈模式

以下区域和引擎版本可用于面向 RDS for Oracle 的双堆栈模式。

区域	RDS for Oracle 21c	RDS for Oracle 19c
美国东部（弗吉尼亚州北部）	所有可用版本	所有可用版本
美国东部（俄亥俄州）	所有可用版本	所有可用版本
美国西部（加利福尼亚北部）	所有可用版本	所有可用版本
美国西部（俄勒冈州）	所有可用版本	所有可用版本
非洲（开普敦）	所有可用版本	所有可用版本
亚太地区（香港）	所有可用版本	所有可用版本
亚太地区（海得拉巴）	不可用	不可用
亚太地区（雅加达）	所有可用版本	所有可用版本
亚太地区（马来西亚）	不可用	不可用
亚太地区（墨尔本）	不可用	不可用
亚太地区（孟买）	所有可用版本	所有可用版本
亚太地区（大阪）	所有可用版本	所有可用版本

区域	RDS for Oracle 21c	RDS for Oracle 19c
亚太地区（首尔）	所有可用版本	所有可用版本
亚太地区（新加坡）	所有可用版本	所有可用版本
亚太地区（悉尼）	所有可用版本	所有可用版本
亚太地区（东京)	所有可用版本	所有可用版本
加拿大（中部）	所有可用版本	所有可用版本
加拿大西部（卡尔加里）	不可用	不可用
中国（北京）	所有可用版本	所有可用版本
中国（宁夏）	所有可用版本	所有可用版本
欧洲地区（法兰克福）	所有可用版本	所有可用版本
欧洲地区（爱尔兰）	所有可用版本	所有可用版本
欧洲地区（伦敦）	所有可用版本	所有可用版本
欧洲地区（米兰）	所有可用版本	所有可用版本
欧洲地区（巴黎）	所有可用版本	所有可用版本
欧洲地区（西班牙）	不可用	不可用
欧洲地区（斯德哥尔摩）	所有可用版本	所有可用版本
欧洲（苏黎世）	不可用	不可用
以色列（特拉维夫）	不可用	不可用
中东（巴林）	所有可用版本	所有可用版本
中东（阿联酋）	不可用	不可用
南美洲（圣保罗）	所有可用版本	所有可用版本

区域	RDS for Oracle 21c	RDS for Oracle 19c
Amazon GovCloud（美国东部）	所有可用版本	所有可用版本
Amazon GovCloud（美国西部）	所有可用版本	所有可用版本

适用于 RDS for PostgreSQL 的双堆栈模式

以下区域和引擎版本可用于面向 RDS for PostgreSQL 的双堆栈模式。

区域	RDS for PostgreSQL 17	RDS for PostgreSQL 16	RDS for PostgreSQL 15	RDS for PostgreSQL 14	RDS for PostgreSQL 13	RDS for PostgreSQL 12	RDS for PostgreSQL 11	RDS for PostgreSQL 10
美国东部（弗吉尼亚州北部）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
美国东部（俄亥俄州）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
美国西部（加利福尼亚北部）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
美国西部（俄勒冈州）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本

区域	RDS for PostgreSQL 17	RDS for PostgreSQL 16	RDS for PostgreSQL 15	RDS for PostgreSQL 14	RDS for PostgreSQL 13	RDS for PostgreSQL 12	RDS for PostgreSQL 11	RDS for PostgreSQL 10
非洲（开普敦）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（香港）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（海得拉巴）	不可用	不可用	不可用	不可用	不可用	不可用	不可用	不可用
亚太地区（雅加达）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（马来西亚）	不可用	不可用	不可用	不可用	不可用	不可用	不可用	不可用
亚太地区（墨尔本）	不可用	不可用	不可用	不可用	不可用	不可用	不可用	不可用
亚太地区（孟买）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（大阪）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本

区域	RDS for PostgreSQL 17	RDS for PostgreSQL 16	RDS for PostgreSQL 15	RDS for PostgreSQL 14	RDS for PostgreSQL 13	RDS for PostgreSQL 12	RDS for PostgreSQL 11	RDS for PostgreSQL 10
亚太地区（首尔）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（新加坡）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（悉尼）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（东京)	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
加拿大（中部）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
加拿大西部（卡尔加里）	不可用	不可用	不可用	不可用	不可用	不可用	不可用	不可用
中国（北京）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
中国（宁夏）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本

区域	RDS for PostgreSQL 17	RDS for PostgreSQL 16	RDS for PostgreSQL 15	RDS for PostgreSQL 14	RDS for PostgreSQL 13	RDS for PostgreSQL 12	RDS for PostgreSQL 11	RDS for PostgreSQL 10
欧洲地区（法兰克福）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（爱尔兰）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（伦敦）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（米兰）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（巴黎）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（西班牙）	不可用	不可用	不可用	不可用	不可用	不可用	不可用	不可用
欧洲地区（斯德哥尔摩）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲（苏黎世）	不可用	不可用	不可用	不可用	不可用	不可用	不可用	不可用

区域	RDS for PostgreSQL 17	RDS for PostgreSQL 16	RDS for PostgreSQL 15	RDS for PostgreSQL 14	RDS for PostgreSQL 13	RDS for PostgreSQL 12	RDS for PostgreSQL 11	RDS for PostgreSQL 10
以色列（特拉维夫）	不可用	不可用	不可用	不可用	不可用	不可用	不可用	不可用
中东（巴林）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
中东（阿联酋）	不可用	不可用	不可用	不可用	不可用	不可用	不可用	不可用
南美洲（圣保罗）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
Amazon GovCloud 美国东部）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
Amazon GovCloud 美国西部）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本

适用于 RDS for SQL Server 的双堆栈模式

以下区域和引擎版本可用于面向 RDS for SQL Server 的双堆栈模式。

区域	RDS for SQL Server 2019	RDS for SQL Server 2017	RDS for SQL Server 2016
美国东部（弗吉尼亚州北部）	所有可用版本	所有可用版本	所有可用版本
美国东部（俄亥俄州）	所有可用版本	所有可用版本	所有可用版本
美国西部（加利福尼亚北部）	所有可用版本	所有可用版本	所有可用版本
美国西部（俄勒冈州）	所有可用版本	所有可用版本	所有可用版本
非洲（开普敦）	所有可用版本	所有可用版本	所有可用版本
亚太地区（香港）	所有可用版本	所有可用版本	所有可用版本
亚太地区（海得拉巴）	不可用	不可用	不可用
亚太地区（雅加达）	所有可用版本	所有可用版本	所有可用版本
亚太地区（马来西亚）	不可用	不可用	不可用
亚太地区（墨尔本）	不可用	不可用	不可用
亚太地区（孟买）	所有可用版本	所有可用版本	所有可用版本
亚太地区（大阪）	所有可用版本	所有可用版本	所有可用版本
亚太地区（首尔）	所有可用版本	所有可用版本	所有可用版本
亚太地区（新加坡）	所有可用版本	所有可用版本	所有可用版本
亚太地区（悉尼）	所有可用版本	所有可用版本	所有可用版本
亚太地区（东京)	所有可用版本	所有可用版本	所有可用版本

区域	RDS for SQL Server 2019	RDS for SQL Server 2017	RDS for SQL Server 2016
加拿大（中部）	所有可用版本	所有可用版本	所有可用版本
加拿大西部（卡尔加里）	不可用	不可用	不可用
中国（北京）	所有可用版本	所有可用版本	所有可用版本
中国（宁夏）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（法兰克福）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（爱尔兰）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（伦敦）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（米兰）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（巴黎）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（西班牙）	不可用	不可用	不可用
欧洲地区（斯德哥尔摩）	所有可用版本	所有可用版本	所有可用版本
欧洲（苏黎世）	不可用	不可用	不可用
以色列（特拉维夫）	不可用	不可用	不可用
中东（巴林）	所有可用版本	所有可用版本	所有可用版本
中东（阿联酋）	不可用	不可用	不可用
南美洲（圣保罗）	所有可用版本	所有可用版本	所有可用版本
Amazon GovCloud（美国东部）	所有可用版本	所有可用版本	所有可用版本

区域	RDS for SQL Server 2019	RDS for SQL Server 2017	RDS for SQL Server 2016
Amazon GovCloud (美国西部)	所有可用版本	所有可用版本	所有可用版本

支持在 Amazon RDS 中将快照导出到 S3 的区域和数据库引擎

您可以将 RDS 数据库快照数据导出到 Amazon S3 存储桶。您可以导出所有类型的数据库快照 – 包括手动快照、自动系统快照和 Amazon Backup 创建的快照。导出数据后，您可以通过 Amazon Athena 或 Amazon Redshift Spectrum 等工具直接分析导出的数据。有关更多信息，请参阅 将 Amazon RDS 的数据库快照数据导出到 Amazon S3。

将快照导出到 S3 的功能不可用于以下引擎：

- RDS for Db2
- RDS for Oracle
- RDS for SQL Server

主题

- 使用 RDS for MariaDB 将快照导出到 S3
- 使用 RDS for MySQL 将快照导出到 S3
- 使用 RDS for PostgreSQL 将快照导出到 S3

使用 RDS for MariaDB 将快照导出到 S3

以下区域和引擎版本可用于面向 RDS for MariaDB 的将快照导出到 S3 功能。

区域	RDS for MariaDB 11.4	RDS for MariaDB 10.11	RDS for MariaDB 10.6	RDS for MariaDB 10.5	RDS for MariaDB 10.4
美国东部（弗吉尼亚州北部）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本

区域	RDS for MariaDB 11.4	RDS for MariaDB 10.11	RDS for MariaDB 10.6	RDS for MariaDB 10.5	RDS for MariaDB 10.4
美国东部（俄亥俄州）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
美国西部（加利福尼亚北部）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
美国西部（俄勒冈州）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
非洲（开普敦）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（香港）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（海得拉巴）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（雅加达）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（马来西亚）	不可用	不可用	不可用	不可用	不可用
亚太地区（墨尔本）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（孟买）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（大阪）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本

区域	RDS for MariaDB 11.4	RDS for MariaDB 10.11	RDS for MariaDB 10.6	RDS for MariaDB 10.5	RDS for MariaDB 10.4
亚太地区（首尔）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（新加坡）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（悉尼）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（东京)	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
加拿大（中部）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
加拿大西部（卡尔加里）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
中国（北京）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
中国（宁夏）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（法兰克福）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（爱尔兰）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（伦敦）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（米兰）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（巴黎）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本

区域	RDS for MariaDB 11.4	RDS for MariaDB 10.11	RDS for MariaDB 10.6	RDS for MariaDB 10.5	RDS for MariaDB 10.4
欧洲地区（西班牙）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（斯德哥尔摩）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲（苏黎世）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
以色列（特拉维夫）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
中东（巴林）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
中东（阿联酋）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
南美洲（圣保罗）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
Amazon GovCloud（美国东部）	不可用	不可用	不可用	不可用	不可用
Amazon GovCloud（美国西部）	不可用	不可用	不可用	不可用	不可用

使用 RDS for MySQL 将快照导出到 S3

以下区域和引擎版本适用于通过 RDS for MySQL 将快照导出到 S3 的过程。

区域	RDS for MySQL 8.4	RDS for MySQL 8.0	RDS for MySQL 5.7（在 RDS 扩展支持下）
美国东部（俄亥俄州）	所有可用版本	所有可用版本	所有可用版本
美国东部（弗吉尼亚州北部）	所有可用版本	所有可用版本	所有可用版本
美国西部（加利福尼亚北部）	所有可用版本	所有可用版本	所有可用版本
美国西部（俄勒冈州）	所有可用版本	所有可用版本	所有可用版本
非洲（开普敦）	所有可用版本	所有可用版本	所有可用版本
亚太地区（香港）	所有可用版本	所有可用版本	所有可用版本
亚太地区（海得拉巴）	所有可用版本	所有可用版本	所有可用版本
亚太地区（雅加达）	所有可用版本	所有可用版本	所有可用版本
亚太地区（马来西亚）	不可用	不可用	不可用
亚太地区（墨尔本）	所有可用版本	所有可用版本	所有可用版本
亚太地区（孟买）	所有可用版本	所有可用版本	所有可用版本
亚太地区（大阪）	所有可用版本	所有可用版本	所有可用版本
亚太地区（首尔）	所有可用版本	所有可用版本	所有可用版本
亚太地区（新加坡）	所有可用版本	所有可用版本	所有可用版本
亚太地区（悉尼）	所有可用版本	所有可用版本	所有可用版本

区域	RDS for MySQL 8.4	RDS for MySQL 8.0	RDS for MySQL 5.7（在 RDS 扩展支持下）
亚太地区（东京)	所有可用版本	所有可用版本	所有可用版本
加拿大（中部）	所有可用版本	所有可用版本	所有可用版本
加拿大西部（卡尔加里）	所有可用版本	所有可用版本	所有可用版本
中国（北京）	所有可用版本	所有可用版本	所有可用版本
中国（宁夏）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（法兰克福）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（爱尔兰）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（伦敦）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（米兰）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（巴黎）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（西班牙）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（斯德哥尔摩）	所有可用版本	所有可用版本	所有可用版本
欧洲（苏黎世）	所有可用版本	所有可用版本	所有可用版本
以色列（特拉维夫）	所有可用版本	所有可用版本	所有可用版本
中东（巴林）	所有可用版本	所有可用版本	所有可用版本
中东（阿联酋）	所有可用版本	所有可用版本	所有可用版本
南美洲（圣保罗）	所有可用版本	所有可用版本	所有可用版本

区域	RDS for MySQL 8.4	RDS for MySQL 8.0	RDS for MySQL 5.7（在 RDS 扩展支持下）
Amazon GovCloud (美国东部)	不可用	不可用	不可用
Amazon GovCloud (美国西部)	不可用	不可用	不可用

使用 RDS for PostgreSQL 将快照导出到 S3

以下区域和引擎版本可用于面向 RDS for PostgreSQL 的将快照导出到 S3 功能。

区域	RDS for PostgreSQL 17	RDS for PostgreSQL 16	RDS for PostgreSQL 15	RDS for PostgreSQL 14	RDS for PostgreSQL 13	RDS for PostgreSQL 12	RDS for PostgreSQL 11	RDS for PostgreSQL 10
美国东部（俄亥俄州）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
美国东部（弗吉尼亚州北部）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
美国西部（加利福尼亚北部）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
美国西部（俄	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本

区域	RDS for PostgreSQL 17	RDS for PostgreSQL 16	RDS for PostgreSQL 15	RDS for PostgreSQL 14	RDS for PostgreSQL 13	RDS for PostgreSQL 12	RDS for PostgreSQL 11	RDS for PostgreSQL 10
勒冈州）								
非洲（开普敦）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（香港）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（海得拉巴）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（雅加达）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（马来西亚）	不可用	不可用	不可用	不可用	不可用	不可用	不可用	不可用
亚太地区（墨尔本）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（孟买）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本

区域	RDS for PostgreSQL 17	RDS for PostgreSQL 16	RDS for PostgreSQL 15	RDS for PostgreSQL 14	RDS for PostgreSQL 13	RDS for PostgreSQL 12	RDS for PostgreSQL 11	RDS for PostgreSQL 10
亚太地区（大阪）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（首尔）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（新加坡）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（悉尼）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（东京)	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
加拿大（中部）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
加拿大西部（卡尔加里）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
中国（北京）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本

区域	RDS for PostgreSQL 17	RDS for PostgreSQL 16	RDS for PostgreSQL 15	RDS for PostgreSQL 14	RDS for PostgreSQL 13	RDS for PostgreSQL 12	RDS for PostgreSQL 11	RDS for PostgreSQL 10
中国（宁夏）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（法兰克福）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（爱尔兰）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（伦敦）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（米兰）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（巴黎）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（西班牙）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（斯德哥尔摩）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本

区域	RDS for PostgreSQL 17	RDS for PostgreSQL 16	RDS for PostgreSQL 15	RDS for PostgreSQL 14	RDS for PostgreSQL 13	RDS for PostgreSQL 12	RDS for PostgreSQL 11	RDS for PostgreSQL 10
欧洲（苏黎世）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
以色列（特拉维夫）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
中东（巴林）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
中东（阿联酋）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
南美洲（圣保罗）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
Amazon GovCloud 美国东部）	不可用	不可用	不可用	不可用	不可用	不可用	不可用	不可用
Amazon GovCloud 美国西部）	不可用	不可用	不可用	不可用	不可用	不可用	不可用	不可用

支持 Amazon RDS 中 IAM 数据库身份验证功能的区域和数据库引擎

通过在 Amazon RDS 中使用 IAM 数据库身份验证，您在连接到数据库实例时无需密码即可进行身份验证。而是使用身份验证令牌。有关更多信息，请参阅 适用于 MariaDB、MySQL 和 PostgreSQL 的 IAM 数据库身份验证。

IAM 数据库身份验证不可用于以下引擎：

- RDS for Db2
- RDS for Oracle
- RDS for SQL Server

主题

- 适用于 RDS for MariaDB 的 IAM 数据库身份验证
- 适用于 RDS for MySQL 的 IAM 数据库身份验证
- 适用于 RDS for PostgreSQL 的 IAM 数据库身份验证

适用于 RDS for MariaDB 的 IAM 数据库身份验证

以下区域和引擎版本可用于面向 RDS for MariaDB 的 IAM 数据库身份验证。

区域	RDS for MariaDB 11.4	RDS for MariaDB 10.11	RDS for MariaDB 10.6	RDS for MariaDB 10.5	RDS for MariaDB 10.4
美国东部（弗吉尼亚州北部）	所有可用版本	所有可用版本	所有可用版本	不可用	不可用
美国东部（俄亥俄州）	所有可用版本	所有可用版本	所有可用版本	不可用	不可用
美国西部（加利福尼亚北部）	所有可用版本	所有可用版本	所有可用版本	不可用	不可用

区域	RDS for MariaDB 11.4	RDS for MariaDB 10.11	RDS for MariaDB 10.6	RDS for MariaDB 10.5	RDS for MariaDB 10.4
美国西部（俄勒冈州）	所有可用版本	所有可用版本	所有可用版本	不可用	不可用
非洲（开普敦）	所有可用版本	所有可用版本	所有可用版本	不可用	不可用
亚太地区（香港）	所有可用版本	所有可用版本	所有可用版本	不可用	不可用
亚太地区（海得拉巴）	不可用	不可用	不可用	不可用	不可用
亚太地区（雅加达）	所有可用版本	所有可用版本	所有可用版本	不可用	不可用
亚太地区（马来西亚）	所有可用版本	所有可用版本	不可用	不可用	不可用
亚太地区（墨尔本）	不可用	不可用	不可用	不可用	不可用
亚太地区（孟买）	所有可用版本	所有可用版本	所有可用版本	不可用	不可用
亚太地区（大阪）	所有可用版本	所有可用版本	所有可用版本	不可用	不可用
亚太地区（首尔）	所有可用版本	所有可用版本	所有可用版本	不可用	不可用
亚太地区（新加坡）	所有可用版本	所有可用版本	所有可用版本	不可用	不可用
亚太地区（悉尼）	所有可用版本	所有可用版本	所有可用版本	不可用	不可用

区域	RDS for MariaDB 11.4	RDS for MariaDB 10.11	RDS for MariaDB 10.6	RDS for MariaDB 10.5	RDS for MariaDB 10.4
亚太地区（东京)	所有可用版本	所有可用版本	所有可用版本	不可用	不可用
加拿大（中部）	所有可用版本	所有可用版本	所有可用版本	不可用	不可用
加拿大西部（卡尔加里）	所有可用版本	所有可用版本	所有可用版本	不可用	不可用
中国（北京）	所有可用版本	所有可用版本	所有可用版本	不可用	不可用
中国（宁夏）	所有可用版本	所有可用版本	所有可用版本	不可用	不可用
欧洲地区（法兰克福）	所有可用版本	所有可用版本	所有可用版本	不可用	不可用
欧洲地区（爱尔兰）	所有可用版本	所有可用版本	所有可用版本	不可用	不可用
欧洲地区（伦敦）	所有可用版本	所有可用版本	所有可用版本	不可用	不可用
欧洲地区（米兰）	所有可用版本	所有可用版本	所有可用版本	不可用	不可用
欧洲地区（巴黎）	所有可用版本	所有可用版本	所有可用版本	不可用	不可用
欧洲地区（西班牙）	不可用	不可用	不可用	不可用	不可用
欧洲地区（斯德哥尔摩）	所有可用版本	所有可用版本	所有可用版本	不可用	不可用
欧洲（苏黎世）	不可用	不可用	不可用	不可用	不可用

区域	RDS for MariaDB 11.4	RDS for MariaDB 10.11	RDS for MariaDB 10.6	RDS for MariaDB 10.5	RDS for MariaDB 10.4
以色列（特拉维夫）	不可用	不可用	不可用	不可用	不可用
中东（巴林）	所有可用版本	所有可用版本	所有可用版本	不可用	不可用
中东（阿联酋）	不可用	不可用	不可用	不可用	不可用
南美洲（圣保罗）	所有可用版本	所有可用版本	所有可用版本	不可用	不可用
Amazon GovCloud（美国东部）	所有可用版本	所有可用版本	所有可用版本	不可用	不可用
Amazon GovCloud（美国西部）	所有可用版本	所有可用版本	所有可用版本	不可用	不可用

适用于 RDS for MySQL 的 IAM 数据库身份验证

适用于 RDS for MySQL 的 IAM 数据库身份验证可用于所有区域中的以下版本：

- RDS for MySQL 8.4 – 所有可用版本
- RDS for MySQL 8.0 – 所有可用版本
- RDS for MySQL 5.7 – 所有可用版本

适用于 RDS for PostgreSQL 的 IAM 数据库身份验证

适用于 RDS for PostgreSQL 的 IAM 数据库身份验证可用于所有区域中的以下版本：

- RDS for PostgreSQL 17 – 所有可用版本
- RDS for PostgreSQL 16 – 所有可用版本

- RDS for PostgreSQL 15 – 所有可用版本
- RDS for PostgreSQL 14 – 所有可用版本
- RDS for PostgreSQL 13 – 所有可用版本
- RDS for PostgreSQL 12 – 所有可用版本
- RDS for PostgreSQL 11 – 所有可用版本
- RDS for PostgreSQL 10 – 所有可用版本

支持 Amazon RDS 中 Kerberos 身份验证功能的区域和数据库引擎

通过在 Amazon RDS 中使用 Kerberos 身份验证，可以支持使用 Kerberos 和 Microsoft Active Directory 对数据库用户进行外部身份验证。使用 Kerberos 和 Active Directory 为数据库用户提供了单点登录和集中身份验证的优势。

Kerberos 身份验证不可用于以下引擎：

- RDS for MariaDB

尽管大多数 Amazon 区域默认情况下对于您的 Amazon 账户处于活动状态，但对于某些区域，仅当您手动选择它们时才会激活。这些区域称为选择加入区域。相比之下，一旦创建了您的 Amazon 账户，默认情况下处于活动状态的区域就称为商业区域，或者简称为区域。对于选择加入区域，您必须使用格式为 `directoryservice.rds.`*`region_name`*`.amazonaws.com` 的区域化服务主体。例如，对于非洲（开普敦），您必须将服务主体 `directoryservice.rds.af-south-1.amazonaws.com` 添加到您的信任策略中。有关更多信息，请参阅 Kerberos 身份验证。

主题

- 适用于 RDS for Db2 的 Kerberos 身份验证
- 适用于 RDS for MySQL 的 Kerberos 身份验证
- 适用于 RDS for Oracle 的 Kerberos 身份验证
- 适用于 RDS for PostgreSQL 的 Kerberos 身份验证
- 适用于 RDS for SQL Server 的 Kerberos 身份验证

适用于 RDS for Db2 的 Kerberos 身份验证

以下区域和引擎版本可用于面向 RDS for Db2 的 Kerberos 身份验证。

区域	RDS for Db2 11.5
美国东部（弗吉尼亚州北部）	所有版本
美国东部（俄亥俄州）	所有版本
美国西部（加利福尼亚北部）	所有版本
美国西部（俄勒冈州）	所有版本
非洲（开普敦）	所有版本
亚太地区（香港）	不可用
亚太地区（海得拉巴）	所有版本
亚太地区（雅加达）	所有版本
亚太地区（马来西亚）	不可用
亚太地区（墨尔本）	所有版本
亚太地区（孟买）	所有版本
亚太地区（大阪）	不可用
亚太地区（首尔）	所有版本
亚太地区（新加坡）	所有版本
亚太地区（悉尼）	所有版本
亚太地区（东京）	所有版本
加拿大（中部）	所有版本
加拿大西部（卡尔加里）	不可用
中国（北京）	不可用
中国（宁夏）	不可用

区域	RDS for Db2 11.5
欧洲地区（法兰克福）	所有版本
欧洲地区（爱尔兰）	所有版本
欧洲地区（伦敦）	所有版本
欧洲地区（米兰）	所有版本
欧洲地区（巴黎）	不可用
欧洲地区（西班牙）	所有版本
欧洲地区（斯德哥尔摩）	所有版本
欧洲（苏黎世）	所有版本
以色列（特拉维夫）	所有版本
中东（巴林）	所有版本
中东（阿联酋）	所有版本
南美洲（圣保罗）	所有版本
Amazon GovCloud（美国东部）	不可用
Amazon GovCloud（美国西部）	不可用

适用于 RDS for MySQL 的 Kerberos 身份验证

以下区域和引擎版本可用于面向 RDS for MySQL 的 Kerberos 身份验证。

区域	RDS for MySQL 8.4	RDS for MySQL 8.0	RDS for MySQL 5.7（在 RDS 扩展支持下）
美国东部（弗吉尼亚州北部）	所有版本	所有版本	所有版本

区域	RDS for MySQL 8.4	RDS for MySQL 8.0	RDS for MySQL 5.7（在 RDS 扩展支持下）
美国东部（俄亥俄州）	所有版本	所有版本	所有版本
美国西部（加利福尼亚北部）	所有版本	所有版本	所有版本
美国西部（俄勒冈州）	所有版本	所有版本	所有版本
非洲（开普敦）	所有版本	所有版本	所有版本
亚太地区（香港）	所有版本	所有版本	所有版本
亚太地区（海得拉巴）	所有版本	所有版本	所有版本
亚太地区（雅加达）	所有版本	所有版本	所有版本
亚太地区（马来西亚）	不可用	不可用	不可用
亚太地区（墨尔本）	所有版本	所有版本	所有版本
亚太地区（孟买）	所有版本	所有版本	所有版本
亚太地区（大阪）	所有版本	所有版本	所有版本
亚太地区（首尔）	所有版本	所有版本	所有版本
亚太地区（新加坡）	所有版本	所有版本	所有版本
亚太地区（悉尼）	所有版本	所有版本	所有版本
亚太地区（东京）	所有版本	所有版本	所有版本
加拿大（中部）	所有版本	所有版本	所有版本

区域	RDS for MySQL 8.4	RDS for MySQL 8.0	RDS for MySQL 5.7（在 RDS 扩展支持下）
加拿大西部（卡尔加里）	不可用	不可用	不可用
中国（北京）	所有版本	所有版本	所有版本
中国（宁夏）	所有版本	所有版本	所有版本
欧洲地区（法兰克福）	所有版本	所有版本	所有版本
欧洲地区（爱尔兰）	所有版本	所有版本	所有版本
欧洲地区（伦敦）	所有版本	所有版本	所有版本
欧洲地区（米兰）	所有版本	所有版本	所有版本
欧洲地区（巴黎）	所有版本	所有版本	所有版本
欧洲地区（西班牙）	所有版本	所有版本	所有版本
欧洲地区（斯德哥尔摩）	所有版本	所有版本	所有版本
欧洲（苏黎世）	所有版本	所有版本	所有版本
以色列（特拉维夫）	所有版本	所有版本	所有版本
中东（巴林）	所有版本	所有版本	所有版本
中东（阿联酋）	所有版本	所有版本	所有版本
南美洲（圣保罗）	所有版本	所有版本	所有版本
Amazon GovCloud（美国东部）	不可用	不可用	不可用

区域	RDS for MySQL 8.4	RDS for MySQL 8.0	RDS for MySQL 5.7（在 RDS 扩展支持下）
Amazon GovCloud (美国西部）	不可用	不可用	不可用

适用于 RDS for Oracle 的 Kerberos 身份验证

以下区域和引擎版本可用于面向 RDS for Oracle 的 Kerberos 身份验证。

区域	RDS for Oracle 21c	RDS for Oracle 19c
美国东部（弗吉尼亚州北部）	所有版本	所有版本
美国东部（俄亥俄州）	所有版本	所有版本
美国西部（加利福尼亚北部）	所有版本	所有版本
美国西部（俄勒冈州）	所有版本	所有版本
非洲（开普敦）（选择加入区域）	所有版本	所有版本
亚太地区（香港）（选择加入区域）	所有版本	所有版本
亚太地区（海得拉巴）（选择加入区域）	所有版本	所有版本
亚太地区（雅加达）（选择加入区域）	所有版本	所有版本
亚太地区（马来西亚）	不可用	不可用
亚太地区（墨尔本）（选择加入区域）	所有版本	所有版本
亚太地区（孟买）	所有版本	所有版本

区域	RDS for Oracle 21c	RDS for Oracle 19c
亚太地区（大阪）	不可用	不可用
亚太地区（首尔）	所有版本	所有版本
亚太地区（新加坡）	所有版本	所有版本
亚太地区（悉尼）	所有版本	所有版本
亚太地区（东京）	所有版本	所有版本
加拿大（中部）	所有版本	所有版本
加拿大西部（卡尔加里）	不可用	不可用
中国（北京）	不可用	不可用
中国（宁夏）	不可用	不可用
欧洲地区（法兰克福）	所有版本	所有版本
欧洲地区（爱尔兰）	所有版本	所有版本
欧洲地区（伦敦）	所有版本	所有版本
欧洲地区（米兰）（选择加入区域）	所有版本	所有版本
欧洲地区（巴黎）	不可用	不可用
欧洲（西班牙）（选择加入区域）	所有版本	所有版本
欧洲地区（斯德哥尔摩）	所有版本	所有版本
欧洲（苏黎世）（选择加入区域）	所有版本	所有版本
以色列（特拉维夫）（选择加入区域）	所有版本	所有版本

区域	RDS for Oracle 21c	RDS for Oracle 19c
中东（巴林）（选择加入区域）	所有版本	所有版本
中东（阿联酋）（选择加入区域）	所有版本	所有版本
南美洲（圣保罗）	所有版本	所有版本
Amazon GovCloud（美国东部）	所有版本	所有版本
Amazon GovCloud（美国西部）	所有版本	所有版本

适用于 RDS for PostgreSQL 的 Kerberos 身份验证

以下区域和引擎版本可用于面向 RDS for PostgreSQL 的 Kerberos 身份验证。

区域	RDS for PostgreSQL 17	RDS for PostgreSQL 16	RDS for PostgreSQL 15	RDS for PostgreSQL 14	RDS for PostgreSQL 13	RDS for PostgreSQL 12	RDS for PostgreSQL 11	RDS for PostgreSQL 10
美国东部（弗吉尼亚州北部）	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本
美国东部（俄亥俄州）	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本
美国西部（加利福尼	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本

区域	RDS for PostgreSQL 17	RDS for PostgreSQL 16	RDS for PostgreSQL 15	RDS for PostgreSQL 14	RDS for PostgreSQL 13	RDS for PostgreSQL 12	RDS for PostgreSQL 11	RDS for PostgreSQL 10
亚北部）								
美国西部（俄勒冈州）	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本
非洲（开普敦）	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本
亚太地区（香港）	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本
亚太地区（海得拉巴）	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本
亚太地区（雅加达）	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本
亚太地区（马来西亚）	不可用	不可用	不可用	不可用	不可用	不可用	不可用	不可用
亚太地区（墨尔本）	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本

区域	RDS for PostgreSQL 17	RDS for PostgreSQL 16	RDS for PostgreSQL 15	RDS for PostgreSQL 14	RDS for PostgreSQL 13	RDS for PostgreSQL 12	RDS for PostgreSQL 11	RDS for PostgreSQL 10
亚太地区（孟买）	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本
亚太地区（大阪）	不可用	不可用	不可用	不可用	不可用	不可用	不可用	不可用
亚太地区（首尔）	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本
亚太地区（新加坡）	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本
亚太地区（悉尼）	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本
亚太地区（东京）	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本
加拿大（中部）	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本
加拿大西部（卡尔加里）	不可用	不可用	不可用	不可用	不可用	不可用	不可用	不可用

区域	RDS for PostgreSQL 17	RDS for PostgreSQL 16	RDS for PostgreSQL 15	RDS for PostgreSQL 14	RDS for PostgreSQL 13	RDS for PostgreSQL 12	RDS for PostgreSQL 11	RDS for PostgreSQL 10
中国（北京）	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本
中国（宁夏）	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本
欧洲地区（法兰克福）	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本
欧洲地区（爱尔兰）	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本
欧洲地区（伦敦）	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本
欧洲地区（米兰）	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本
欧洲地区（巴黎）	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本
欧洲地区（西班牙）	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本

区域	RDS for PostgreSQL 17	RDS for PostgreSQL 16	RDS for PostgreSQL 15	RDS for PostgreSQL 14	RDS for PostgreSQL 13	RDS for PostgreSQL 12	RDS for PostgreSQL 11	RDS for PostgreSQL 10
欧洲地区（斯德哥尔摩）	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本
欧洲（苏黎世）	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本
以色列（特拉维夫）	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本
中东（巴林）	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本
中东（阿联酋）	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本
南美洲（圣保罗）	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本	所有版本
Amazon GovCloud美国东部）	不可用	不可用	不可用	不可用	不可用	不可用	不可用	不可用
Amazon GovCloud美国西部）	不可用	不可用	不可用	不可用	不可用	不可用	不可用	不可用

适用于 RDS for SQL Server 的 Kerberos 身份验证

以下区域和引擎版本可用于面向 RDS for SQL Server 的 Kerberos 身份验证。

区域	RDS for SQL Server 2022	RDS for SQL Server 2019	RDS for SQL Server 2017	RDS for SQL Server 2016
美国东部（弗吉尼亚州北部）	所有版本	所有版本	所有版本	所有版本
美国东部（俄亥俄州）	所有版本	所有版本	所有版本	所有版本
美国西部（加利福尼亚北部）	所有版本	所有版本	所有版本	所有版本
美国西部（俄勒冈州）	所有版本	所有版本	所有版本	所有版本
非洲（开普敦）	所有版本	所有版本	所有版本	所有版本
亚太地区（香港）	所有版本	所有版本	所有版本	所有版本
亚太地区（海得拉巴）	所有版本	所有版本	所有版本	所有版本
亚太地区（马来西亚）	不可用	不可用	不可用	不可用
亚太地区（墨尔本）	所有版本	所有版本	所有版本	所有版本
亚太地区（孟买）	所有版本	所有版本	所有版本	所有版本
亚太地区（大阪）	所有版本	所有版本	所有版本	所有版本

区域	RDS for SQL Server 2022	RDS for SQL Server 2019	RDS for SQL Server 2017	RDS for SQL Server 2016
亚太地区（首尔）	所有版本	所有版本	所有版本	所有版本
亚太地区（新加坡）	所有版本	所有版本	所有版本	所有版本
亚太地区（悉尼）	所有版本	所有版本	所有版本	所有版本
亚太地区（东京）	所有版本	所有版本	所有版本	所有版本
加拿大（中部）	所有版本	所有版本	所有版本	所有版本
加拿大西部（卡尔加里）	不可用	不可用	不可用	不可用
中国（北京）	所有版本	所有版本	所有版本	所有版本
中国（宁夏）	所有版本	所有版本	所有版本	所有版本
欧洲地区（法兰克福）	所有版本	所有版本	所有版本	所有版本
欧洲地区（爱尔兰）	所有版本	所有版本	所有版本	所有版本
欧洲地区（伦敦）	所有版本	所有版本	所有版本	所有版本
欧洲地区（米兰）	所有版本	所有版本	所有版本	所有版本
欧洲地区（巴黎）	所有版本	所有版本	所有版本	所有版本

区域	RDS for SQL Server 2022	RDS for SQL Server 2019	RDS for SQL Server 2017	RDS for SQL Server 2016
欧洲地区（西班牙）	所有版本	所有版本	所有版本	所有版本
欧洲地区（斯德哥尔摩）	所有版本	所有版本	所有版本	所有版本
欧洲（苏黎世）	所有版本	所有版本	所有版本	所有版本
以色列（特拉维夫）	不可用	不可用	不可用	不可用
中东（巴林）	所有版本	所有版本	所有版本	所有版本
中东（阿联酋）	所有版本	所有版本	所有版本	所有版本
南美洲（圣保罗）	所有版本	所有版本	所有版本	所有版本
Amazon GovCloud（美国东部）	所有版本	所有版本	所有版本	所有版本
Amazon GovCloud（美国西部）	所有版本	所有版本	所有版本	所有版本

支持 Amazon RDS 中多可用区数据库集群的区域和数据库引擎

Amazon RDS 中的多可用区数据库集群部署提供 Amazon RDS 的高可用性部署模式，具有两个可读的备用数据库实例。多可用区数据库集群在同一区域的三个独立可用区中有一个写入器数据库实例和两个读取器数据库实例。与多可用区数据库实例部署相比，多可用区数据库集群可提供高可用性、增加读取工作负载容量以及更低的写入延迟。有关更多信息，请参阅Amazon RDS 的多可用区数据库集群部署。

多可用区数据库集群不适用于以下引擎：

- RDS for Db2
- RDS for MariaDB
- RDS for Oracle
- RDS for SQL Server

主题

- 适用于 RDS for MySQL 的多可用区数据库集群
- 适用于 RDS for PostgreSQL 的多可用区数据库集群

适用于 RDS for MySQL 的多可用区数据库集群

以下区域和引擎版本可用于面向 RDS for MySQL 的多可用区数据库集群。

区域	RDS for MySQL 8.4	RDS for MySQL 8.0
美国东部（弗吉尼亚州北部）	所有可用版本	所有可用版本
美国东部（俄亥俄州）	所有可用版本	所有可用版本
美国西部（加利福尼亚北部）	不可用	不可用
美国西部（俄勒冈州）	所有可用版本	所有可用版本
非洲（开普敦）	所有可用版本	所有可用版本
亚太地区（香港）	所有可用版本	所有可用版本
亚太地区（海得拉巴）	所有可用版本	所有可用版本
亚太地区（雅加达）	所有可用版本	所有可用版本

区域	RDS for MySQL 8.4	RDS for MySQL 8.0
亚太地区（马来西亚）	所有可用版本	所有可用版本
亚太地区（墨尔本）	所有可用版本	所有可用版本
亚太地区（孟买）	所有可用版本	所有可用版本
亚太地区（大阪）	所有可用版本	所有可用版本
亚太地区（首尔）	所有可用版本	所有可用版本
亚太地区（新加坡）	所有可用版本	所有可用版本
亚太地区（悉尼）	所有可用版本	所有可用版本
亚太地区（东京)	所有可用版本	所有可用版本
加拿大（中部）	所有可用版本	所有可用版本
加拿大（中部）	所有可用版本	所有可用版本
加拿大西部（卡尔加里）	所有可用版本	所有可用版本
中国（北京）	所有可用版本	所有可用版本
中国（宁夏）	所有可用版本	所有可用版本
欧洲地区（法兰克福）	所有可用版本	所有可用版本
欧洲地区（爱尔兰）	所有可用版本	所有可用版本
欧洲地区（伦敦）	所有可用版本	所有可用版本
欧洲地区（米兰）	所有可用版本	所有可用版本
欧洲地区（巴黎）	所有可用版本	所有可用版本
欧洲地区（西班牙）	所有可用版本	所有可用版本

区域	RDS for MySQL 8.4	RDS for MySQL 8.0
欧洲地区（斯德哥尔摩）	所有可用版本	所有可用版本
欧洲（苏黎世）	所有可用版本	所有可用版本
以色列（特拉维夫）	所有可用版本	所有可用版本
中东（巴林）	所有可用版本	所有可用版本
中东（阿联酋）	所有可用版本	所有可用版本
南美洲（圣保罗）	所有可用版本	所有可用版本
Amazon GovCloud (美国东部）	不可用	不可用
Amazon GovCloud (美国西部）	不可用	不可用

您可以使用 Amazon CLI 列出区域中可用于给定数据库实例类的版本。更改数据库实例类可显示可用于该实例类的引擎版本。

对于 Linux、macOS 或 Unix：

```
aws rds describe-orderable-db-instance-options \
--engine mysql \
--db-instance-class db.r5d.large \
--query '*[]|[?SupportsClusters == `true`].[EngineVersion]'  \
--output text
```

对于 Windows：

```
aws rds describe-orderable-db-instance-options ^
--engine mysql ^
--db-instance-class db.r5d.large ^
--query "*[]|[?SupportsClusters == `true`].[EngineVersion]"  ^
--output text
```

适用于 RDS for PostgreSQL 的多可用区数据库集群

以下区域和引擎版本可用于面向 RDS for PostgreSQL 的多可用区数据库集群。

区域	RDS for PostgreSQL 17	RDS for PostgreSQL 16	RDS for PostgreSQL 15	RDS for PostgreSQL 14	RDS for PostgreSQL 13
美国东部（弗吉尼亚州北部）	所有 PostgreSQL 17 版本	所有 PostgreSQL 16 版本	所有 PostgreSQL 15 版本	版本 14.5 及更高版本	版本 13.4 和版本 13.7 及更高版本
美国东部（俄亥俄州）	所有 PostgreSQL 17 版本	所有 PostgreSQL 16 版本	所有 PostgreSQL 15 版本	版本 14.5 及更高版本	版本 13.4 和版本 13.7 及更高版本
美国西部（加利福尼亚北部）	不可用	不可用	不可用	不可用	不可用
美国西部（俄勒冈州）	所有 PostgreSQL 17 版本	所有 PostgreSQL 16 版本	所有 PostgreSQL 15 版本	版本 14.5 及更高版本	版本 13.4 和版本 13.7 及更高版本
非洲（开普敦）	所有 PostgreSQL 17 版本	所有 PostgreSQL 16 版本	所有 PostgreSQL 15 版本	版本 14.5 及更高版本	版本 13.4 和版本 13.7 及更高版本
亚太地区（香港）	所有 PostgreSQL 17 版本	所有 PostgreSQL 16 版本	所有 PostgreSQL 15 版本	版本 14.5 及更高版本	版本 13.4 和版本 13.7 及更高版本
亚太地区（海得拉巴）	所有 PostgreSQL 17 版本	所有 PostgreSQL 16 版本	所有 PostgreSQL 15 版本	版本 14.5 及更高版本	版本 13.4 和版本 13.7 及更高版本
亚太地区（雅加达）	所有 PostgreSQL 17 版本	所有 PostgreSQL 16 版本	所有 PostgreSQL 15 版本	版本 14.5 及更高版本	版本 13.4 和版本 13.7 及更高版本

区域	RDS for PostgreSQL 17	RDS for PostgreSQL 16	RDS for PostgreSQL 15	RDS for PostgreSQL 14	RDS for PostgreSQL 13
亚太地区（马来西亚）	所有 PostgreSQL 17 版本	所有 PostgreSQL 16 版本	所有 PostgreSQL 15 版本	版本 14.5 及更高版本	版本 13.4 和版本 13.7 及更高版本
亚太地区（墨尔本）	所有 PostgreSQL 17 版本	所有 PostgreSQL 16 版本	所有 PostgreSQL 15 版本	版本 14.5 及更高版本	版本 13.4 和版本 13.7 及更高版本
亚太地区（孟买）	所有 PostgreSQL 17 版本	所有 PostgreSQL 16 版本	所有 PostgreSQL 15 版本	版本 14.5 及更高版本	版本 13.4 和版本 13.7 及更高版本
亚太地区（大阪）	所有 PostgreSQL 17 版本	所有 PostgreSQL 16 版本	所有 PostgreSQL 15 版本	版本 14.5 及更高版本	版本 13.4 和版本 13.7 及更高版本
亚太地区（首尔）	所有 PostgreSQL 17 版本	所有 PostgreSQL 16 版本	所有 PostgreSQL 15 版本	版本 14.5 及更高版本	版本 13.4 和版本 13.7 及更高版本
亚太地区（新加坡）	所有 PostgreSQL 17 版本	所有 PostgreSQL 16 版本	所有 PostgreSQL 15 版本	版本 14.5 及更高版本	版本 13.4 和版本 13.7 及更高版本
亚太地区（悉尼）	所有 PostgreSQL 17 版本	所有 PostgreSQL 16 版本	所有 PostgreSQL 15 版本	版本 14.5 及更高版本	版本 13.4 和版本 13.7 及更高版本
亚太地区（东京）	所有 PostgreSQL 17 版本	所有 PostgreSQL 16 版本	所有 PostgreSQL 15 版本	版本 14.5 及更高版本	版本 13.4 和版本 13.7 及更高版本
加拿大（中部）	所有 PostgreSQL 17 版本	所有 PostgreSQL 16 版本	所有 PostgreSQL 15 版本	版本 14.5 及更高版本	版本 13.4 和版本 13.7 及更高版本

区域	RDS for PostgreSQL 17	RDS for PostgreSQL 16	RDS for PostgreSQL 15	RDS for PostgreSQL 14	RDS for PostgreSQL 13
加拿大西部（卡尔加里）	所有 PostgreSQL 17 版本	所有 PostgreSQL 16 版本	所有 PostgreSQL 15 版本	版本 14.5 及更高版本	版本 13.4 和版本 13.7 及更高版本
中国（北京）	所有 PostgreSQL 17 版本	所有 PostgreSQL 16 版本	所有 PostgreSQL 15 版本	版本 14.5 及更高版本	版本 13.4 和版本 13.7 及更高版本
中国（宁夏）	所有 PostgreSQL 17 版本	所有 PostgreSQL 16 版本	所有 PostgreSQL 15 版本	版本 14.5 及更高版本	版本 13.4 和版本 13.7 及更高版本
欧洲地区（法兰克福）	所有 PostgreSQL 17 版本	所有 PostgreSQL 16 版本	所有 PostgreSQL 15 版本	版本 14.5 及更高版本	版本 13.4 和版本 13.7 及更高版本
欧洲地区（爱尔兰）	所有 PostgreSQL 17 版本	所有 PostgreSQL 16 版本	所有 PostgreSQL 15 版本	版本 14.5 及更高版本	版本 13.4 和版本 13.7 及更高版本
欧洲地区（伦敦）	所有 PostgreSQL 17 版本	所有 PostgreSQL 16 版本	所有 PostgreSQL 15 版本	版本 14.5 及更高版本	版本 13.4 和版本 13.7 及更高版本
欧洲地区（米兰）	所有 PostgreSQL 17 版本	所有 PostgreSQL 16 版本	所有 PostgreSQL 15 版本	版本 14.5 及更高版本	版本 13.4 和版本 13.7 及更高版本
欧洲地区（巴黎）	所有 PostgreSQL 17 版本	所有 PostgreSQL 16 版本	所有 PostgreSQL 15 版本	版本 14.5 及更高版本	版本 13.4 和版本 13.7 及更高版本
欧洲地区（西班牙）	所有 PostgreSQL 17 版本	所有 PostgreSQL 16 版本	所有 PostgreSQL 15 版本	版本 14.5 及更高版本	版本 13.4 和版本 13.7 及更高版本

区域	RDS for PostgreSQL 17	RDS for PostgreSQL 16	RDS for PostgreSQL 15	RDS for PostgreSQL 14	RDS for PostgreSQL 13
欧洲地区（斯德哥尔摩）	所有 PostgreSQL 17 版本	所有 PostgreSQL 16 版本	所有 PostgreSQL 15 版本	版本 14.5 及更高版本	版本 13.4 和版本 13.7 及更高版本
欧洲（苏黎世）	所有 PostgreSQL 17 版本	所有 PostgreSQL 16 版本	所有 PostgreSQL 15 版本	版本 14.5 及更高版本	版本 13.4 和版本 13.7 及更高版本
以色列（特拉维夫）	所有 PostgreSQL 17 版本	所有 PostgreSQL 16 版本	所有 PostgreSQL 15 版本	版本 14.5 及更高版本	版本 13.4 和版本 13.7 及更高版本
中东（巴林）	所有 PostgreSQL 17 版本	所有 PostgreSQL 16 版本	所有 PostgreSQL 15 版本	版本 14.5 及更高版本	版本 13.4 和版本 13.7 及更高版本
中东（阿联酋）	所有 PostgreSQL 17 版本	所有 PostgreSQL 16 版本	所有 PostgreSQL 15 版本	版本 14.5 及更高版本	版本 13.4 和版本 13.7 及更高版本
南美洲（圣保罗）	所有 PostgreSQL 17 版本	所有 PostgreSQL 16 版本	所有 PostgreSQL 15 版本	版本 14.5 及更高版本	版本 13.4 和版本 13.7 及更高版本
Amazon GovCloud（美国东部）	不可用	不可用	不可用	不可用	不可用
Amazon GovCloud（美国西部）	不可用	不可用	不可用	不可用	不可用

您可以使用 Amazon CLI 列出区域中可用于给定数据库实例类的版本。更改数据库实例类可显示可用于该实例类的引擎版本。

对于 Linux、macOS 或 Unix：

```
aws rds describe-orderable-db-instance-options \
--engine postgres \
--db-instance-class db.r5d.large \
--query '*[]|[?SupportsClusters == `true`].[EngineVersion]'  \
--output text
```

对于 Windows：

```
aws rds describe-orderable-db-instance-options ^
--engine postgres ^
--db-instance-class db.r5d.large ^
--query "*[]|[?SupportsClusters == `true`].[EngineVersion]"  ^
--output text
```

支持 Amazon RDS 中 Performance Insights 的区域和数据库引擎

> **Important**
>
> Amazon 已宣布性能详情的生命周期终止日期为：2025 年 11 月 30 日。在此日期之后，Amazon RDS 将不再支持性能详情控制台体验、灵活的保留期（1-24 个月）及其关联定价。性能详情 API 将继续存在，定价不变。性能详情 API 的费用将与 CloudWatch 数据库洞察的费用一起显示在您的 Amazon 账单中。
>
> 我们建议您在 2025 年 11 月 30 日之前，将使用性能详情付费套餐的所有数据库实例升级到数据库洞察的高级模式。有关升级到数据库洞察的高级模式的信息，请参阅开启适用于 Amazon RDS 的数据库洞察的高级模式。
>
> 如果您不采取任何行动，则使用性能详情的数据库实例将默认为使用数据库洞察的标准模式。在数据库洞察的标准模式下，您可能会在 7 天后无法访问性能数据历史记录，并且可能无法在 Amazon RDS 控制台中使用执行计划和按需分析功能。在 2025 年 11 月 30 日之后，只有数据库洞察的高级模式将支持执行计划和按需分析。
>
> 借助 CloudWatch 数据库洞察，您可以监控数据库实例集的数据库负载，并对性能进行大规模分析和故障排除。有关数据库洞察的更多信息，请参阅使用 CloudWatch 数据库洞察监控 Amazon RDS 数据库。有关定价信息，请参阅 Amazon CloudWatch 定价。

Amazon RDS 中的 Performance Insights 在现有 Amazon RDS 监控功能的基础上进行了扩展，以便展示并帮助您分析数据库性能。利用 Performance Insights 控制面板，您可以直观呈现 Amazon RDS

数据库实例上的数据库负载。您还可以按等待状态、SQL 语句、主机或用户来筛选负载。有关更多信息，请参阅 在 Amazon RDS 上使用性能详情监控数据库负载。

Performance Insights 适用于所有 RDS 数据库引擎，RDS for Db2 除外。

对于可用的数据库引擎，Performance Insights 适用于所有可用的引擎版本和所有 Amazon Web Services 区域。

有关 Performance Insights 功能的区域、数据库引擎和实例类支持的信息，请参阅支持性能详情功能的 Amazon RDS 数据库引擎、区域和实例类。

支持 RDS Custom 的区域和数据库引擎

Amazon RDS Custom 自动执行数据库管理任务和操作。通过使用 RDS Custom，您能够作为数据库管理员访问和自定义数据库环境和操作系统。使用 RDS Custom，您可以进行自定义以满足传统、自定义和打包应用程序的要求。有关更多信息，请参阅 Amazon RDS Custom。

仅以下数据库引擎支持 RDS Custom：

主题

- 支持 RDS Custom for Oracle 的区域和数据库引擎
- 支持 RDS Custom for SQL Server 的区域和数据库引擎

支持 RDS Custom for Oracle 的区域和数据库引擎

以下区域和引擎版本可用于 RDS Custom for Oracle。

区域	Oracle Database 19c	Oracle Database 18c	Oracle Database 12c
美国东部（弗吉尼亚北部）	19c，含有 2021 年 1 月或更高版本的 RU/RUR	18c，含有 2021 年 1 月或更高版本的 RU/RUR	12.1 和 12.2，含有 2021 年 1 月或更高版本的 RU/RUR
美国东部（俄亥俄州）	19c，含有 2021 年 1 月或更高版本的 RU/RUR	18c，含有 2021 年 1 月或更高版本的 RU/RUR	12.1 和 12.2，含有 2021 年 1 月或更高版本的 RU/RUR
美国西部（加利福尼亚北部）	不可用	不可用	不可用

区域	Oracle Database 19c	Oracle Database 18c	Oracle Database 12c
美国西部（俄勒冈州）	19c，含有 2021 年 1 月或更高版本的 RU/RUR	18c，含有 2021 年 1 月或更高版本的 RU/RUR	12.1 和 12.2，含有 2021 年 1 月或更高版本的 RU/RUR
非洲（开普敦）	不可用	不可用	不可用
亚太地区（香港）	不可用	不可用	不可用
亚太地区（雅加达）	19c，含有 2021 年 1 月或更高版本的 RU/RUR	18c，含有 2021 年 1 月或更高版本的 RU/RUR	12.1 和 12.2，含有 2021 年 1 月或更高版本的 RU/RUR
亚太地区（马来西亚）	不可用	不可用	不可用
亚太地区（墨尔本）	不可用	不可用	不可用
亚太地区（孟买）	19c，含有 2021 年 1 月或更高版本的 RU/RUR	18c，含有 2021 年 1 月或更高版本的 RU/RUR	12.1 和 12.2，含有 2021 年 1 月或更高版本的 RU/RUR
亚太地区（大阪）	19c，含有 2021 年 1 月或更高版本的 RU/RUR	18c，含有 2021 年 1 月或更高版本的 RU/RUR	12.1 和 12.2，含有 2021 年 1 月或更高版本的 RU/RUR
亚太地区（首尔）	19c，含有 2021 年 1 月或更高版本的 RU/RUR	18c，含有 2021 年 1 月或更高版本的 RU/RUR	12.1 和 12.2，含有 2021 年 1 月或更高版本的 RU/RUR
亚太地区（新加坡）	19c，含有 2021 年 1 月或更高版本的 RU/RUR	18c，含有 2021 年 1 月或更高版本的 RU/RUR	12.1 和 12.2，含有 2021 年 1 月或更高版本的 RU/RUR
亚太地区（悉尼）	19c，含有 2021 年 1 月或更高版本的 RU/RUR	18c，含有 2021 年 1 月或更高版本的 RU/RUR	12.1 和 12.2，含有 2021 年 1 月或更高版本的 RU/RUR

区域	Oracle Database 19c	Oracle Database 18c	Oracle Database 12c
亚太地区（东京）	19c，含有 2021 年 1 月或更高版本的 RU/RUR	18c，含有 2021 年 1 月或更高版本的 RU/RUR	12.1 和 12.2，含有 2021 年 1 月或更高版本的 RU/RUR
加拿大（中部）	19c，含有 2021 年 1 月或更高版本的 RU/RUR	18c，含有 2021 年 1 月或更高版本的 RU/RUR	12.1 和 12.2，含有 2021 年 1 月或更高版本的 RU/RUR
加拿大西部（卡尔加里）	不可用	不可用	不可用
中国（北京）	不可用	不可用	不可用
中国（宁夏）	不可用	不可用	不可用
欧洲地区（法兰克福）	19c，含有 2021 年 1 月或更高版本的 RU/RUR	18c，含有 2021 年 1 月或更高版本的 RU/RUR	12.1 和 12.2，含有 2021 年 1 月或更高版本的 RU/RUR
欧洲地区（爱尔兰）	19c，含有 2021 年 1 月或更高版本的 RU/RUR	18c，含有 2021 年 1 月或更高版本的 RU/RUR	12.1 和 12.2，含有 2021 年 1 月或更高版本的 RU/RUR
欧洲地区（伦敦）	19c，含有 2021 年 1 月或更高版本的 RU/RUR	18c，含有 2021 年 1 月或更高版本的 RU/RUR	12.1 和 12.2，含有 2021 年 1 月或更高版本的 RU/RUR
欧洲地区（米兰）	19c，含有 2021 年 1 月或更高版本的 RU/RUR	18c，含有 2021 年 1 月或更高版本的 RU/RUR	12.1 和 12.2，含有 2021 年 1 月或更高版本的 RU/RUR
欧洲地区（巴黎）	19c，含有 2021 年 1 月或更高版本的 RU/RUR	18c，含有 2021 年 1 月或更高版本的 RU/RUR	12.1 和 12.2，含有 2021 年 1 月或更高版本的 RU/RUR

区域	Oracle Database 19c	Oracle Database 18c	Oracle Database 12c
欧洲地区（斯德哥尔摩）	19c，含有 2021 年 1 月或更高版本的 RU/RUR	18c，含有 2021 年 1 月或更高版本的 RU/RUR	12.1 和 12.2，含有 2021 年 1 月或更高版本的 RU/RUR
以色列（特拉维夫）	不可用	不可用	不可用
中东（巴林）	不可用	不可用	不可用
中东（阿联酋）	19c，含有 2021 年 1 月或更高版本的 RU/RUR	18c，含有 2021 年 1 月或更高版本的 RU/RUR	12.1 和 12.2，含有 2021 年 1 月或更高版本的 RU/RUR
南美洲（圣保罗）	19c，含有 2021 年 1 月或更高版本的 RU/RUR	18c，含有 2021 年 1 月或更高版本的 RU/RUR	12.1 和 12.2，含有 2021 年 1 月或更高版本的 RU/RUR
Amazon GovCloud（美国东部）	19c，含有 2021 年 1 月或更高版本的 RU/RUR	18c，含有 2021 年 1 月或更高版本的 RU/RUR	12.1 和 12.2，含有 2021 年 1 月或更高版本的 RU/RUR
Amazon GovCloud（美国西部）	19c，含有 2021 年 1 月或更高版本的 RU/RUR	18c，含有 2021 年 1 月或更高版本的 RU/RUR	12.1 和 12.2，含有 2021 年 1 月或更高版本的 RU/RUR

支持 RDS Custom for SQL Server 的区域和数据库引擎

您可以使用 RDS 提供的引擎版本（RPEV）或自定义引擎版本（CEV）来部署 RDS Custom for SQL Server：

- 如果您使用 RPEV，则它包括默认的亚马逊机器映像（AMI）和 SQL Server 安装。如果您自定义或修改操作系统（OS），则在修补、快照还原或自动恢复期间，您的更改可能不会持续存在。
- 如果您使用 CEV，您可以选择自己的 AMI，其中包括预安装的 Microsoft SQL Server 或使用自己的媒体安装的 SQL Server。使用 Amazon 提供的 CEV 时，您可以选择 Amazon 提供的最新 Amazon EC2 镜像（AMI），该镜像具有 RDS Custom for SQL Server 支持的累积更新（CU）。使用 CEV，您可以自定义操作系统和 SQL Server 配置以满足您的企业需求。

以下 Amazon Web Services 区域和数据库引擎版本可用于 RDS Custom for SQL Server。引擎版本支持取决于您使用的是带有 RPEV 的 RDS Custom for SQL Server、Amazon 提供的 CEV 还是客户提供的 CEV。

区域	PREV	Amazon 提供的 CEV	客户提供的 CEV
美国东部（弗吉尼亚州北部）	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。
美国东部（俄亥俄州）	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。

区域	PREV	Amazon 提供的 CEV	客户提供的 CEV
美国西部（加利福尼亚北部）	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。
美国西部（俄勒冈州）	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。
非洲（开普敦）	不可用	不可用	不可用
亚太地区（香港）	不可用	不可用	不可用

区域	PREV	Amazon 提供的 CEV	客户提供的 CEV
亚太地区（海得拉巴）	不可用	不可用	不可用
亚太地区（雅加达）	不可用	不可用	不可用
亚太地区（马来西亚）	不可用	不可用	不可用
亚太地区（墨尔本）	不可用	不可用	不可用
亚太地区（孟买）	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。

区域	PREV	Amazon 提供的 CEV	客户提供的 CEV
亚太地区（大阪）	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。
亚太地区（首尔）	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。

区域	PREV	Amazon 提供的 CEV	客户提供的 CEV
亚太地区（新加坡）	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。 带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。 带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。 带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。
亚太地区（悉尼）	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。 带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。 带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。 带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。

区域	PREV	Amazon 提供的 CEV	客户提供的 CEV
亚太地区（东京）	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。
加拿大（中部）	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。
加拿大西部（卡尔加里）	不可用	不可用	不可用

区域	PREV	Amazon 提供的 CEV	客户提供的 CEV
中国（北京）	不可用	不可用	不可用
中国（宁夏）	不可用	不可用	不可用
欧洲地区（法兰克福）	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。
欧洲地区（爱尔兰）	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。

区域	PREV	Amazon 提供的 CEV	客户提供的 CEV
欧洲地区（伦敦）	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。
欧洲地区（米兰）	不可用	不可用	不可用
欧洲地区（巴黎）	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。

区域	PREV	Amazon 提供的 CEV	客户提供的 CEV
欧洲地区（西班牙）	不可用	不可用	不可用
欧洲地区（斯德哥尔摩）	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。
欧洲（苏黎世）	不可用	不可用	不可用
以色列（特拉维夫）	不可用	不可用	不可用
中东（巴林）	不可用	不可用	不可用
中东（阿联酋）	不可用	不可用	不可用

区域	PREV	Amazon 提供的 CEV	客户提供的 CEV
南美洲（圣保罗）	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。	带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。
Amazon GovCloud（美国东部）	不可用	不可用	不可用
Amazon GovCloud（美国西部）	不可用	不可用	不可用

支持 Amazon RDS 代理的区域和数据库引擎

Amazon RDS 代理是一种完全托管的高可用性数据库代理，它通过汇集和共享已建立的数据库连接，提高应用程序的可扩展性。有关更多信息，请参阅 的 Amazon RDS 代理。

RDS 代理不适用于以下引擎：

- RDS for Db2
- RDS for Oracle

主题

- 适用于 RDS for MariaDB 的 RDS 代理
- 适用于 RDS for MySQL 的 RDS 代理
- 适用于 RDS for PostgreSQL 的 RDS 代理
- 适用于 RDS for SQL Server 的 RDS 代理

适用于 RDS for MariaDB 的 RDS 代理

以下区域和引擎版本可用于面向 RDS for MariaDB 的 RDS 代理。

区域	RDS for MariaDB 11.4	RDS for MariaDB 10.11	RDS for MariaDB 10.6	RDS for MariaDB 10.5	RDS for MariaDB 10.4
美国东部（弗吉尼亚州北部）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
美国东部（俄亥俄州）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
美国西部（加利福尼亚北部）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
美国西部（俄勒冈州）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
非洲（开普敦）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（香港）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（海得拉巴）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（雅加达）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本

区域	RDS for MariaDB 11.4	RDS for MariaDB 10.11	RDS for MariaDB 10.6	RDS for MariaDB 10.5	RDS for MariaDB 10.4
亚太地区（马来西亚）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（墨尔本）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（孟买）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（大阪）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（首尔）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（新加坡）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（悉尼）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（泰国）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（东京)	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
加拿大（中部）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
加拿大西部（卡尔加里）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
中国（北京）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
中国（宁夏）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本

区域	RDS for MariaDB 11.4	RDS for MariaDB 10.11	RDS for MariaDB 10.6	RDS for MariaDB 10.5	RDS for MariaDB 10.4
欧洲地区（法兰克福）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（爱尔兰）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（伦敦）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（米兰）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（巴黎）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（西班牙）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（斯德哥尔摩）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲（苏黎世）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
以色列（特拉维夫）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
中东（巴林）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
中东（阿联酋）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
墨西哥（中部）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本

区域	RDS for MariaDB 11.4	RDS for MariaDB 10.11	RDS for MariaDB 10.6	RDS for MariaDB 10.5	RDS for MariaDB 10.4
南美洲（圣保罗）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
Amazon GovCloud（美国东部）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
Amazon GovCloud（美国西部）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本

适用于 RDS for MySQL 的 RDS 代理

以下区域和引擎版本可用于面向 RDS for MySQL 的 RDS 代理。

区域	RDS for MySQL 8.4	RDS for MySQL 8.0	RDS for MySQL 5.7
美国东部（弗吉尼亚州北部）	所有可用版本	所有可用版本	所有可用版本
美国东部（俄亥俄州）	所有可用版本	所有可用版本	所有可用版本
美国西部（加利福尼亚北部）	所有可用版本	所有可用版本	所有可用版本
美国西部（俄勒冈州）	所有可用版本	所有可用版本	所有可用版本
非洲（开普敦）	所有可用版本	所有可用版本	所有可用版本
亚太地区（香港）	所有可用版本	所有可用版本	所有可用版本

区域	RDS for MySQL 8.4	RDS for MySQL 8.0	RDS for MySQL 5.7
亚太地区（海得拉巴）	所有可用版本	所有可用版本	所有可用版本
亚太地区（雅加达）	所有可用版本	所有可用版本	所有可用版本
亚太地区（马来西亚）	所有可用版本	所有可用版本	所有可用版本
亚太地区（墨尔本）	所有可用版本	所有可用版本	所有可用版本
亚太地区（孟买）	所有可用版本	所有可用版本	所有可用版本
亚太地区（大阪）	所有可用版本	所有可用版本	所有可用版本
亚太地区（首尔）	所有可用版本	所有可用版本	所有可用版本
亚太地区（新加坡）	所有可用版本	所有可用版本	所有可用版本
亚太地区（悉尼）	所有可用版本	所有可用版本	所有可用版本
亚太地区（泰国）	所有可用版本	所有可用版本	所有可用版本
亚太地区（东京)	所有可用版本	所有可用版本	所有可用版本
加拿大（中部）	所有可用版本	所有可用版本	所有可用版本
加拿大西部（卡尔加里）	所有可用版本	所有可用版本	所有可用版本
中国（北京）	所有可用版本	所有可用版本	所有可用版本
中国（宁夏）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（法兰克福）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（爱尔兰）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（伦敦）	所有可用版本	所有可用版本	所有可用版本

区域	RDS for MySQL 8.4	RDS for MySQL 8.0	RDS for MySQL 5.7
欧洲地区（米兰）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（巴黎）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（西班牙）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（斯德哥尔摩）	所有可用版本	所有可用版本	所有可用版本
欧洲（苏黎世）	所有可用版本	所有可用版本	所有可用版本
以色列（特拉维夫）	所有可用版本	所有可用版本	所有可用版本
中东（巴林）	所有可用版本	所有可用版本	所有可用版本
中东（阿联酋）	所有可用版本	所有可用版本	所有可用版本
墨西哥（中部）	所有可用版本	所有可用版本	所有可用版本
南美洲（圣保罗）	所有可用版本	所有可用版本	所有可用版本
Amazon GovCloud（美国东部）	所有可用版本	所有可用版本	所有可用版本
Amazon GovCloud（美国西部）	所有可用版本	所有可用版本	所有可用版本

适用于 RDS for PostgreSQL 的 RDS 代理

以下区域和引擎版本可用于面向 RDS for PostgreSQL 的 RDS 代理。

区域	RDS for PostgreSQL 17	RDS for PostgreSQL 16	RDS for PostgreSQL 15	RDS for PostgreSQL 14	RDS for PostgreSQL 13	RDS for PostgreSQL 12	RDS for PostgreSQL 11	RDS for PostgreSQL 10
美国东部（弗吉尼亚	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本

区域	RDS for PostgreSQL 17	RDS for PostgreSQL 16	RDS for PostgreSQL 15	RDS for PostgreSQL 14	RDS for PostgreSQL 13	RDS for PostgreSQL 12	RDS for PostgreSQL 11	RDS for PostgreSQL 10
州北部）								
美国东部（俄亥俄州）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
美国西部（加利福尼亚北部）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
美国西部（俄勒冈州）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
非洲（开普敦）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（香港）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（海得拉巴）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（雅加达）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本

区域	RDS for PostgreSQL 17	RDS for PostgreSQL 16	RDS for PostgreSQL 15	RDS for PostgreSQL 14	RDS for PostgreSQL 13	RDS for PostgreSQL 12	RDS for PostgreSQL 11	RDS for PostgreSQL 10
亚太地区（马来西亚）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（墨尔本）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（孟买）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（大阪）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（首尔）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（新加坡）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（悉尼）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
亚太地区（泰国）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本

区域	RDS for PostgreSQL 17	RDS for PostgreSQL 16	RDS for PostgreSQL 15	RDS for PostgreSQL 14	RDS for PostgreSQL 13	RDS for PostgreSQL 12	RDS for PostgreSQL 11	RDS for PostgreSQL 10
亚太地区（东京)	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
加拿大（中部）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
加拿大西部（卡尔加里）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
中国（北京）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
中国（宁夏）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（法兰克福）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（爱尔兰）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（伦敦）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本

区域	RDS for PostgreSQL 17	RDS for PostgreSQL 16	RDS for PostgreSQL 15	RDS for PostgreSQL 14	RDS for PostgreSQL 13	RDS for PostgreSQL 12	RDS for PostgreSQL 11	RDS for PostgreSQL 10
欧洲地区（米兰）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（巴黎）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（西班牙）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲地区（斯德哥尔摩）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
欧洲（苏黎世）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
以色列（特拉维夫）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
中东（巴林）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
中东（阿联酋）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本

区域	RDS for PostgreSQL 17	RDS for PostgreSQL 16	RDS for PostgreSQL 15	RDS for PostgreSQL 14	RDS for PostgreSQL 13	RDS for PostgreSQL 12	RDS for PostgreSQL 11	RDS for PostgreSQL 10
墨西哥（中部）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
南美洲（圣保罗）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
Amazon GovCloud 美国东部）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本
Amazon GovCloud 美国西部）	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本	所有可用版本

适用于 RDS for SQL Server 的 RDS 代理

以下区域和引擎版本可用于面向 RDS for SQL Server 的 RDS 代理。

区域	RDS for SQL Server 2019	RDS for SQL Server 2017	RDS for SQL Server 2016
美国东部（弗吉尼亚州北部）	所有可用版本	所有可用版本	所有可用版本
美国东部（俄亥俄州）	所有可用版本	所有可用版本	所有可用版本
美国西部（加利福尼亚北部）	所有可用版本	所有可用版本	所有可用版本

区域	RDS for SQL Server 2019	RDS for SQL Server 2017	RDS for SQL Server 2016
美国西部（俄勒冈州）	所有可用版本	所有可用版本	所有可用版本
非洲（开普敦）	所有可用版本	所有可用版本	所有可用版本
亚太地区（香港）	所有可用版本	所有可用版本	所有可用版本
亚太地区（海得拉巴）	所有可用版本	所有可用版本	所有可用版本
亚太地区（雅加达）	所有可用版本	所有可用版本	所有可用版本
亚太地区（马来西亚）	所有可用版本	所有可用版本	所有可用版本
亚太地区（墨尔本）	所有可用版本	所有可用版本	所有可用版本
亚太地区（孟买）	所有可用版本	所有可用版本	所有可用版本
亚太地区（大阪）	所有可用版本	所有可用版本	所有可用版本
亚太地区（首尔）	所有可用版本	所有可用版本	所有可用版本
亚太地区（新加坡）	所有可用版本	所有可用版本	所有可用版本
亚太地区（悉尼）	所有可用版本	所有可用版本	所有可用版本
亚太地区（泰国）	所有可用版本	所有可用版本	所有可用版本
亚太地区（东京)	所有可用版本	所有可用版本	所有可用版本
加拿大（中部）	所有可用版本	所有可用版本	所有可用版本
加拿大西部（卡尔加里）	所有可用版本	所有可用版本	所有可用版本
中国（北京）	所有可用版本	所有可用版本	所有可用版本

区域	RDS for SQL Server 2019	RDS for SQL Server 2017	RDS for SQL Server 2016
中国（宁夏）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（法兰克福）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（爱尔兰）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（伦敦）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（米兰）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（巴黎）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（西班牙）	所有可用版本	所有可用版本	所有可用版本
欧洲地区（斯德哥尔摩）	所有可用版本	所有可用版本	所有可用版本
欧洲（苏黎世）	所有可用版本	所有可用版本	所有可用版本
以色列（特拉维夫）	所有可用版本	所有可用版本	所有可用版本
中东（巴林）	所有可用版本	所有可用版本	所有可用版本
中东（阿联酋）	所有可用版本	所有可用版本	所有可用版本
墨西哥（中部）	所有可用版本	所有可用版本	所有可用版本
南美洲（圣保罗）	所有可用版本	所有可用版本	所有可用版本
Amazon GovCloud（美国东部）	所有可用版本	所有可用版本	所有可用版本
Amazon GovCloud（美国西部）	所有可用版本	所有可用版本	所有可用版本

支持将 Secrets Manager 与 Amazon RDS 进行集成的区域和数据库引擎

借助 Amazon Secrets Manager，您可以将代码中的硬编码凭证（包括数据库密码）替换为对 Secrets Manager 的 API 调用，从而以编程方式检索密钥。有关 Secrets Manager 的更多信息，请参阅《Amazon Secrets Manager 用户指南》https://docs.amazonaws.cn/secretsmanager/latest/userguide/。

您可以针对 Amazon RDS 数据库实例或多可用区数据库集群指定 Amazon RDS 在 Secrets Manager 中管理主用户密码。RDS 生成密码，将其存储在 Secrets Manager 中，然后定期轮换密码。有关更多信息，请参阅 使用 Amazon RDS 和 Amazon Secrets Manager 管理密码。

Secrets Manager 集成适用于所有 Amazon Web Services 区域。

支持与 Amazon Redshift 进行 Amazon RDS 零 ETL 集成的区域和数据库引擎

RDS 与 Amazon Redshift 的零 ETL 集成是一个完全托管式解决方案，用于在将事务数据写入 Amazon RDS 数据库实例后，在 Amazon Redshift 中提供这些数据。有关更多信息，请参阅 零 ETL 集成。

主题

- RDS for MySQL 的零 ETL 集成

RDS for MySQL 的零 ETL 集成

以下区域和引擎版本可用于 RDS for MySQL 与 Amazon Redshift 的零 ETL 集成。

区域	RDS for MySQL 8.4	RDS for MySQL 8.0
美国东部（弗吉尼亚州北部）	所有可用版本	所有可用版本
美国东部（俄亥俄州）	所有可用版本	所有可用版本
美国西部（加利福尼亚北部）	所有可用版本	所有可用版本

区域	RDS for MySQL 8.4	RDS for MySQL 8.0
美国西部（俄勒冈州）	所有可用版本	所有可用版本
非洲（开普敦）	所有可用版本	所有可用版本
亚太地区（香港）	所有可用版本	所有可用版本
亚太地区（海得拉巴）	不可用	不可用
亚太地区（雅加达）	不可用	不可用
亚太地区（马来西亚）	不可用	不可用
亚太地区（墨尔本）	不可用	不可用
亚太地区（孟买）	所有可用版本	所有可用版本
亚太地区（大阪）	所有可用版本	所有可用版本
亚太地区（首尔）	所有可用版本	所有可用版本
亚太地区（新加坡）	所有可用版本	所有可用版本
亚太地区（悉尼）	所有可用版本	所有可用版本
亚太地区（东京)	所有可用版本	所有可用版本

区域	RDS for MySQL 8.4	RDS for MySQL 8.0
加拿大（中部）	所有可用版本	所有可用版本
加拿大西部（卡尔加里）	不可用	不可用
中国（北京）	不可用	不可用
中国（宁夏）	不可用	不可用
欧洲地区（法兰克福）	所有可用版本	所有可用版本
欧洲地区（爱尔兰）	所有可用版本	所有可用版本
欧洲地区（伦敦）	所有可用版本	所有可用版本
欧洲地区（米兰）	所有可用版本	所有可用版本
欧洲地区（巴黎）	所有可用版本	所有可用版本
欧洲地区（西班牙）	不可用	不可用
欧洲地区（斯德哥尔摩）	所有可用版本	所有可用版本
欧洲（苏黎世）	不可用	不可用
以色列（特拉维夫）	不可用	不可用
中东（巴林）	所有可用版本	所有可用版本

区域	RDS for MySQL 8.4	RDS for MySQL 8.0
中东（阿联酋）	不可用	不可用
南美洲（圣保罗）	所有可用版本	所有可用版本
Amazon GovCloud（美国东部）	不可用	不可用
Amazon GovCloud（美国西部）	不可用	不可用

Amazon RDS 中的引擎原生功能

Amazon RDS 数据库引擎还支持许多最常见的引擎原生特性和功能。这些特性与本页列出的 Amazon RDS 原生特性不同。有些引擎原生特性可能仅提供有限的支持或受限制的权限。

有关引擎原生特性的更多信息，请参阅：

- Amazon RDS for Db2 功能
- Amazon RDS 上的 MariaDB 特征支持
- Amazon RDS 上的 MySQL 功能支持
- RDS for Oracle 的功能
- 使用 Amazon RDS for PostgreSQL 支持的 PostgreSQL 功能
- Amazon RDS 上的 Microsoft SQL Server 功能

Amazon RDS 的数据库实例计费

Amazon RDS 实例根据以下组件进行计费：

- 数据库实例小时数（按小时）– 根据数据库实例的数据库实例类（例如，db.t2.small 或 db.m4.large）。定价以每小时为单位列出，但账单向下计算至秒，并以十进制形式显示时间。RDS 使用量以 1 秒的增量进行计费，时长最少 10 分钟。有关更多信息，请参阅 数据库实例类。
- 存储（每月的 GB 数）– 您为数据库实例预配置的存储容量。如果您在计费期内扩展了预置的存储容量，则应按相应比例付费。有关更多信息，请参阅"Amazon RDS 数据库实例存储"。
- 输入/输出（I/O）请求（每 1 百万个请求）– 您在计费周期内仅针对 Amazon RDS 磁性存储发出的存储 I/O 请求总数。
- 预调配 IOPS（每月的 IOPS 数）– 预调配 IOPS 速率，无论使用的 IOPS 如何，都是指 Amazon RDS 预调配 IOPS（SSD）和通用（SSD）gp3 存储。EBS 卷的预调配存储以 1 秒的增量计费，时长最少 10 分钟。
- 备份存储（每月的 GiB 数）– 备份存储是指与自动数据库备份和拍摄的有效数据库快照相关联的存储。延长备份保留期或增加快照创建数量，将增加数据库所消耗的备份存储。按秒计费不适用于备份存储（按 GB/月计量）。

  有关更多信息，请参阅"备份、还原和导出数据"。
- 数据传输（GB 数）– 从 Internet 和其他Amazon区域传入和传出数据库实例的数据。有关有用的示例，请参阅 Amazon 博客文章 Exploring Data Transfer Costs for Amazon Managed Databases。

Amazon RDS 提供了以下让您根据需求优化成本的购买选项：

- On-Demand Instances（按需实例）– 按使用的数据库实例小时数付费。定价以每小时为单位列出，但账单向下计算至秒，并以十进制形式显示时间。RDS 使用量现在以 1 秒的增量进行计费，时长最少 10 分钟。
- Reserved instances（预留实例）– 将数据库实例预留一年或三年，进而获得比按需数据库实例定价高很多的折扣。通过预留实例用量，您可以在 1 个小时内启动、删除、开始或停止多个实例并获得所有实例的预留实例优势。

有关 Amazon RDS 定价信息，请参阅 Amazon RDS 定价页。

主题

- Amazon RDS 的按需数据库实例

- [Amazon RDS 的预留数据库实例]

Amazon RDS 的按需数据库实例

Amazon RDS 按需数据库实例根据数据库实例的类（例如，db.t3.small 或 db.m5.large）来计费。有关 Amazon RDS 定价信息，请参阅 Amazon RDS 产品页。

数据库实例的账单周期从该数据库实例可用时开始计算。定价以每小时为单位列出，但账单向下计算至秒，并以十进制形式显示时间。Amazon RDS 使用量以一秒的增量进行计费，时长最少 10 分钟。如果可计费配置发生更改，例如扩展计算或存储容量，则最低收取 10 分钟的费用。计费将一直继续到数据库实例终止，当您删除数据库实例或数据库实例出现故障时即终止。

如果不再希望您的数据库实例被收取费用，必须将其停止或删除，以免产生更多应计费数据库实例小时数。有关计费的数据库实例状态的更多信息，请参阅查看Amazon RDS 数据库实例状态。

已停止的数据库实例

当数据库实例停止时，您将为预配置存储付费，包括预置 IOPS。您还将为备份存储付费，包括在指定保留时间内用于手动快照和自动备份的存储。您无需为数据库实例小时数付费。

多可用区数据库实例

多可用区设置会在不同可用区中，自动预置和维护一个同步备用副本，用来增强数据的耐久性和可用性。由于更多的资源和更高的可用性，多可用区部署的价格要高于单可用区部署，而且由于备用实例和关联资源的增加，多可用区部署的成本可能约为两倍。

关于多可用区定价，请考虑以下重要细节：

- 计算成本：对于主实例和备用实例，均按照每数据库实例小时数来计费。
- 存储成本：对于主实例和备用实例，均按照所预置的每月 GB 数量收费。
- 数据传输费用：主实例和备用实例之间的复制已包含在费用中，但根据您的使用情况，可能会收取其他数据传输费用。

要根据您的具体应用场景和 Amazon Web Services 区域准确估算每月费用，您可以使用 Amazon 定价计算器。此工具让您可以输入配置详细信息，然后为您提供全面的成本明细。

> **Note**
> 定价可能随时可能会发生变化。有关最新的信息，请参阅 Amazon RDS 定价页面。

Amazon RDS 的预留数据库实例

通过使用预留数据库实例，您可以将数据库实例预留一年或三年。相比按需数据库实例定价，数据库预留实例可以提供大幅折扣。预留数据库实例不是物理实例，而是对账户中使用的特定按需数据库实例所应用的账单折扣。用于预留数据库实例的折扣与实例类型和 Amazon Web Services 区域相关联。

使用预留数据库实例的一般过程如下：首先，获取有关可用预留数据库实例产品的信息，然后购买预留数据库实例产品，最后获取有关您的现有预留数据库实例的信息。

有关购买预留数据库实例和查看预留数据库实例的账单的信息，请参阅以下各节。

- 购买 Amazon RDS 的预留数据库实例
- 查看 Amazon RDS 的预留数据库实例的账单

预留数据库实例概述

如果购买了 Amazon RDS 中的预留数据库实例，将承诺在预留数据库实例的持续时间内为您提供特定数据库实例类型的折扣费率。要使用 Amazon RDS 预留数据库实例，应创建新的数据库实例，就像您为按需实例创建数据库实例一样。

您创建的新数据库实例必须与预留的数据库实例具有相同的规范，具体为以下方面：

- Amazon Web Services 区域
- 数据库引擎（数据库引擎的版本号不需要匹配。）
- 数据库实例类型
- 数据库实例大小（RDS for Microsoft SQL Server 和 Amazon RDS for Oracle 附带许可版本）
- 版本（RDS for SQL Server 和 RDS for Oracle）
- 许可类型（附带许可或自带许可）

如果新数据库实例的规格与您的账户的现有预留数据库实例匹配，则会按照为预留数据库实例提供的折扣费率向您收费。否则，将以按需费率对数据库实例进行收费。

您可以修改用作预留数据库实例的数据库实例。如果修改在预留数据库实例的规范范围内，则部分或全部折扣仍适用于修改后的数据库实例。如果修改超出规范范围（例如更改实例类），则不再适用折扣。有关更多信息，请参阅 大小灵活的预留数据库实例。

主题

有关预留数据库实例的更多信息（包括定价），请参阅 Amazon RDS 预留实例。

产品类型

预留数据库实例有三种类型（无预付费用、预付部分费用和预付全部费用），使您可以基于预期使用情况优化 Amazon RDS 成本。

> **Note**
>
> 并非所有 RDS 实例类都支持所有预留实例产品类型。例如，某些实例类可能不提供“无预付费用”选项。要确认可用性，请在 Amazon Web Services Management Console中查看预留实例产品或使用 `describe-reserved-db-instances-offerings` Amazon CLI 命令。

无费用预付

　　该选项无需预付款即可访问预留数据库实例。无论使用情况如何，您的“无费用预付”预留数据库实例都将按照期限内的小时数，采用打折小时费率进行计费，无需任何预付款。该选项仅以一年期预留形式提供。

预付部分费用

　　该选项需要预付部分预留数据库实例费用。期限内剩余的小时数无论使用情况如何，都将按照打折小时费率计费。该选项替换了以前的高使用率选项。

预付全部费用

　　所有款项于期限开始时支付，无论使用了多少小时数，剩余期限不会再产生其他任何费用。

如果使用整合账单，则将组织中的所有账户视为一个账户。这意味着，组织中的所有账户都可以享受任何其他账户购买的预留数据库实例的小时成本优惠。有关整合账单的更多信息，请参阅 Amazon 账单和成本管理用户指南中的 Amazon RDS 预留数据库实例。

大小灵活的预留数据库实例

在购买预留数据库实例时，您指定的一项内容是实例类，例如，db.r5.large。有关数据库实例类的更多信息，请参阅 数据库实例类。

如果您具有数据库实例，则需要将其扩展为更大的容量，预留数据库实例将自动应用于扩展的数据库实例。即，在所有数据库实例类大小中自动应用预留数据库实例。大小灵活的预留数据库实例可供具有相同 Amazon Web Services 区域和数据库引擎的数据库实例使用。大小灵活的预留数据库实例只能在其实例类类型中扩展。例如，db.r5.large 的预留数据库实例可以应用于 db.r5.xlarge，但不能应用于 db.r6g.large，因为 db.r5 和 db.r6g 属于不同类型的实例类。

预留数据库实例优惠同时适用于多可用区和单可用区配置。这意味着您可以在相同数据库实例类型的配置之间自由移动。例如，您可以从在一个大型数据库实例（每小时四个标准化单位）上运行的单可用区部署移动到在两个中型数据库实例（2+2 = 每小时 4 个标准化单位）上运行的多可用区部署。

大小灵活的预留数据库实例适用于以下 Amazon RDS 数据库引擎：

- RDS for MariaDB
- RDS for MySQL
- RDS for Oracle、自带许可
- RDS for PostgreSQL

大小灵活性不适用于 RDS for SQL Server 和 RDS for Oracle 附带许可版本。

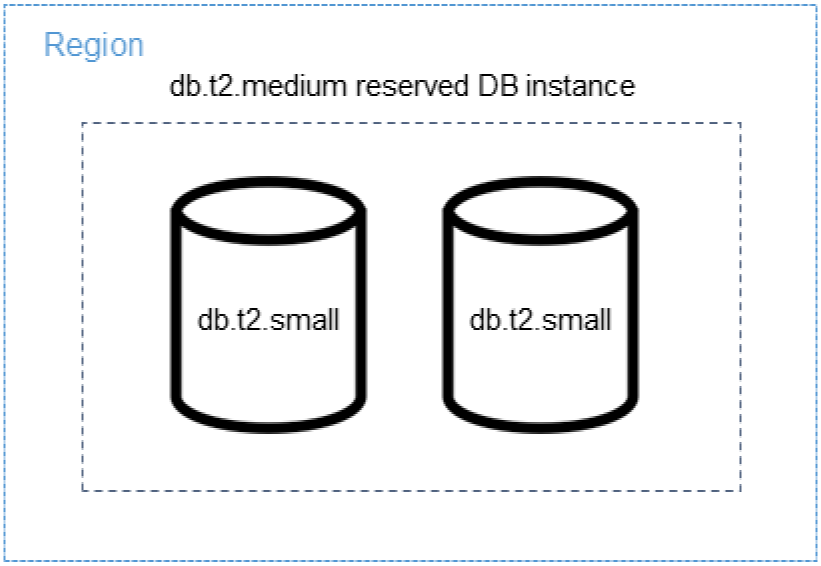
有关在 Aurora 中使用大小灵活的预留实例的详细信息，请参阅 Aurora 的预留数据库实例。

您可以使用每小时标准化单位数来比较不同预留数据库实例大小的使用情况。例如，两个 db.r3.large 数据库实例的一个单位使用量相当于一个 db.r3.small 上的 8 个每小时标准化单位。下表显示了每个数据库实例大小的每小时标准化单位数。

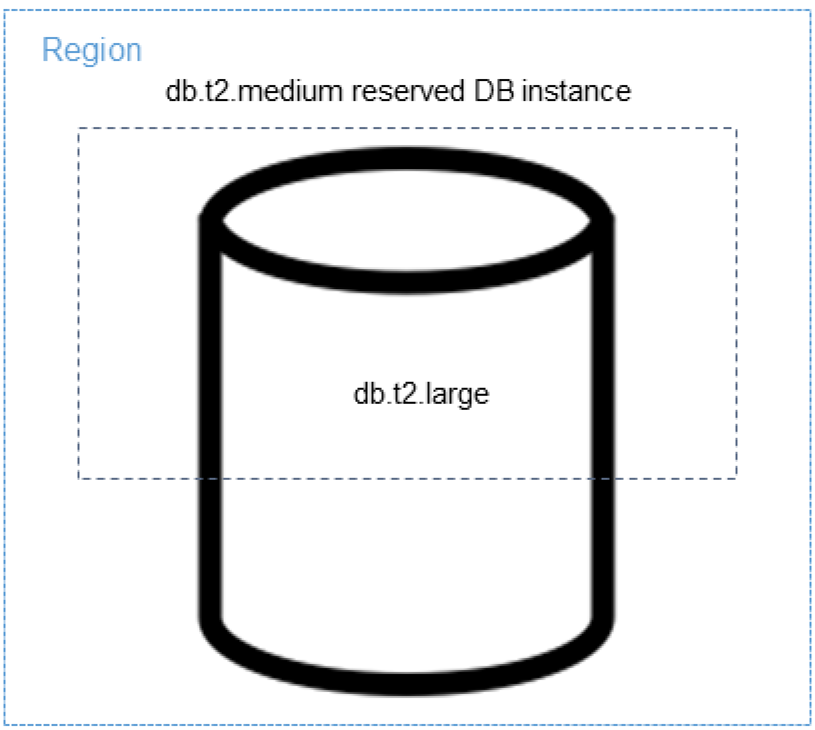
实例大小	单可用区每小时标准化单位（使用一个数据库实例进行部署）	多可用区数据库实例每小时标准化单位（使用一个数据库实例和一个备用数据库实例进行部署）	多可用区数据库集群每小时标准化单位（使用一个数据库实例和两个备用数据库实例进行部署）
micro	0.5	1	1.5
small	1	2	3

实例大小	单可用区每小时标准化单位（使用一个数据库实例进行部署）	多可用区数据库实例每小时标准化单位（使用一个数据库实例和一个备用数据库实例进行部署）	多可用区数据库集群每小时标准化单位（使用一个数据库实例和两个备用数据库实例进行部署）
medium	2	4	6
large	4	8	12
xlarge	8	16	24
2xlarge	16	32	48
4xlarge	32	64	96
6xlarge	48	96	144
8xlarge	64	128	192
10xlarge	80	160	240
12xlarge	96	192	288
16xlarge	128	256	384
24xlarge	192	384	576
32xlarge	256	512	768

例如，假定您购买了 `db.t2.medium` 预留数据库实例，并且您的账户在同一 Amazon Web Services 区域中具有两个正在运行的 `db.t2.small` 数据库实例。在这种情况下，账单优惠将完全应用于两个实例。



或者，如果您的账户在同一 Amazon Web Services 区域中具有一个正在运行的 `db.t2.large` 实例，则账单优惠应用于 50% 的数据库实例使用量。



预留数据库实例计费示例

预留数据库实例的价格并没有为与存储、备份和 I/O 相关的成本提供折扣。它只提供按小时、按需型实例使用量的折扣。以下示例说明了预留数据库实例的每月总成本：

- 美国东部（弗吉尼亚州北部）的 RDS for MySQL 预留单可用区 db.r5.large 数据库实例类，采用“无预付”选项，实例成本为 0.12 美元或每月 90 美元
- 400 GiB 通用型 SSD (gp2) 存储，每月每 GiB 成本为 0.115 美元或每月 45.60 美元
- 600 GiB 备份存储为 0.095 美元或每月 19 美元（400 GiB 免费）

将预留数据库实例的所有这些费用相加（90 + 45.60 + 19 美元），得出每月总成本为 154.60 美元。

如果您选择使用按需数据库实例而不是预留数据库实例，则美国东部（弗吉尼亚州北部）的 RDS for MySQL 单可用区 db.r5.large 数据库实例类的成本为每小时 0.1386 美元或每月 101.18 美元。因此，对于按需数据库实例，将所有这些选项相加（101.18 + 45.60 + 19 美元），得出每月总成本为 165.78 美元。使用预留数据库实例可每月至少为您节省 11 美元。

> **Note**
>
> 该示例中的价格是示例价格，可能与实际价格不符。有关 Amazon RDS 定价信息，请参阅 Amazon RDS 定价。

多可用区数据库集群的预留数据库实例

要为多可用区数据库集群购买等效的预留数据库实例，可以执行以下操作之一：

- 预留三个与集群中的实例具有相同大小的单可用区数据库实例。
- 预留一个多可用区数据库实例和一个单可用区数据库实例，这些实例与集群中的数据库实例大小相同。

例如，假设您具有一个集群，它由三个 db.m6gd.large 数据库实例组成。在这种情况下，您可以购买三个 db.m6gd.large 单可用区预留数据库实例，也可以购买一个 db.m6gd.large 多可用区预留数据库实例和一个 db.m6gd.large 单可用区预留数据库实例。这两个选项中的任何一个都为多可用区数据库集群预留了最大预留实例折扣。

或者，您可以使用大小灵活的数据库实例并购买更大的数据库实例，以涵盖一个或多个集群中的较小数据库实例。例如，如果您有两个集群（总共具有六个 db.m6gd.large 数据库实例），则可以购买三个 db.m6gd.xl 单可用区预留数据库实例。这样做会在两个集群中预留所有六个数据库实例。有关更多信息，请参阅 大小灵活的预留数据库实例。

您可以预留与集群中的数据库实例大小相同的数据库实例，但预留的数据库实例数少于集群中数据库实例的总数。但是，如果您这样做，则只会部分预留集群。例如，假设您的一个集群包含三个

db.m6gd.large 数据库实例，并购买了一个 db.m6gd.large 多可用区预留数据库实例。在这种情况下，仅部分预留此集群，因为预留的数据库实例仅涵盖集群的三个实例中的两个。剩下的数据库实例以按需 db.m6gd.large 每小时费率收费。

有关多可用区数据库集群的更多信息，请参阅 Amazon RDS 的多可用区数据库集群部署。

删除预留数据库实例

预留数据库实例具有一年或三年的使用期限。您无法取消预留数据库实例。不过，您可以删除预留数据库实例折扣涵盖的数据库实例。删除预留数据库实例折扣涵盖的数据库实例的过程与删除任何其他数据库实例相同。

无论您是否使用资源，都需要支付前期费用。

如果删除了预留数据库实例折扣涵盖的数据库实例，您可以启动另一个具有兼容规格的数据库实例。在这种情况下，您可以在预留期限（一年或三年）内继续享受折扣费率。

购买 Amazon RDS 的预留数据库实例

您可以使用 Amazon Web Services Management Console、Amazon CLI 和 RDS API 处理预留数据库实例。

控制台

您可以使用 Amazon Web Services Management Console处理预留数据库实例，如以下过程中所示。

获取有关可用预留数据库实例产品的定价和信息

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择预留实例。
3. 选择购买预留的数据库实例。
4. 对于产品描述，请选择数据库引擎和许可类型。
5. 对于数据库实例类，请选择数据库实例类。
6. 对于部署选项，选择是需要单可用区还是多可用区数据库实例部署。

> **Note**
> 要为多可用区数据库集群部署购买等效的预留数据库实例，请购买三个单可用区预留数据库实例，或者购买一个多可用区和一个单可用区预留数据库实例。有关更多信息，请参阅多可用区数据库集群的预留数据库实例。

7. 对于期限，选择要预留数据库实例的时间长度。
8. 对于产品类型，请选择产品类型。

   选择产品类型后，您可以看到定价信息。

> **Important**
> 可以选择取消以避免购买预留数据库实例和产生任何费用。

在获得有关可用预留数据库实例产品的信息后，您可以使用该信息来购买以下过程中所示的产品。

购买预留数据库实例

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择预留实例。
3. 选择 Purchase reserved DB instance（购买预留数据库实例）。
4. 对于产品描述，请选择数据库引擎和许可类型。
5. 对于数据库实例类，请选择数据库实例类。
6. 对于多可用区部署，选择是需要单可用区还是多可用区数据库实例部署。

> **Note**
> 要为多可用区数据库集群部署购买等效的预留数据库实例，请购买三个单可用区预留数据库实例，或者购买一个多可用区和一个单可用区预留数据库实例。有关更多信息，请参阅多可用区数据库集群的预留数据库实例。

7. 对于期限，选择希望预留数据库实例的时间长度。
8. 对于产品类型，请选择产品类型。

选择产品类型后，您可以看到定价信息。

9. （可选）您可以将自己的标识符分配给购买的预留数据库实例，以帮助您跟踪这些实例。对于预留 ID，请为您的预留数据库实例键入一个标识符。
10. 选择提交。

    您的预留数据库实例已购买，然后显示在 Reserved instances（预留实例）列表中。

在购买了预留数据库实例后，您可以按以下过程中所示来获取有关预留数据库实例的信息。

获取有关 Amazon 账户的预留数据库实例的信息

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择预留实例。

   您的账户的预留数据库实例随即出现。要查看有关特定预留数据库实例的详细信息，请在列表中选择该实例。然后，您可以在控制台底部的详细信息窗格中查看有关该实例的详细信息。

Amazon CLI

您可以使用 Amazon CLI 处理预留数据库实例，如以下示例中所示。

Example 获取可用预留数据库实例服务

要获取有关可用预留数据库实例产品的信息，请调用 Amazon CLI 命令 `describe-reserved-db-instances-offerings`。

```
aws rds describe-reserved-db-instances-offerings
```

此调用返回类似于下述信息的输出：

```

OFFERING  OfferingId                            Class         Multi-AZ  Duration  Fixed
 Price  Usage Price  Description  Offering Type
OFFERING  438012d3-4052-4cc7-b2e3-8d3372e0e706  db.r3.large   y         1y
 1820.00 USD  0.368 USD    mysql         Partial  Upfront
OFFERING  649fd0c8-cf6d-47a0-bfa6-060f8e75e95f  db.r3.small   n         1y
 227.50 USD  0.046 USD    mysql         Partial  Upfront
```

```
OFFERING  123456cd-ab1c-47a0-bfa6-12345667232f  db.r3.small   n         1y
 162.00 USD   0.00 USD    mysql        All      Upfront
    Recurring Charges:   Amount  Currency  Frequency
    Recurring Charges:   0.123   USD       Hourly
OFFERING  123456cd-ab1c-37a0-bfa6-12345667232d  db.r3.large   y         1y
 700.00 USD   0.00 USD    mysql        All      Upfront
    Recurring Charges:   Amount  Currency  Frequency
    Recurring Charges:   1.25    USD       Hourly
OFFERING  123456cd-ab1c-17d0-bfa6-12345667234e  db.r3.xlarge  n         1y
 4242.00 USD   2.42 USD    mysql        No       Upfront
```

在获得有关可用预留数据库实例产品的信息后，您可以使用该信息来购买产品。

要购买预留数据库实例，请将 Amazon CLI 命令 [purchase-reserved-db-instances-offering] 与以下参数结合使用：

- `--reserved-db-instances-offering-id` – 您要购买的产品的 ID。请参阅上述示例以获取产品 ID。
- `--reserved-db-instance-id` – 您可以将自己的标识符分配给购买的预留数据库实例，以帮助您跟踪这些实例。

Example 购买预留数据库实例

以下示例将购买 ID 为 *649fd0c8-cf6d-47a0-bfa6-060f8e75e95f* 的预留数据库实例产品，并分配标识符 *MyReservation*。

对于 Linux、macOS 或 Unix：

```
aws rds purchase-reserved-db-instances-offering \
    --reserved-db-instances-offering-id 649fd0c8-cf6d-47a0-bfa6-060f8e75e95f \
    --reserved-db-instance-id MyReservation
```

对于 Windows：

```
aws rds purchase-reserved-db-instances-offering ^
    --reserved-db-instances-offering-id 649fd0c8-cf6d-47a0-bfa6-060f8e75e95f ^
    --reserved-db-instance-id MyReservation
```

该命令返回的输出类似于下方内容：

```
RESERVATION  ReservationId      Class         Multi-AZ  Start Time
 Duration  Fixed Price  Usage Price  Count  State            Description  Offering Type
RESERVATION  MyReservation      db.r3.small  y         2011-12-19T00:30:23.247Z  1y
    455.00 USD   0.092 USD    1      payment-pending  mysql        Partial  Upfront
```

在购买了预留数据库实例后，您可以获取有关预留数据库实例的信息。

要获取有关您的 Amazon 账户的预留数据库实例的信息，请按照以下示例所示调用 Amazon CLI 命令 `describe-reserved-db-instances`。

Example 获取预留数据库实例

```
aws rds describe-reserved-db-instances
```

该命令返回的输出类似于下方内容：

```
RESERVATION  ReservationId     Class         Multi-AZ  Start Time
 Duration  Fixed Price  Usage Price  Count  State    Description  Offering Type
RESERVATION  MyReservation     db.r3.small  y         2011-12-09T23:37:44.720Z  1y
   455.00 USD   0.092 USD    1      retired  mysql        Partial  Upfront
```

RDS API

您可以使用 RDS API 处理预留数据库实例。

- 要获取有关可用预留数据库实例产品的信息，请调用 Amazon RDS API 操作 `DescribeReservedDBInstancesOfferings`。
- 在获得有关可用预留数据库实例产品的信息后，您可以使用该信息来购买产品。调用带以下参数的 `PurchaseReservedDBInstancesOffering` RDS API 操作：
  - `--reserved-db-instances-offering-id` – 您要购买的产品的 ID。
  - `--reserved-db-instance-id` – 您可以将自己的标识符分配给购买的预留数据库实例，以帮助您跟踪这些实例。
- 在购买了预留数据库实例后，您可以获取有关预留数据库实例的信息。调用 `DescribeReservedDBInstances` RDS API 操作。

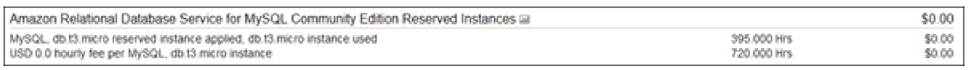
查看 Amazon RDS 的预留数据库实例的账单

您可以在 Amazon Web Services Management Console 中账单控制面板上查看预留数据库实例的账单。

要查看预留数据库实例账单

1. 登录到 Amazon Web Services Management Console。
2. 从右上角的 account menu（账户菜单）中，选择 Billing Dashboard（账单控制面板）。
3. 选择控制面板右上角的 Bill Details（账单详细信息）。
4. 在 Amazon Service Charges（ 服务费用）项下，展开 Relational Database Service（关系数据库服务）。
5. 展开您的预留数据库实例所在的 Amazon Web Services 区域，例如 US West (Oregon) [美国西部（俄勒冈州）]。

   您的预留数据库实例及其当月的每小时费用显示在 Amazon Relational Database Service for ***Database Engine*** Reserved Instances（适用于数据库引擎预留实例的 Amazon Relational Database Service）中。

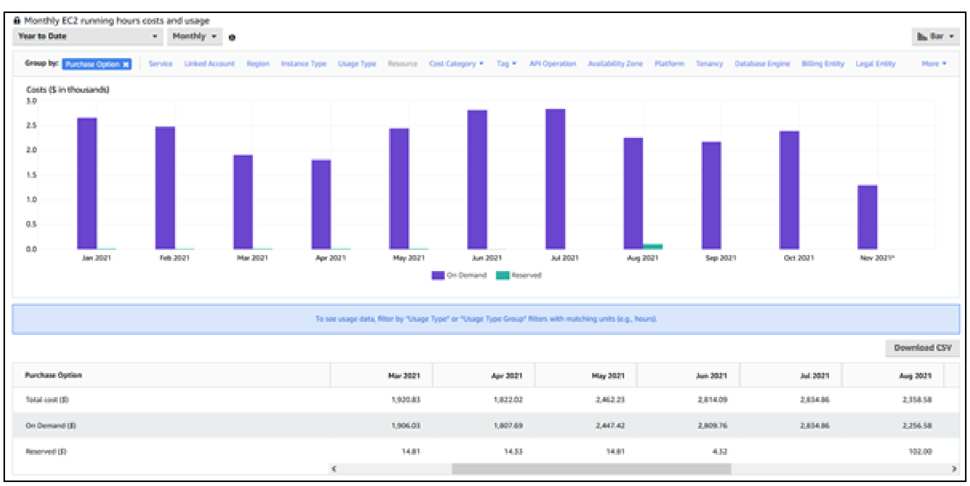
   

Amazon Relational Database Service for MySQL Community Edition Reserved Instances		$0.00
MySQL, db.t3.micro reserved instance applied, db.t3.micro instance used	395.000 Hrs	$0.00
USD 0.0 hourly fee per MySQL, db.t3.micro instance	720.000 Hrs	$0.00

   此示例中的预留数据库实例是预付全部费用购买的，因此不收取小时费用。
6. 选择 Reserved Instances（预留实例）标题旁的 Cost Explorer（条形图）图标。

   Cost Explorer 将显示 Monthly EC2 running hours costs and usage（每月 EC2 运行时间成本和使用情况）图表。
7. 清除图表右侧的 Usage Type Group（使用类型组）筛选条件。
8. 选择要检查使用成本的时间段和时间单位。

   以下示例按月显示了今年迄今按需和预留数据库实例的使用成本。

Purchase Option	Mar 2021	Apr 2021	May 2021	Jun 2021	Jul 2021	Aug 2021
Total cost ($)	1,920.83	1,822.02	2,462.23	2,814.09	2,834.86	2,358.58
On Demand ($)	1,906.03	1,807.69	2,447.42	2,809.76	2,834.86	2,256.58
Reserved ($)	14.81	14.33	14.81	4.32		102.00

2021 年 1 月至 6 月的预留数据库实例成本是预付部分费用实例的月度费用，而 2021 年 8 月的费用是预付全部费用实例的一次性费用。

预付部分费用实例的预留实例折扣已于 2021 年 6 月到期，但该数据库实例未被删除。到期日期之后，只需以按需费率收费。

设置 Amazon RDS 环境

本页提供了设置 Amazon Relational Database Service 的全面指南，包括账户配置、安全性和资源管理。它为您演练高效创建、管理和保护数据库环境的基本步骤。无论您是刚接触 Amazon RDS，还是正在进行设置来满足特定的要求，这些章节都有助于确保您的设置得到优化并符合最佳实践。

主题

如果您已有 Amazon Web Services 账户，知道您的 Amazon RDS 要求并且想要使用 IAM 和 VPC 安全组的默认值，请向前跳至开始使用 Amazon RDS。

注册 Amazon Web Services 账户

如果您还没有 Amazon Web Services 账户，请完成以下步骤来创建一个。

注册 Amazon Web Services 账户

1. 打开 https://portal.aws.amazon.com/billing/signup。
2. 按照屏幕上的说明操作。

   在注册时，将接到电话或收到短信，要求使用电话键盘输入一个验证码。

   当您注册 Amazon Web Services 账户 时，系统将会创建一个 Amazon Web Services 账户根用户。根用户有权访问该账户中的所有 Amazon Web Services 服务和资源。作为最佳安全实践，请为用户分配管理访问权限，并且只使用根用户来执行需要根用户访问权限的任务。

注册过程完成后，Amazon 会向您发送一封确认电子邮件。在任何时候，您都可以通过转至 https://aws.amazon.com/ 并选择我的账户来查看当前的账户活动并管理您的账户。

保护 IAM 用户

注册 Amazon Web Services 账户 后，启用多重身份验证（MFA）保护您的管理用户。有关说明，请参阅《IAM 用户指南》中的 为 IAM 用户启用虚拟 MFA 设备（控制台）。

要授予其他用户访问您的 Amazon Web Services 账户资源的权限，请创建 IAM 用户。为了保护您的 IAM 用户，请启用 MFA 并仅向 IAM 用户授予执行任务所需的权限。

有关创建和保护 IAM 用户的更多信息，请参阅《IAM 用户指南》中的以下主题：

- 在您的 Amazon Web Services 账户中创建 IAM 用户
- 适用于 Amazon 资源的访问权限管理
- 基于 IAM 身份的策略示例

授权以编程方式访问

如果用户需要在 Amazon Web Services Management Console之外与 Amazon 交互，则需要编程式访问权限。Amazon API 和 Amazon Command Line Interface 需要访问密钥。可能的话，创建临时凭证，该凭证由一个访问密钥 ID、一个秘密访问密钥和一个指示凭证何时到期的安全令牌组成。

要向用户授予编程式访问权限，请选择以下选项之一。

哪个用户需要编程式访问权限？	目的	方式
IAM	使用短期凭证签署对 Amazon CLI 或 Amazon API 的编程式请求（直接或使用 Amazon SDK）。	按照《IAM 用户指南》中将临时凭证用于 Amazon 资源中的说明进行操作。
IAM	（不推荐使用） 使用长期凭证签署对 Amazon CLI 或 Amazon API 的编程式请求（直接或使用 Amazon SDK）。	按照《IAM 用户指南》中管理 IAM 用户的访问密钥中的说明进行操作。

确定要求

Amazon RDS 的基本构建基块是数据库实例。在数据库实例中，您可以创建数据库。数据库实例提供了称为终端节点 的网络地址。您的应用程序使用此终端节点连接到您的数据库实例。在创建数据库实例时，您指定存储、内存、数据库引擎和版本、网络配置、安全性以及维护时段等详细信息。您通过安全组控制对数据库实例的网络访问。

在创建数据库实例和安全组之前，您必须知道数据库实例和网络需求。下面是要考虑的一些重要事项：

- 资源要求 – 应用程序或服务的内存和处理器要求是什么？ 您使用这些设置来帮助您确定要使用哪个数据库实例类。有关数据库实例类的规范，请参阅 数据库实例类。
- VPC、子网和安全组 – 您的数据库实例很可能在一个 Virtual Private Cloud (VPC) 中。要连接到数据库实例，您需要设置安全组规则。根据您使用的 VPC 的类型和您使用它的方式，设置这些规则的方式有所不同。例如，您可以使用：默认 VPC 或用户定义的 VPC。

  下面的列表说明每个 VPC 选项的规则：

  - 默认 VPC – 如果您的Amazon账户在当前Amazon区域内有一个默认 VPC，则该 VPC 将配置为支持数据库实例。如果您在创建数据库实例时指定默认 VPC，请执行以下操作：
    - 确保创建一个 VPC 安全组，该安全组对从应用程序或服务到 Amazon RDS 数据库实例的连接进行授权。使用 VPC 控制台中的 Security Group (安全组) 选项或 Amazon CLI 创建 VPC 安全组。有关信息，请参阅 步骤 3：创建 VPC 安全组。
    - 指定默认数据库子网组。如果这是您在此Amazon区域内创建的第一个数据库实例，则 Amazon RDS 将在创建数据库实例时创建默认的数据库子网组。
  - 用户定义的 VPC – 如果您希望在创建数据库实例时指定用户定义的 VPC，请了解以下内容：
    - 确保创建一个 VPC 安全组，该安全组对从应用程序或服务到 Amazon RDS 数据库实例的连接进行授权。使用 VPC 控制台中的 Security Group (安全组) 选项或 Amazon CLI 创建 VPC 安全组。有关信息，请参阅 步骤 3：创建 VPC 安全组。
    - 该 VPC 必须满足特定要求才能托管数据库实例，例如至少拥有两个子网，每个子网分别位于一个独立的可用区中。有关信息，请参阅 Amazon VPC 和 Amazon RDS。
    - 确保指定数据库子网组，以定义数据库实例可以使用该 VPC 中的哪些子网。有关信息，请参阅在 VPC 中使用数据库实例中的“数据库子网组”部分。
- 高可用性 – 是否需要失效转移支持？ 在 Amazon RDS 上，多可用区部署会创建一个主数据库实例，并在另一个可用区中创建第二个备用数据库实例以支持失效转移。我们建议将多可用区部署用于生产工作负载以保持高可用性。对于开发和测试用途，您可以使用不是多可用区的部署。有关更多信息，请参阅配置和管理 Amazon RDS 的多可用区部署。

- IAM 策略 – 您的 Amazon 账户是否具有相应策略来授予执行 Amazon RDS 操作所需的权限？ 如要使用 IAM 证书连接到Amazon，您的 IAM 账户必须拥有 IAM 政策来授予执行 Amazon RDS 操作所需的权限。有关更多信息，请参阅Amazon RDS 的 Identity and Access Management。
- 开放端口 – 您的数据库侦听哪个 TCP/IP 端口？ 有些公司的防火墙可能会阻止与您的数据库引擎的默认端口进行连接。如果您公司的防火墙不允许连接默认端口，请为新数据库实例选择其他端口。在创建对指定端口进行侦听的数据库实例后，您可以通过修改该数据库实例来更改端口。
- Amazon 区域 – 您希望您的数据库位于哪个 Amazon 区域内？ 通过让数据库紧邻应用程序或 Web 服务，可以减小网络延迟。有关更多信息，请参阅区域、可用区和 Local Zones。
- 数据库磁盘子系统 – 您的存储要求是什么？ Amazon RDS 提供三种存储类型：
  - 通用型 (SSD)
  - 预置 IOPS (PIOPS)
  - 磁性（也称为标准存储）

  有关 Amazon RDS 存储的更多信息，请参阅Amazon RDS 数据库实例存储。

在了解创建安全组和数据库实例所需的信息之后，请继续执行下一步。

通过创建安全组提供对 VPC 中的数据库实例的访问

VPC 安全组提供了对 VPC 中的数据库实例的访问权。它们充当关联数据库实例的防火墙，在数据库实例级别控制入站和出站流量。默认情况下，系统将创建带防火墙和默认安全组 (用于保护数据库实例) 的数据库实例。

在连接到数据库实例之前，必须先向允许连接的安全组添加规则。使用网络和配置信息来创建允许访问数据库实例的规则。

例如，假设您有一个访问 VPC 中的数据库实例上的数据库的应用程序。在这种情况下，您必须添加指定应用程序用于访问数据库的端口范围和 IP 地址的自定义 TCP 规则。如果您有位于 Amazon EC2 实例上的应用程序，则可以使用您为 Amazon EC2 实例设置的安全组。

创建数据库实例时，您可以配置 Amazon EC2 实例与数据库实例之间的连接。有关更多信息，请参阅配置与 EC2 实例的自动网络连接。

> **Tip**
>
> 创建数据库实例时，您可以在 Amazon EC2 实例和数据库实例之间自动设置网络连接。有关更多信息，请参阅配置与 EC2 实例的自动网络连接。

有关如何将 Amazon Lightsail 中的资源连接到数据库实例的信息，请参阅 Connect Lightsail resources to Amazon Web Services 服务 using VPC peering。

有关访问数据库实例的常见场景的信息，请参阅 在 VPC 中访问数据库实例的场景。

创建 VPC 安全组

1. 登录到Amazon Web Services Management Console并打开 Amazon VPC 控制台，网址：https://console.aws.amazon.com/vpc。

> **Note**
> 确保您在 VPC 控制台中，而不是 RDS 控制台。

2. 在 Amazon Web Services Management Console的右上角，选择要在其中创建 VPC 安全组和数据库实例的 Amazon 区域。在该Amazon区域的 Amazon VPC 资源列表中，您应看到至少一个 VPC 和多个子网。如果您没有看到，则说明您在该 Amazon 区域中没有默认 VPC。
3. 在导航窗格中，选择 Security Groups（安全组）。
4. 选择 Create security group（创建安全组）。

   此时将显示 Create security group (创建安全组) 页面。
5. 在 Basic details (基本详细信息) 中，输入 Security group name (安全组名称) 和 Description (描述)。对于 VPC，选择要在其中创建数据库实例的 VPC。
6. 在 Inbound rules (入站规则) 中，请选择 Add rule (添加规则)。

   a. 对于 Type (类型)，选择 Custom TCP (自定义TCP)。
   b. 对于 Port range (端口范围)，输入要用于数据库实例的端口值。
   c. 对于 Source (源)，选择安全组名称或键入您从中访问数据库实例的 IP 地址范围（CIDR 值）。如果您选择我的 IP，这会允许从浏览器中检测到的 IP 地址访问数据库实例。
7. 如果需要添加更多 IP 地址或不同的端口范围，请选择 Add rule (添加规则) 并输入规则的信息。
8. （可选）在 Outbound rules (出站规则) 中，为出站流量添加规则。默认情况下，允许所有出站流量。
9. 选择 Create security group（创建安全组）。

在创建数据库实例时，您可以使用刚创建的 VPC 安全组作为数据库实例的安全组。

> **Note**
>
> 如果您使用默认 VPC，则为您创建跨越该 VPC 的所有子网的默认子网组。在创建数据库实例时，您可以选择默认 VPC 并为数据库子网组选择默认。

完成所需的设置后，可以使用您的要求和安全组来创建数据库实例。为此，请遵循创建 Amazon RDS 数据库实例中的说明。有关通过创建使用特定数据库引擎的数据库实例的入门信息，请参阅下表中的相关文档。

数据库引擎	文档
MariaDB	创建并连接到 MariaDB 数据库实例
Microsoft SQL Server	创建 Microsoft SQL Server 数据库实例并连接到该实例
MySQL	创建并连接到 MySQL 数据库实例
Oracle	创建并连接到 Oracle 数据库实例
PostgreSQL	创建 PostgreSQL 数据库实例并连接到该实例

> **Note**
>
> 如果在创建数据库实例后无法连接到该实例，请参阅无法连接到 Amazon RDS 数据库实例中的故障排除信息。

开始使用 Amazon RDS

在以下示例中，您可以了解如何使用 Amazon Relational Database Service (Amazon RDS) 创建和连接到数据库实例。您可以创建使用 Db2、MariaDB、MySQL、Microsoft SQL Server、Oracle 或 PostgreSQL 的数据库实例。

> **Important**
>
> 务必先完成设置 Amazon RDS 环境部分中的任务，然后才能创建或连接到数据库实例。

对于每个数据库引擎来说，创建数据库实例并连接到数据库实例上的数据库的过程略为不同。选择要使用的以下数据库引擎之一，以了解有关创建并连接到数据库实例的详细信息。在创建并连接到数据库实例后，将提供可帮助您删除数据库实例的说明。

主题

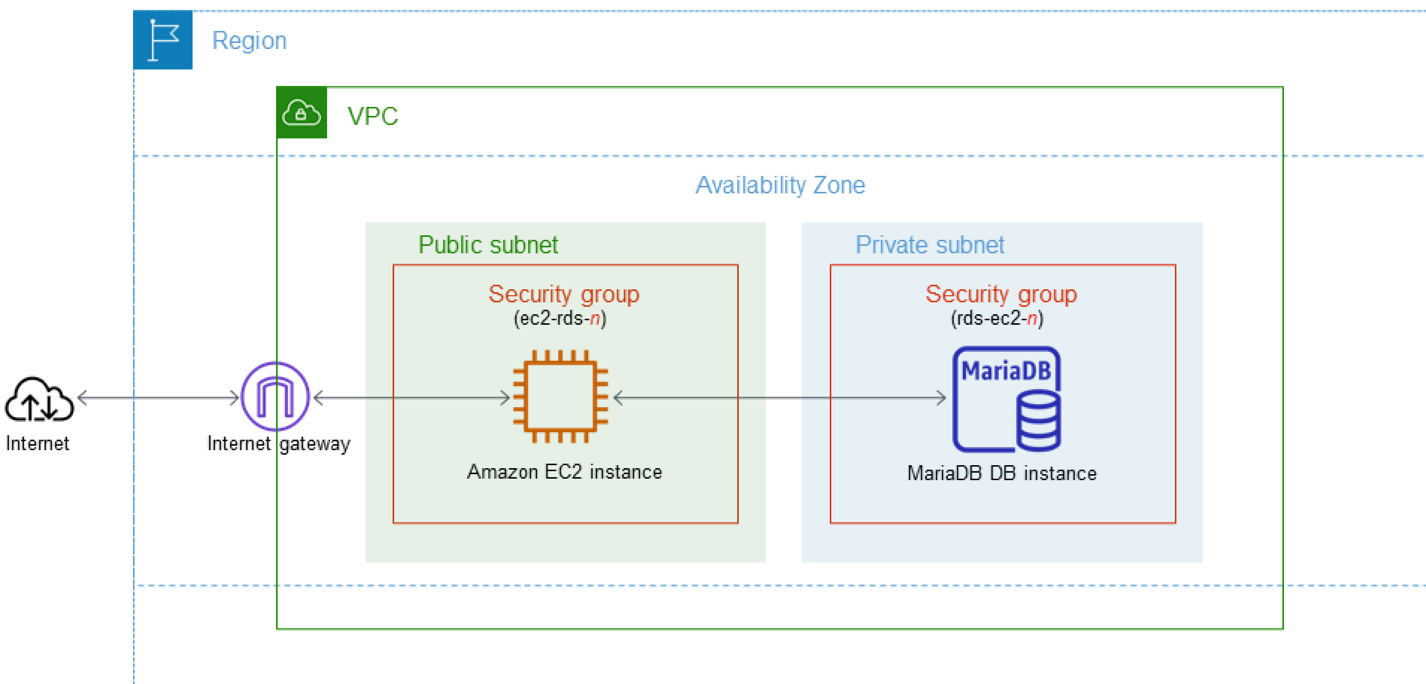
创建并连接到 MariaDB 数据库实例

本教程创建一个 EC2 实例和一个 RDS for MariaDB 数据库实例。本教程向您展示如何使用标准 MySQL 客户端从 EC2 实例访问数据库实例。作为最佳实践，本教程在虚拟私有云（VPC）中创建了私有数据库实例。在大多数情况下，同一 VPC 中的其他资源（例如 EC2 实例）可以访问数据库实例，但 VPC 之外的资源无法访问该实例。

完成本教程后，VPC 的每个可用区中都有一个公有和私有子网。在一个可用区中，EC2 实例在公有子网中，数据库实例在私有子网中。

> **Important**
>
> 创建 Amazon Web Services 账户不会收费。但是，在完成本教程过程中使用 资源可能会产生费用。完成本教程后，如果不再需要这些资源，可以将其删除。

下图显示了教程完成时的配置。



本教程可帮助您通过以下方法之一创建资源：

1. 使用 Amazon Web Services Management Console - 创建 EC2 实例 和 创建 MariaDB 数据库实例
2. 使用 Amazon CloudFormation 创建数据库实例和 EC2 实例 - （可选）使用 Amazon CloudFormation 创建 VPC、EC2 实例和 MariaDB 实例

第一种方法使用轻松创建，通过 Amazon Web Services Management Console 创建私有 MariaDB 数据库实例。您可以仅指定数据库引擎类型、数据库实例大小和数据库实例标识符。轻松创建为其他配置选项使用默认设置。

如果改用标准创建，则在创建数据库实例时可以指定更多配置选项。这些选项包括可用性、安全性、备份和维护设置。要创建公有数据库实例，必须使用标准创建。有关信息，请参阅创建 Amazon RDS 数据库实例。

主题

先决条件

在开始之前，请完成以下各节中的步骤：

- 注册 Amazon Web Services 账户
- 保护 IAM 用户

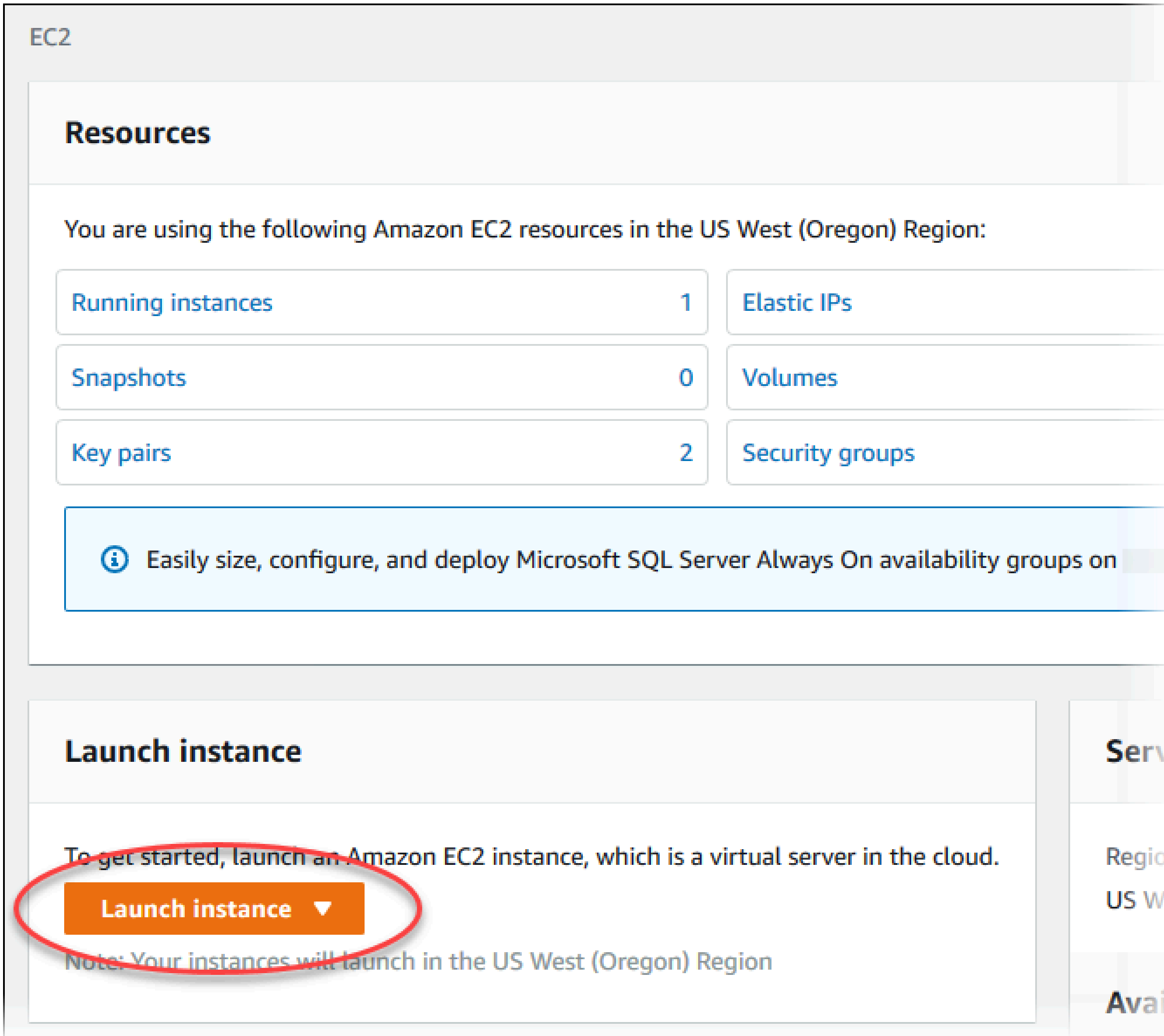
创建 EC2 实例

创建将用于连接到数据库的 Amazon EC2 实例。

创建 EC2 实例

1. 登录到 Amazon Web Services Management Console 并打开 Amazon EC2 控制台（https://console.aws.amazon.com/ec2/）。
2. 在 Amazon Web Services Management Console的右上角，选择要在其中创建 EC2 实例的 Amazon Web Services 区域。

3. 选择 EC2 控制面板，然后选择启动实例，如下图所示。

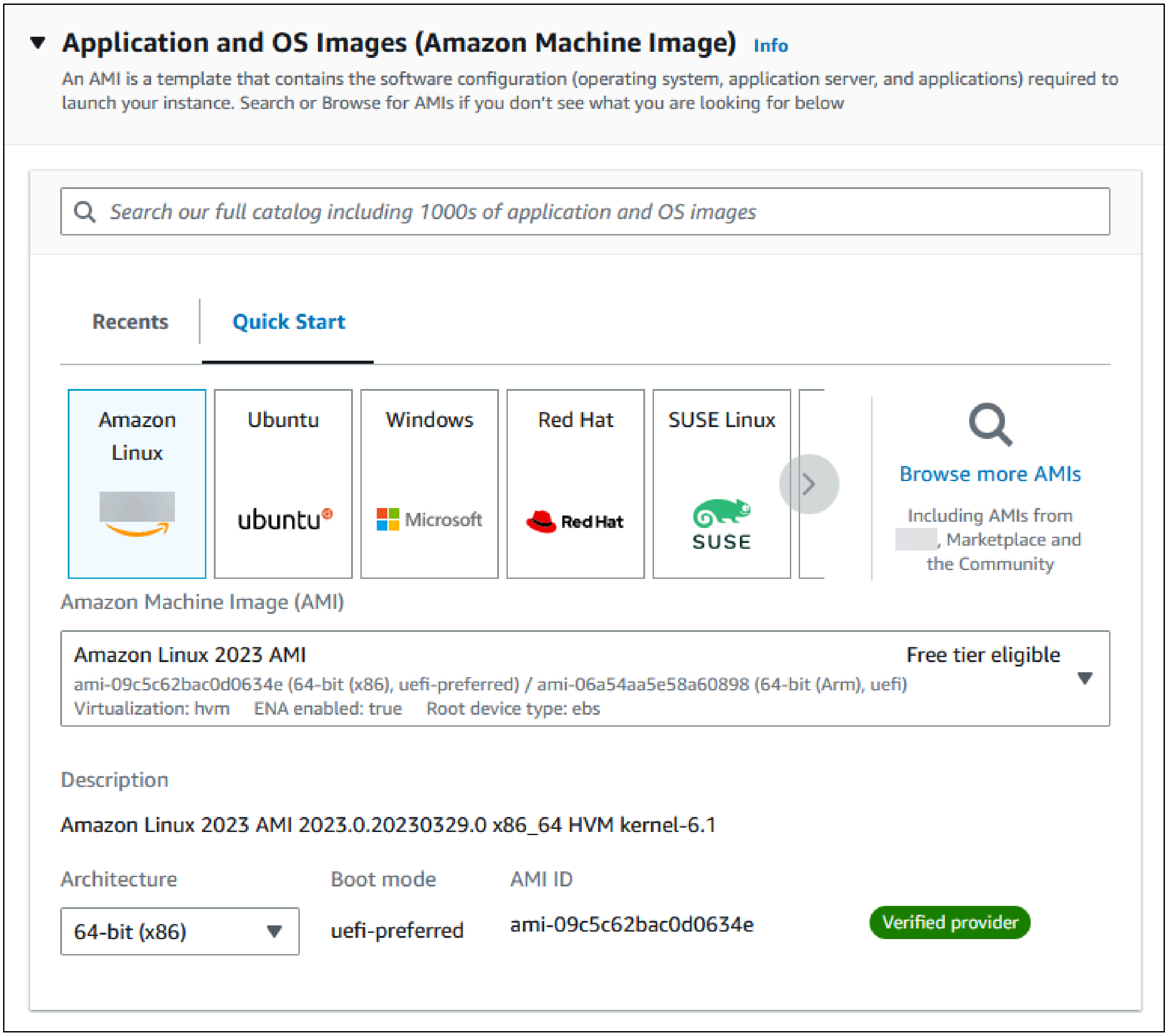


启动实例页面打开。

4. 在启动实例页面上选择以下设置。

   a. 在 Name and tags（名称和标签）下，对于 Name（名称），输入 **ec2-database-connect**。

   b. 在应用程序和操作系统映像（Amazon 机器映像）下，选择 Amazon Linux，然后选择 Amazon Linux 2023 AMI。对于其他选项，保留默认选择。



c. 在 Instance type（实例类型）下，选择 t2.micro。

d. 在 Key pair (login) [密钥对（登录）] 下，选择 Key pair name（密钥对名称）以使用现有密钥对。要为 Amazon EC2 实例创建新的密钥对，请选择 Create new key pair（创建新的密钥对），然后使用 Create key pair（创建密钥对）窗口来创建它。

   有关创建新的密钥对的更多信息，请参阅《Amazon EC2 用户指南》中的创建密钥对。

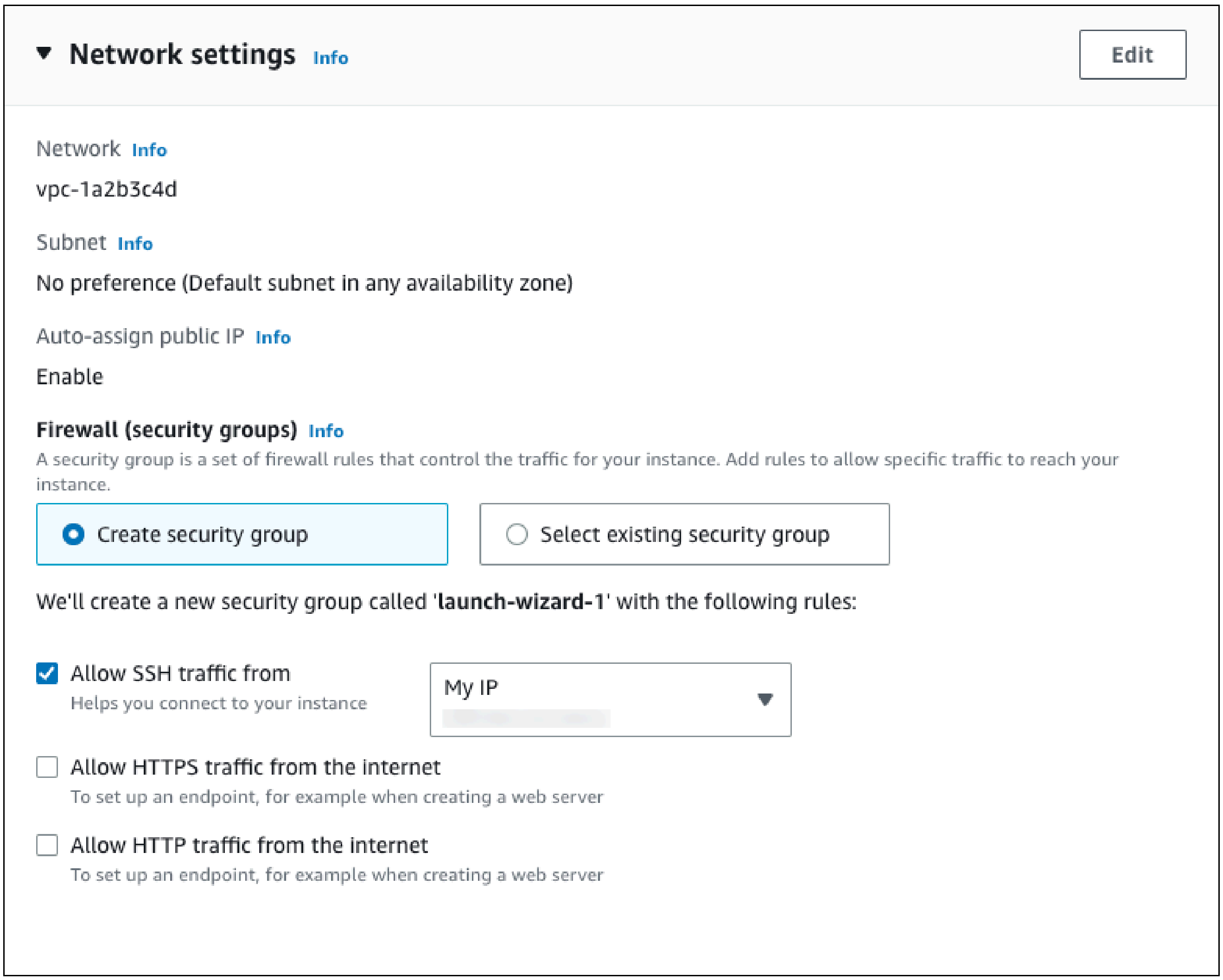
e. 对于网络设置中的允许 SSH 流量，选择 EC2 实例的 SSH 连接来源。

   如果显示的 IP 地址对于 SSH 连接是正确的，您可以选择 My IP（我的 IP）。否则，您可以确定要用来通过 Secure Shell（SSH）连接到 VPC 中的 EC2 实例的 IP 地址。要确定您的公有 IP 地址，请在新的浏览器窗口或标签页中，使用 https://checkip.amazonaws.com 上的服务。IP 地址的示例为 192.0.2.1/32。

在许多情况下，您可能通过互联网服务提供商（ISP）进行连接，或者在不使用静态 IP 地址的情况下从防火墙之后进行连接。如果是这样，请确保确定客户端计算机使用的 IP 地址范围。

> **Warning**
>
> 如果您使用 `0.0.0.0/0` 进行 SSH 访问，则所有 IP 地址可能能够使用 SSH 访问您的公有 EC2 实例。在测试环境下短时间内，此方法尚可接受，但它对于生产环境并不安全。在生产环境中，将仅向特定 IP 地址或地址范围授权使用 SSH 访问您的 EC2 实例。

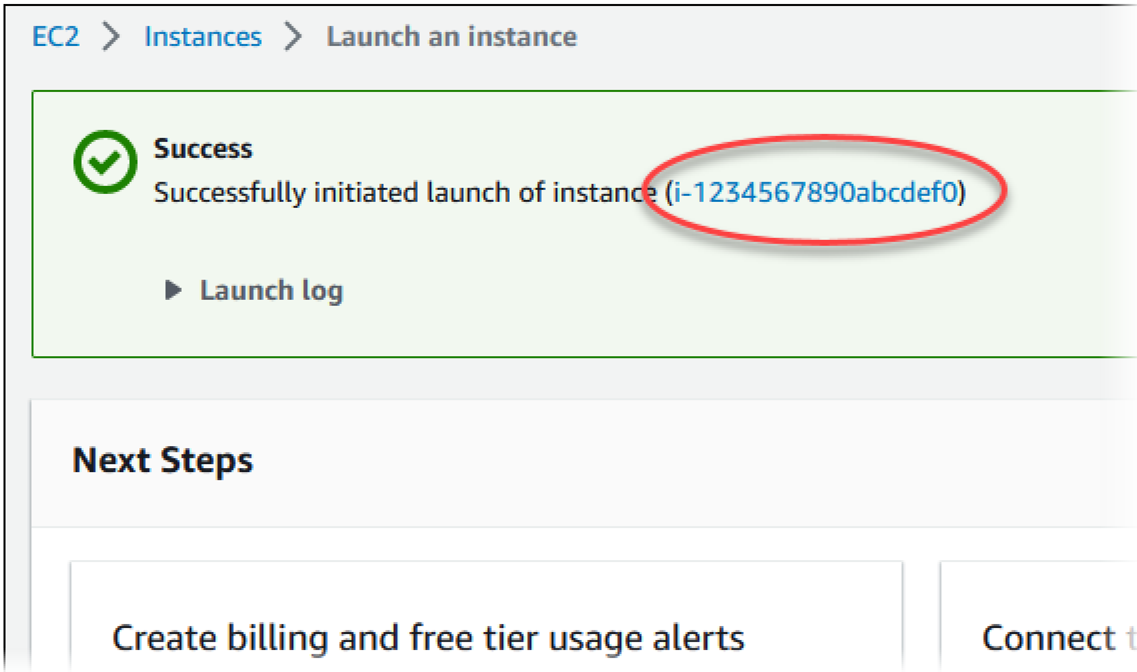
下图显示了网络设置部分的示例。



f. 对于其余部分保留默认值。

g. 查看摘要面板中您的 EC2 实例配置的摘要，当您准备好后，选择启动实例。

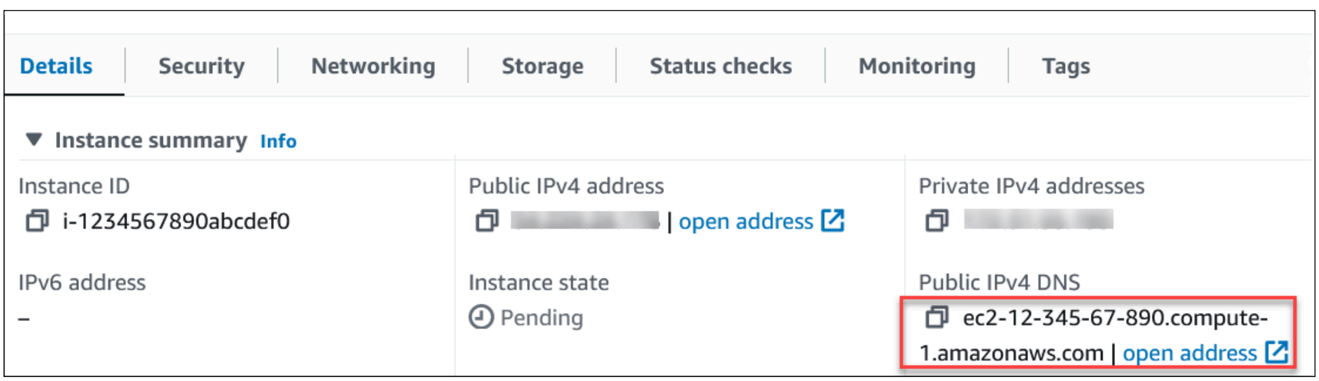
5. 在启动状态页面上，记下新 EC2 实例的标识符，例如：`i-1234567890abcdef0`。



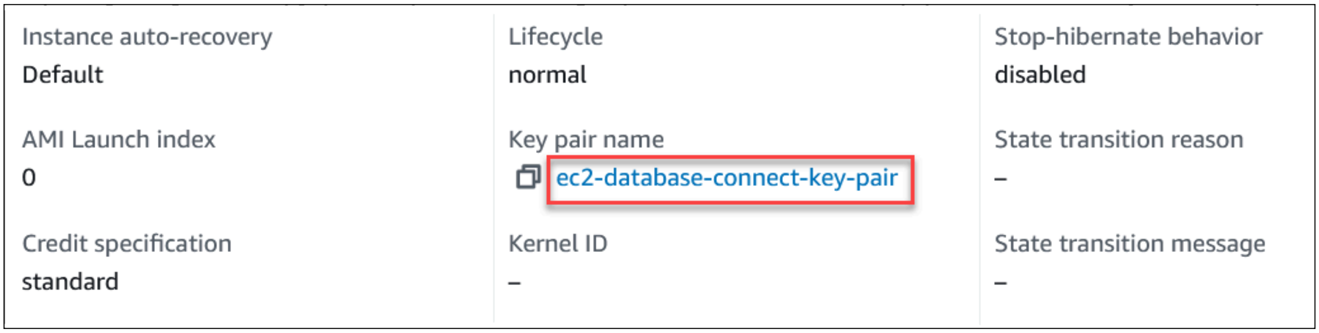
6. 选择 EC2 实例标识符以打开 EC2 实例列表，然后选择您的 EC2 实例。

7. 在详细信息选项卡中，记下使用 SSH 进行连接时所需的以下值：

a. 在实例摘要中，记下公有 IPv4 DNS 的值。



b. 在实例详细信息中，记下密钥对名称的值。



8. 等待 EC2 实例的实例状态变为正在运行，然后继续。

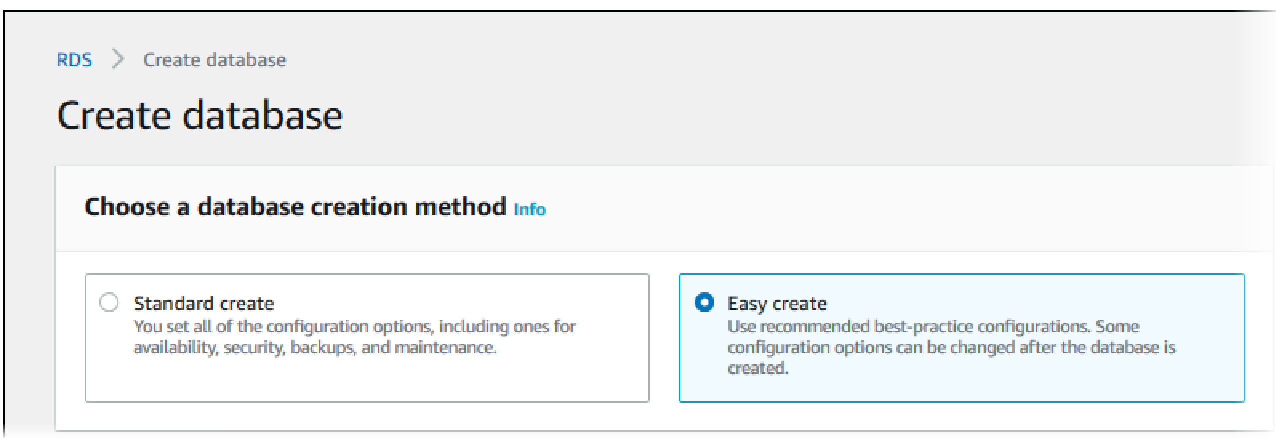
创建 MariaDB 数据库实例

Amazon RDS 的基本构建基块是数据库实例。这是您将要在其中运行 MariaDB 数据库的环境。

在该示例中，您使用轻松创建以创建一个数据库实例，此实例运行带有 db.t4g.micro 数据库实例类的 MariaDB 数据库引擎。

在启用“Easy create (轻松创建)”的情况下创建 MariaDB 数据库实例

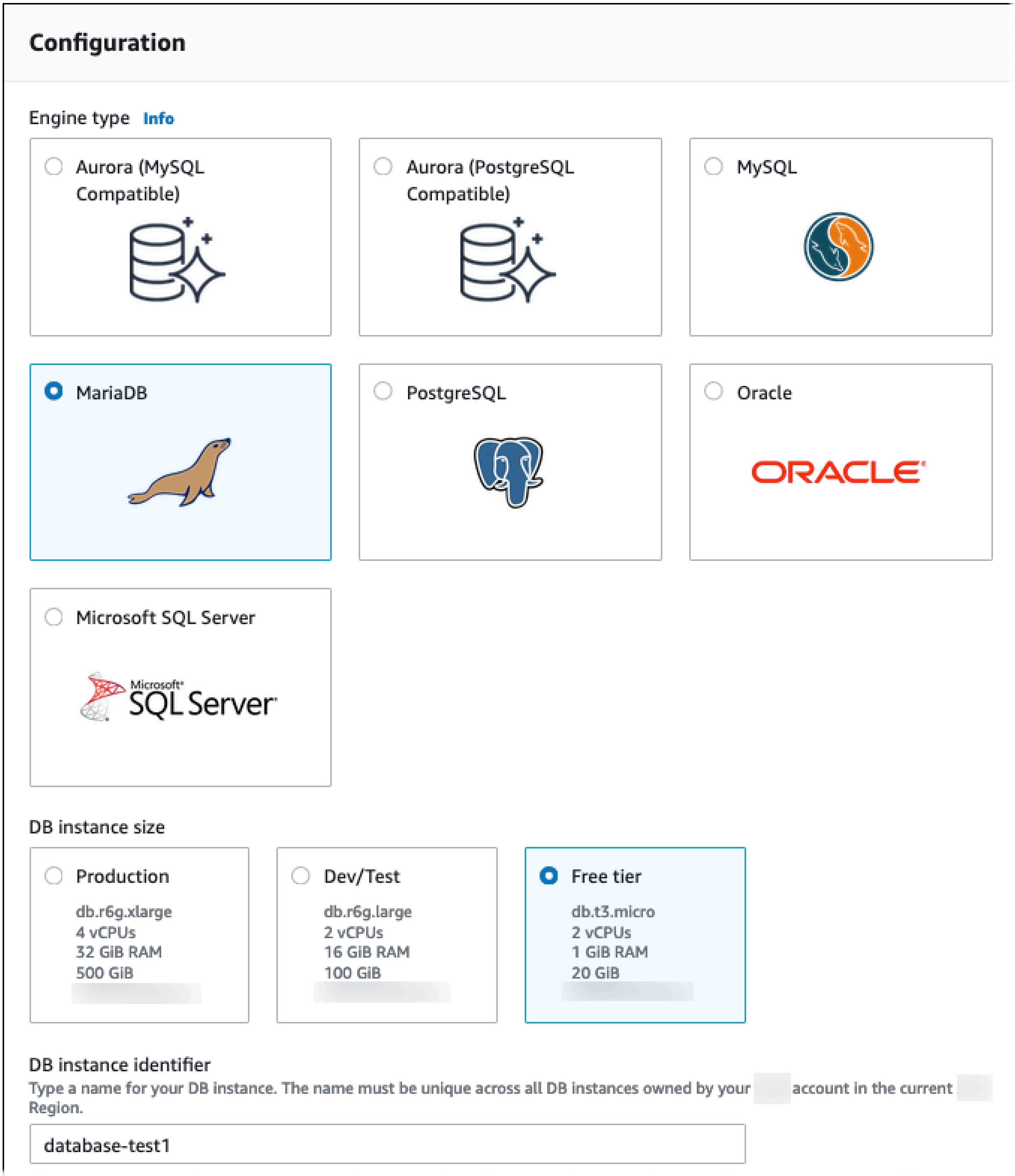
1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在 Amazon RDS 控制台的右上角，选择要在其中创建数据库实例的 Amazon Web Services 区域。
3. 在导航窗格中，选择 Databases（数据库）。
4. 选择创建数据库，然后选择轻松创建。



5. 在配置中，选择 MariaDB。
6. 对于 数据库实例大小，选择 免费套餐。

7. 对于 DB instance identifier（数据库实例标识符），输入 **database-test1**。
8. 对于主用户名，输入主用户的名称，或者保留默认名称。

   创建数据库页面应类似于以下图像。

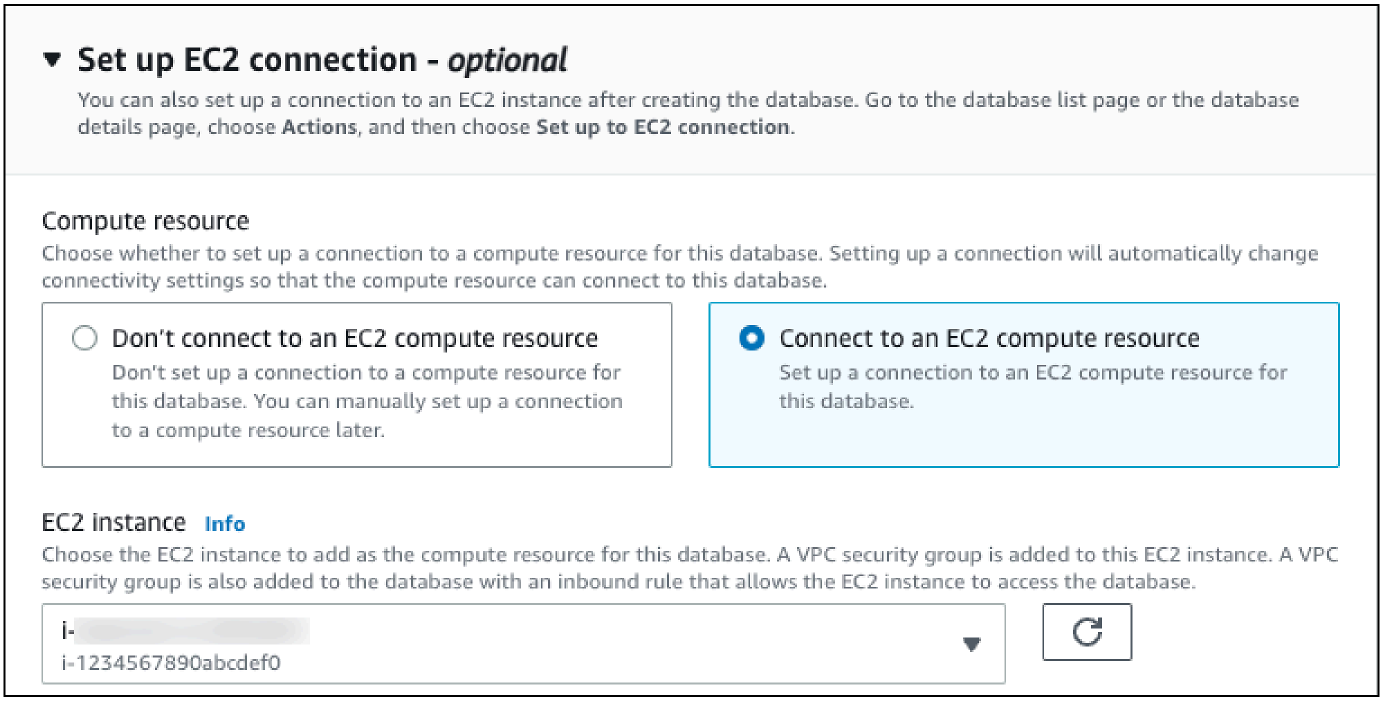


9. 要为数据库实例使用自动生成的主密码，请选择自动生成密码。

   要输入主密码，请清除自动生成密码，然后在主密码和确认主密码中输入相同的密码。

10. 要设置与您之前创建的 EC2 实例的连接，请展开设置 EC2 连接 – 可选。

    选择连接到 EC2 计算资源。选择您之前创建的 EC2 实例。



11. 展开查看轻松创建的默认设置。

View default settings for Easy create

Easy create sets the following configurations to their default values, some of which can be changed later. If you want to change any of these settings now, use **Standard create**.

Configuration	Value	Editable after database is created
Encryption	Enabled	No
VPC	Default VPC (vpc-1a2b3c4d)	No
Option group	default:mariadb-10-6	Yes
Subnet group	default	Yes
Automatic backups	Enabled	Yes
VPC security group	sg-1234567	Yes
Publicly accessible	No	Yes
Database port	3306	Yes
DB instance identifier	database-test1	Yes
DB engine version	10.6.10	Yes
DB parameter group	default.mariadb10.6	Yes
Performance insights	Enabled	Yes
Monitoring	Enabled	Yes
Maintenance	Auto minor version upgrade enabled	Yes
Delete protection	Not enabled	Yes

您可以检查在启用 Easy create (轻松创建) 时使用的默认设置。可在创建数据库后编辑列显示在创建数据库后可以更改的选项。

- 如果该列中的某个设置为否，而您想要不同的设置，则可以使用标准创建来创建数据库实例。

- 如果某个设置在该列中为是，您可以使用标准创建来创建数据库实例，也可以在创建后修改数据库实例以更改该设置。

12. 选择创建数据库。

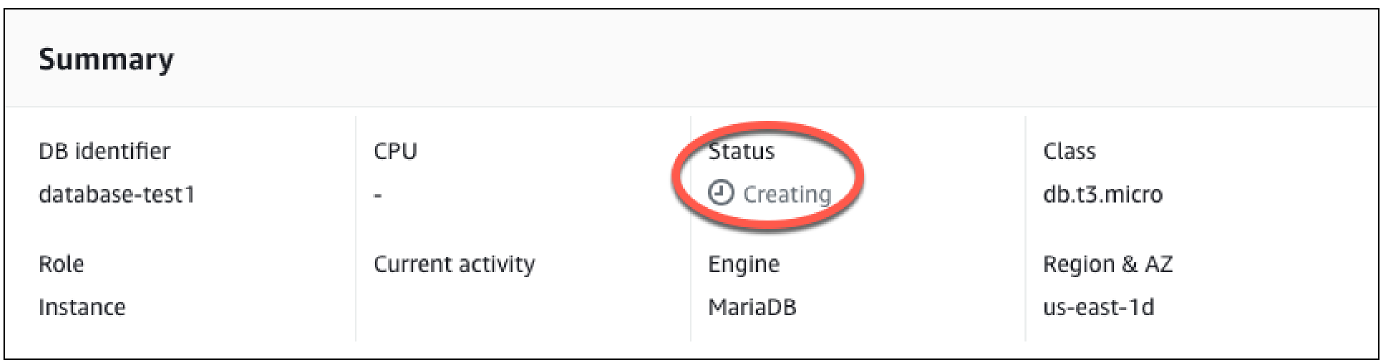
    要查看数据库实例的主用户名和密码，请选择查看凭证详细信息。

    您可以使用显示的用户名和密码，以主用户身份连接到数据库实例。

    > **Important**
    > 您无法再次查看主用户密码。如果您不记录它，您可能需要更改它。
    > 如果需要在数据库实例可用后更改主用户密码，则可以修改数据库实例以执行此操作。有关修改 数据库实例的更多信息，请参阅 修改 Amazon RDS 数据库实例。

13. 在数据库列表中，选择新的 MariaDB 数据库实例的名称以显示其详细信息。

    数据库实例具有正在创建状态，直到此实例就绪可供使用。



    当状态变为 Available（可用）时，您便可以连接到该数据库实例。根据数据库实例类和存储量，新实例可能需要等待 20 分钟时间才可用。

（可选）使用 Amazon CloudFormation 创建 VPC、EC2 实例和 MariaDB 实例

您可以使用 Amazon CloudFormation 通过将基础设施视为代码来配置 Amazon 资源，而不必使用控制台创建 VPC、EC2 实例和 MariaDB 实例。为了帮助您将 Amazon 资源组织成更小、更易于管理的单元，您可以使用 Amazon CloudFormation 嵌套堆栈功能。有关更多信息，请参阅在 Amazon CloudFormation 控制台上创建堆栈和使用嵌套堆栈。

Important

Amazon CloudFormation 是免费的，但 CloudFormation 创建的资源是实时的。您需要为这些资源支付标准使用费，直到您终止使用它们为止。总费用将应该最少的。有关如何最大限度降低费用的信息，请转至 Amazon Free Tier。

要使用 Amazon CloudFormation 控制台创建资源，请执行以下步骤：

- 下载 CloudFormation 模板
- 使用 CloudFormation 配置资源

下载 CloudFormation 模板

CloudFormation 模板是一个 JSON 或 YAML 文本文件，其中包含有关您希望在堆栈中创建的资源的配置信息。此模板还为您创建 VPC 和堡垒主机以及 RDS 实例。

要下载模板文件，请打开以下链接 MariaDB CloudFormation 模板。

在 Github 页面中，单击 Download raw file 按钮以保存模板 YAML 文件。

使用 CloudFormation 配置资源

Note

在开始此过程之前，请确保您的 Amazon Web Services 账户中具有 EC2 实例密钥对。有关更多信息，请参阅 Amazon EC2 密钥对和 Linux 实例。

使用 Amazon CloudFormation 模板时，必须选择正确的参数以确保正确创建资源。按以下步骤操作：

1. 登录到 Amazon Web Services Management Console 并打开 Amazon CloudFormation 控制台 https://console.aws.amazon.com/cloudformation。
2. 选择创建堆栈。
3. 在“指定模板”部分，选择从您的计算机上传模板文件，然后选择下一步。
4. 在指定堆栈详细信息页面上，设置以下参数：
   a. 将堆栈名称设置为 MariaDBTestStack。
   b. 在参数下，通过选择三个可用区来设置可用区。

c. 在 Linux 堡垒主机配置下，在密钥名称中，选择用于登录您的 EC2 实例的密钥对。

d. 在 Linux 堡垒主机配置设置中，将允许的 IP 范围设置为您的 IP 地址。要使用 Secure Shell（SSH）连接到 VPC 中的 EC2 实例，请通过 https://checkip.amazonaws.com 上的服务确定您的公有 IP 地址。IP 地址的示例为 192.0.2.1/32。

> **Warning**
>
> 如果您使用 `0.0.0.0/0` 进行 SSH 访问，则所有 IP 地址可能能够使用 SSH 访问您的公有 EC2 实例。在测试环境下短时间内，此方法尚可接受，但它对于生产环境并不安全。在生产环境中，将仅向特定 IP 地址或地址范围授权使用 SSH 访问您的 EC2 实例。

e. 在数据库常规配置下，将数据库实例类设置为 db.t3.micro。

f. 将数据库名称设置为 **`database-test1`**。

g. 在数据库主用户名中，输入主用户的名称。

h. 在本教程中，将使用 Secrets Manager 管理数据库主用户密码设置为 `false`。

i. 在数据库密码中，设置所选密码。请记住此密码以便在教程中的后续步骤中使用。

j. 在数据库存储配置下，将数据库存储类型设置为 gp2。

k. 在数据库监控配置下，将启用 RDS 性能详情设置为 false。

l. 将所有其他设置保留为默认值。单击下一步继续。

5. 在查看堆栈页面中，在检查数据库和 Linux 堡垒主机选项后，选择提交。

堆栈创建过程完成后，查看名为 BastionStack 和 RDSNS 的堆栈，记下连接到数据库所需的信息。有关更多信息，请参阅在 Amazon Web Services Management Console 上查看 Amazon CloudFormation 堆栈数据和资源。

连接到 MariaDB 数据库实例

您可以使用任何标准 SQL 客户端应用程序连接到数据库实例。在该示例中，您使用 mysql 命令行客户端连接到 MariaDB 数据库实例。

连接到 MariaDB 数据库实例

1. 找到您的数据库实例的端点（DNS 名称）和端口号。

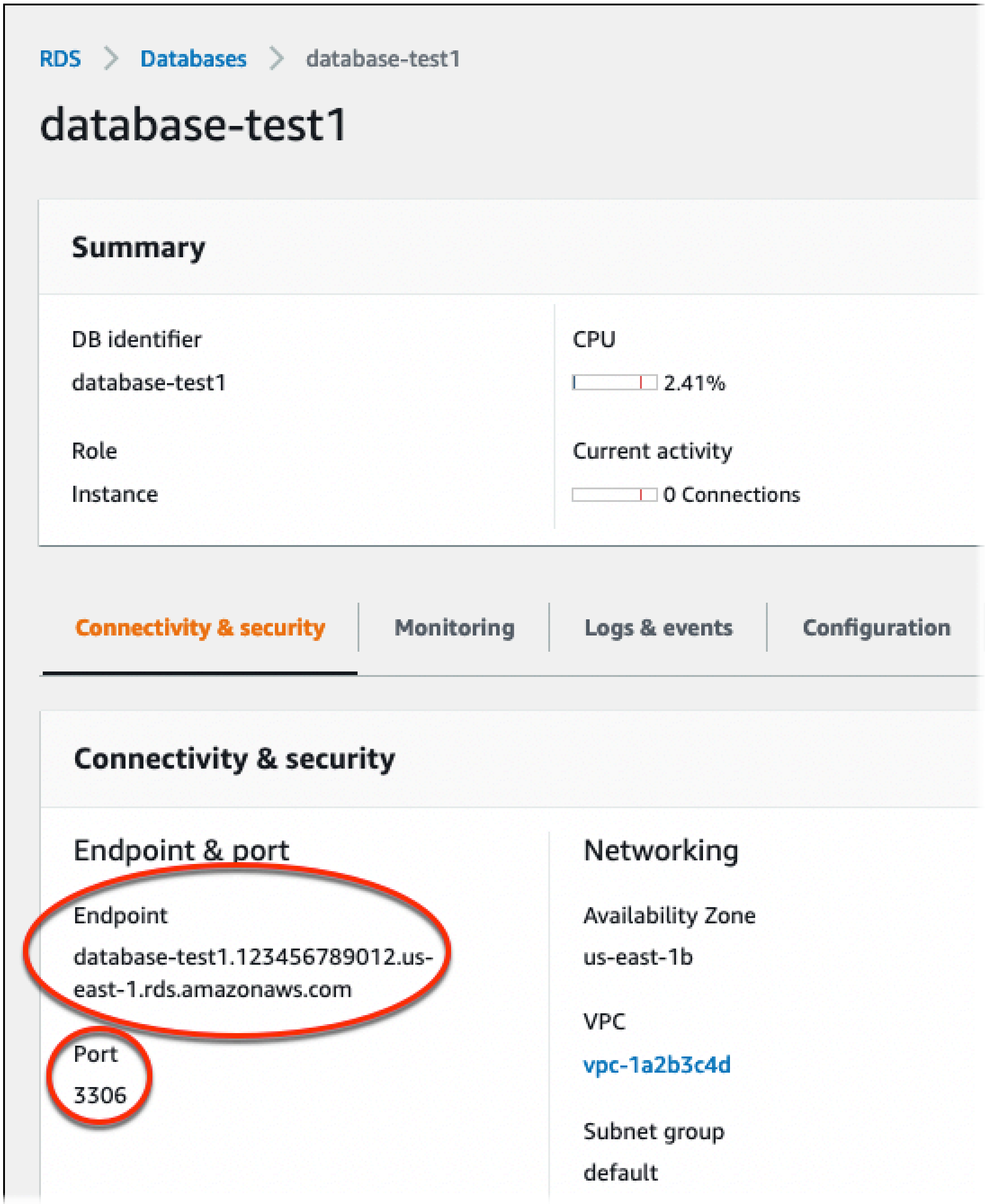
   a. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。

b. 在 Amazon RDS 控制台的右上角，选择数据库实例的 Amazon Web Services 区域。

c. 在导航窗格中，选择 Databases（数据库）。

d. 选择 MariaDB 数据库实例名称以显示其详细信息。

e. 在连接和安全性选项卡上，复制端点。另请注意端口号。您需要端点和端口号才能连接到数据库实例。



2. 按照《Amazon EC2 用户指南》中的[连接到 Linux 实例](#)中的步骤操作，连接到您之前创建的 EC2 实例。

建议您使用 SSH 连接到 EC2 实例。如果 SSH 客户端实用程序安装在 Windows、Linux 或 Mac 上，则可以使用以下命令格式连接到该实例：

```
ssh -i location_of_pem_file ec2-user@ec2-instance-public-dns-name
```

例如，假设在 Linux 上 `ec2-database-connect-key-pair.pem` 存储在 `/dir1` 中，而 EC2 实例的公有 IPv4 DNS 为 `ec2-12-345-678-90.compute-1.amazonaws.com`。SSH 命令将如下所示：

```
ssh -i /dir1/ec2-database-connect-key-pair.pem ec2-
user@ec2-12-345-678-90.compute-1.amazonaws.com
```

3. 通过更新 EC2 实例上的软件，获取最新的错误修复和安全更新。要执行此操作，请使用以下命令。

   > **Note**
   >
   > `-y` 选项安装更新时不提示确认。要在安装前检查更新，请忽略该选项。

   ```
   sudo dnf update -y
   ```

4. 从 MariaDB 安装 mysql 命令行客户端。

   要在 Amazon Linux 2023 上安装 MariaDB 命令行客户端，请运行以下命令：

   ```
   sudo dnf install mariadb105
   ```

5. 连接到 MariaDB 数据库实例。例如，输入以下命令。此操作可让您使用 MySQL 客户端连接到 MariaDB 数据库实例。

   将 *`endpoint`* 替换为数据库实例端点（DNS 名称），并替换用于 *`admin`* 的主用户名。提示输入密码时，提供所使用的主密码。

   ```
   mysql -h endpoint -P 3306 -u admin -p
   ```

   在输入用户的密码后，您应该会看到类似于以下内容的输出。

```
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 156
Server version: 10.6.10-MariaDB-log managed by https://aws.amazon.com/rds/

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

有关连接到 MariaDB 数据库实例的更多信息，请参阅连接到 MariaDB 数据库实例。如果您无法连接到数据库实例，请参阅 无法连接到 Amazon RDS 数据库实例。

出于安全考虑，最佳做法是使用加密连接。仅当客户端和服务器位于同一 VPC 中，并且网络受信任时，才会使用未加密的 MariaDB 连接。有关使用加密连接的信息，请参阅 从 MySQL 命令行客户端使用 SSL/TLS 连接到 Amazon RDS 上的 MariaDB 数据库实例（已加密）。

6. 运行 SQL 命令。

   例如，以下 SQL 命令显示了当前日期和时间：

```
SELECT CURRENT_TIMESTAMP;
```

删除 EC2 实例和数据库实例

在连接到并浏览您创建的示例 EC2 实例和数据库实例之后，删除它们，以便不再为其付费。

如果您使用 Amazon CloudFormation 创建了资源，请跳过此步骤，转至下一步。

删除 EC2 实例

1. 登录到 Amazon Web Services Management Console 并打开 Amazon EC2 控制台（https://console.aws.amazon.com/ec2/）。
2. 在导航窗格中，选择实例。
3. 选择 EC2 实例，然后依次选择实例状态、终止实例。
4. 当系统提示您确认时，选择终止。

有关删除 EC2 实例的更多信息，请参阅《Amazon EC2 用户指南》中的终止实例。

删除无最终数据库快照的数据库实例

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。
3. 选择要删除的数据库实例。
4. 对于 Actions，选择 Delete。
5. 清除是否创建最终快照？和保留自动备份。
6. 完成确认并选择删除。

（可选）删除使用 CloudFormation 创建的 EC2 实例和数据库实例

如果您使用 Amazon CloudFormation 创建了资源，请在连接并浏览 EC2 实例和数据库实例示例之后，删除 CloudFormation 堆栈，以便不再为其付费。

删除 CloudFormation 资源

1. 打开 Amazon CloudFormation 控制台。
2. 在 CloudFormationconsole 的堆栈页面上，选择根堆栈（名称不是 VPCStack、BastionStack 或 RDSNS 的堆栈）。
3. 选择删除。
4. 提示进行确认时，选择删除堆栈。

有关如何删除 CloudFormation 中堆栈的信息，请参阅《Amazon CloudFormation 用户指南》中的在 Amazon CloudFormation 控制台上删除堆栈。

（可选）将您的数据库实例连接到 Lambda 函数

您也可以将您的 RDS for MariaDB 数据库实例连接到 Lambda 无服务器计算资源。Lambda 函数允许您在不预调配或管理基础设施的情况下运行代码。Lambda 函数还支持自动响应任何规模的代码执行请求，从每天十几个事件到每秒数百个事件。有关更多信息，请参阅 自动连接 Lambda 函数和数据库实例。

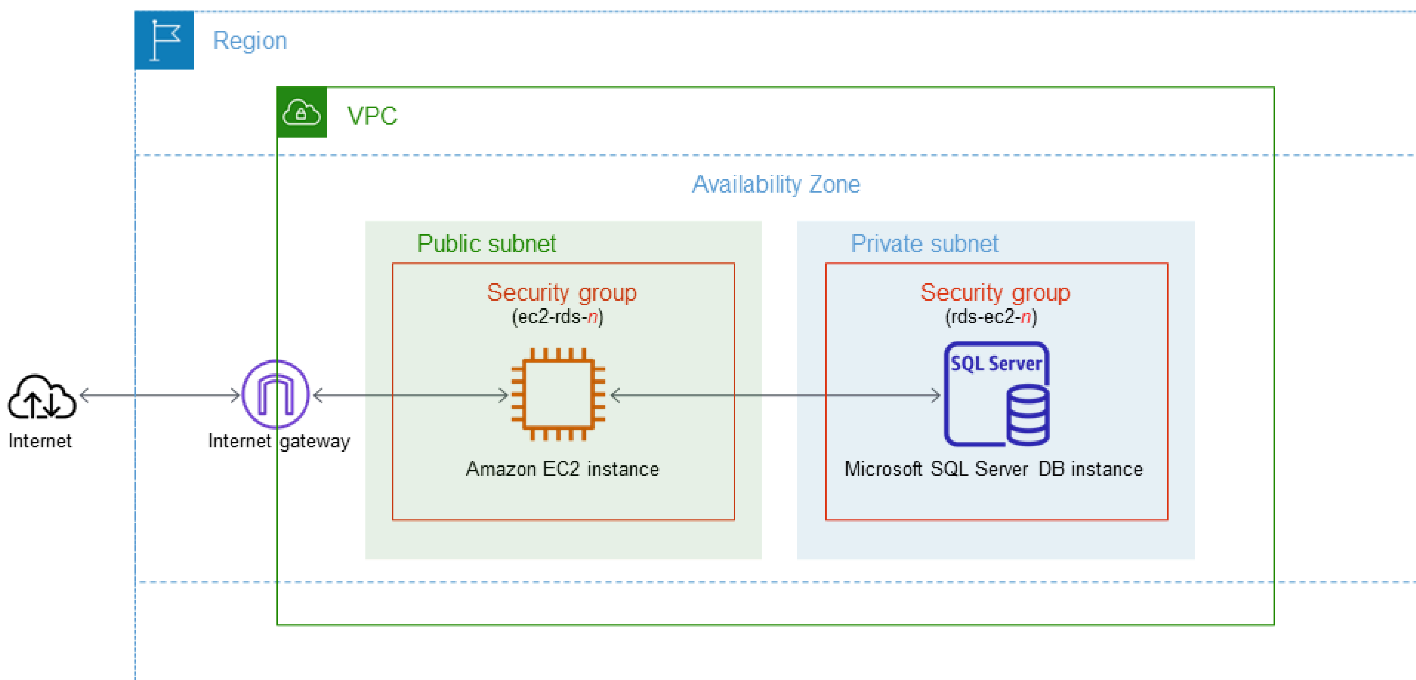
创建 Microsoft SQL Server 数据库实例并连接到该实例

本教程创建一个 EC2 实例和一个 RDS for Microsoft SQL Server 数据库实例。本教程向您展示如何使用 Microsoft SQL Server Management Studio 客户端从 EC2 实例访问数据库实例。作为最佳实践，本教程在虚拟私有云（VPC）中创建了私有数据库实例。在大多数情况下，同一 VPC 中的其他资源（例如 EC2 实例）可以访问数据库实例，但 VPC 之外的资源无法访问该实例。

完成本教程后，VPC 的每个可用区中都有一个公有和私有子网。在一个可用区中，EC2 实例在公有子网中，数据库实例在私有子网中。

> **Important**
>
> 创建 Amazon 账户并不会收费；但是，在完成本教程过程中使用 Amazon 资源可能会产生费用。完成本教程后，如果不再需要这些资源，可以将其删除。

下图显示了教程完成时的配置。



本教程可帮助您通过以下方法之一创建资源：

1. 使用 Amazon Web Services Management Console - [创建 SQL Server 数据库实例](#) 和 [创建 EC2 实例](#)

2. 使用 Amazon CloudFormation 创建数据库实例和 EC2 实例 - （可选）使用 Amazon CloudFormation 创建 VPC、EC2 实例和 SQL Server 实例

第一种方法使用轻松创建，通过 Amazon Web Services Management Console 创建私有 SQL Server 数据库实例。您可以仅指定数据库引擎类型、数据库实例大小和数据库实例标识符。轻松创建为其他配置选项使用默认设置。

如果改用标准创建，则在创建数据库实例时可以指定更多配置选项。这些选项包括可用性、安全性、备份和维护设置。要创建公有数据库实例，必须使用标准创建。有关信息，请参阅创建 Amazon RDS 数据库实例。

主题

先决条件

在开始之前，请完成以下各节中的步骤：

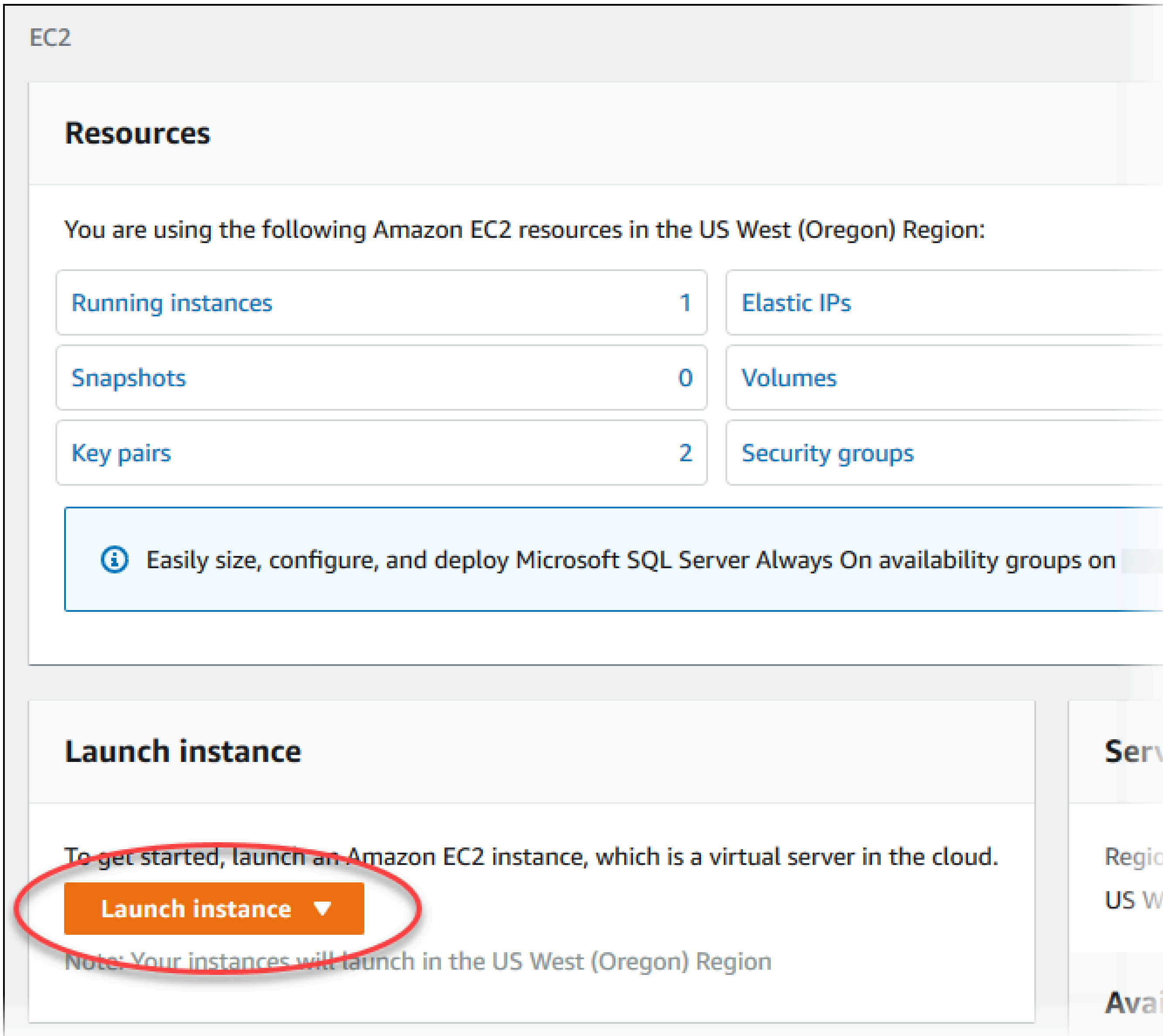
- 注册 Amazon Web Services 账户
- 保护 IAM 用户

创建 EC2 实例

创建将用于连接到数据库的 Amazon EC2 实例。

创建 EC2 实例

1. 登录到 Amazon Web Services Management Console 并打开 Amazon EC2 控制台（https://console.aws.amazon.com/ec2/）。
2. 在 Amazon Web Services Management Console的右上角，选择之前用于数据库的 Amazon Web Services 区域。
3. 选择 EC2 控制面板，然后选择启动实例，如下图所示。

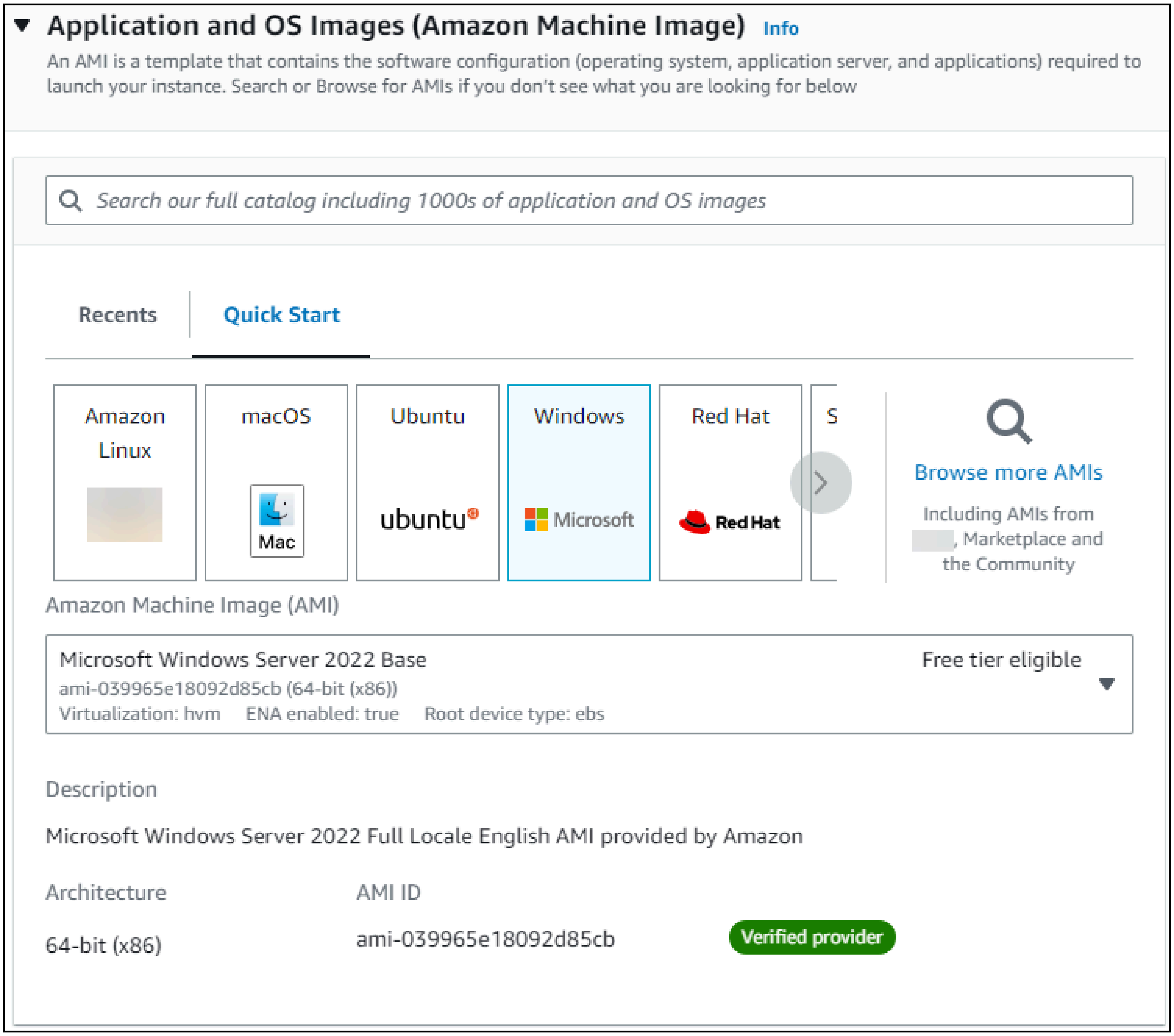


启动实例页面打开。

4. 在启动实例页面上选择以下设置。

a. 在 Name and tags（名称和标签）下，对于 Name（名称），输入 **ec2-database-connect**。

b. 在应用程序和操作系统映像（亚马逊云机器镜像）下，选择 Windows，然后选择 Microsoft Windows Server 2022 Base。对于其他选项，保留默认选择。



c. 在 Instance type（实例类型）下，选择 t2.micro。

d. 在 Key pair (login) [密钥对（登录）] 下，选择 Key pair name（密钥对名称）以使用现有密钥对。要为 Amazon EC2 实例创建新的密钥对，请选择 Create new key pair（创建新的密钥对），然后使用 Create key pair（创建密钥对）窗口来创建它。

有关创建新的密钥对的更多信息，请参阅《适用于 Windows 实例的 Amazon EC2 用户指南》中的创建密钥对。

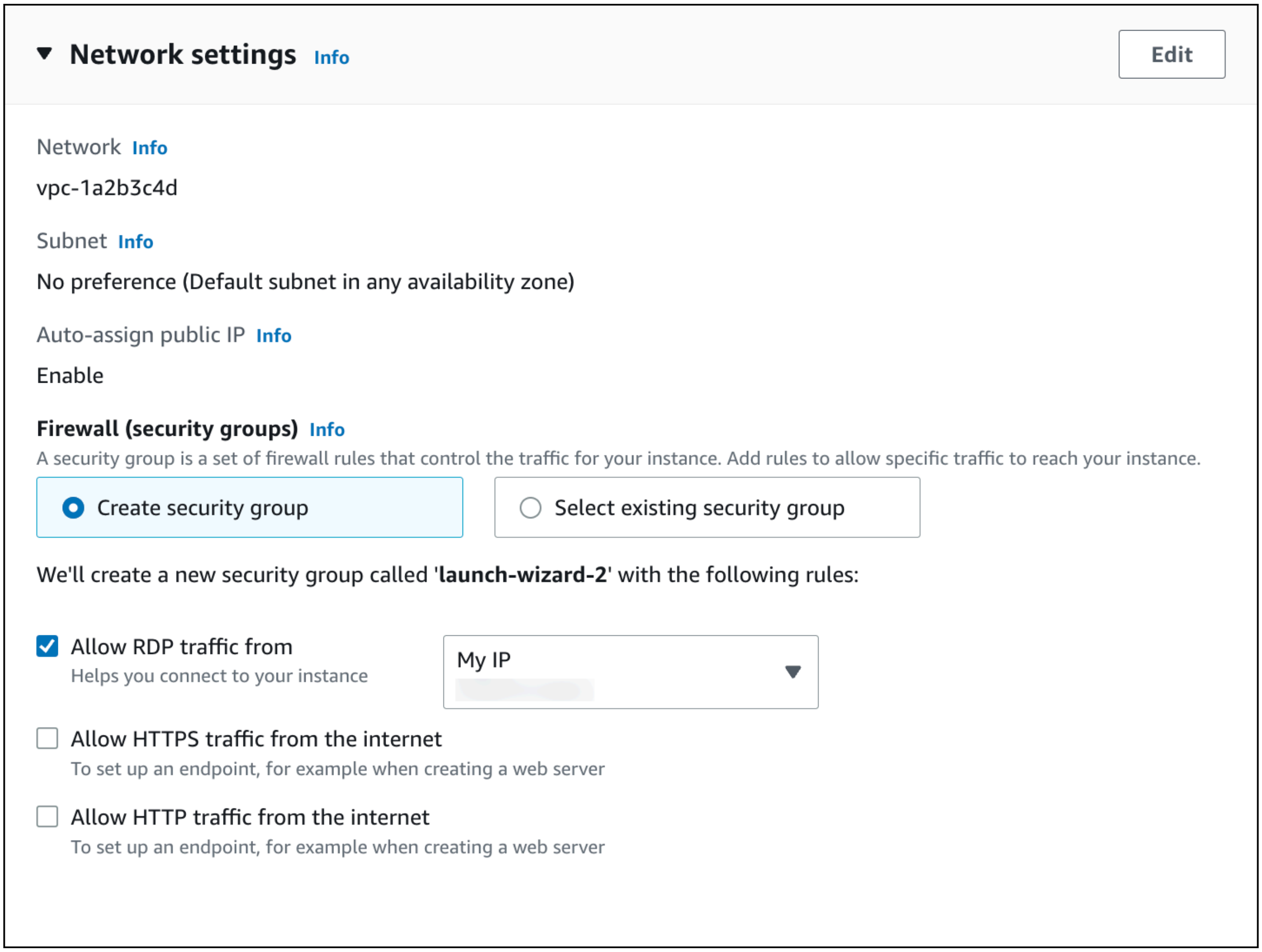
e. 对于网络设置中的防火墙（安全组），选择允许 RDP 流量来自以连接到 EC2 实例。

如果显示的 IP 地址对于 RDP 连接是正确的，您可以选择我的 IP。否则，您可以确定要用来通过 RDP 连接到 VPC 中的 EC2 实例的 IP 地址。要确定您的公有 IP 地址，请在新的浏览器窗口或标签页中，使用 https://checkip.amazonaws.com 上的服务。IP 地址的示例为 192.0.2.1/32。

在许多情况下，您可能通过互联网服务提供商（ISP）进行连接，或者在不使用静态 IP 地址的情况下从防火墙之后进行连接。如果是这样，请确保确定客户端计算机使用的 IP 地址范围。

> **Warning**
>
> 如果您使用 `0.0.0.0/0` 进行 RDP 访问，则所有 IP 地址可能能够使用 RDP 访问您的公有 EC2 实例。在测试环境下短时间内，此方法尚可接受，但它对于生产环境并不安全。在生产环境中，将仅向特定 IP 地址或地址范围授权使用 RDP 访问您的 EC2 实例。

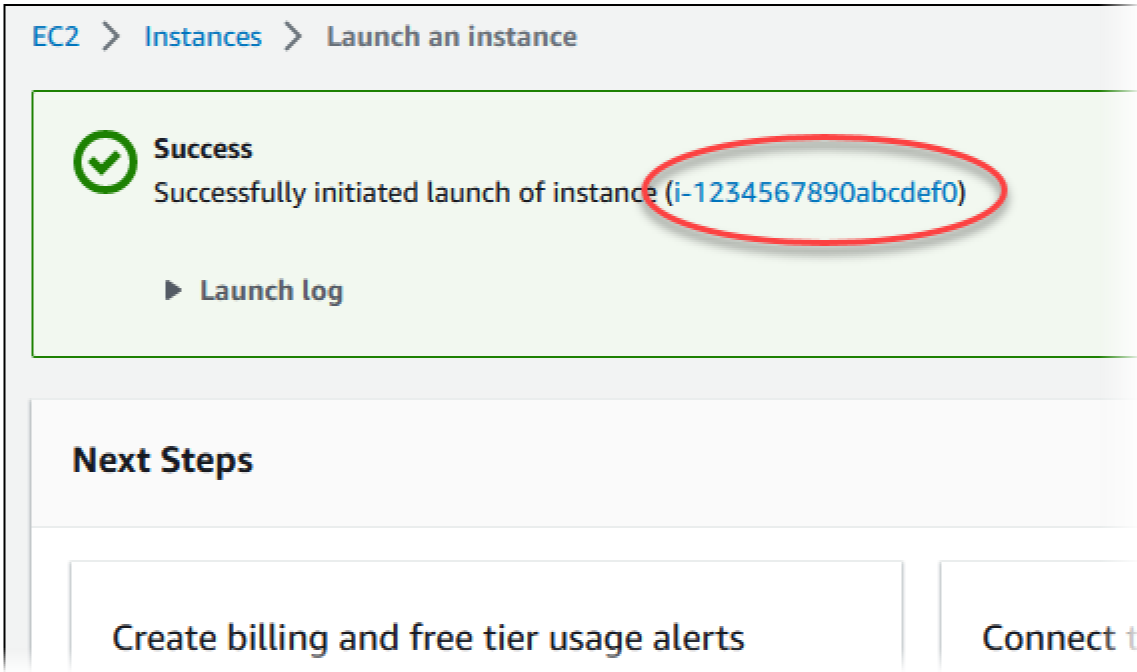
下图显示了网络设置部分的示例。



f. 对于其余部分保留原定设置值。

g. 查看摘要面板中您的 EC2 实例配置的摘要，当您准备好后，选择启动实例。

5. 在启动状态页面上，记下新 EC2 实例的标识符，例如：`i-1234567890abcdef0`。



6. 选择 EC2 实例标识符以打开 EC2 实例的列表。
7. 等待 EC2 实例的实例状态变为正在运行，然后继续。

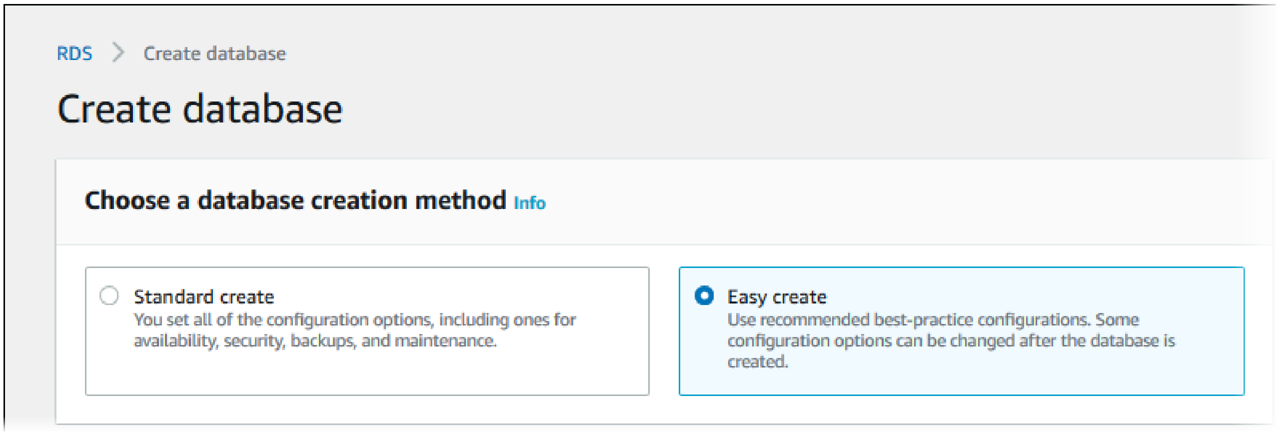
创建 SQL Server 数据库实例

Amazon RDS 的基本构建基块是数据库实例。这是您将要在其中运行 SQL Server 数据库的环境。

在该示例中，您使用轻松创建来创建一个数据库实例，它运行带有 db.t2.micro 数据库实例类的 SQL Server 数据库引擎。

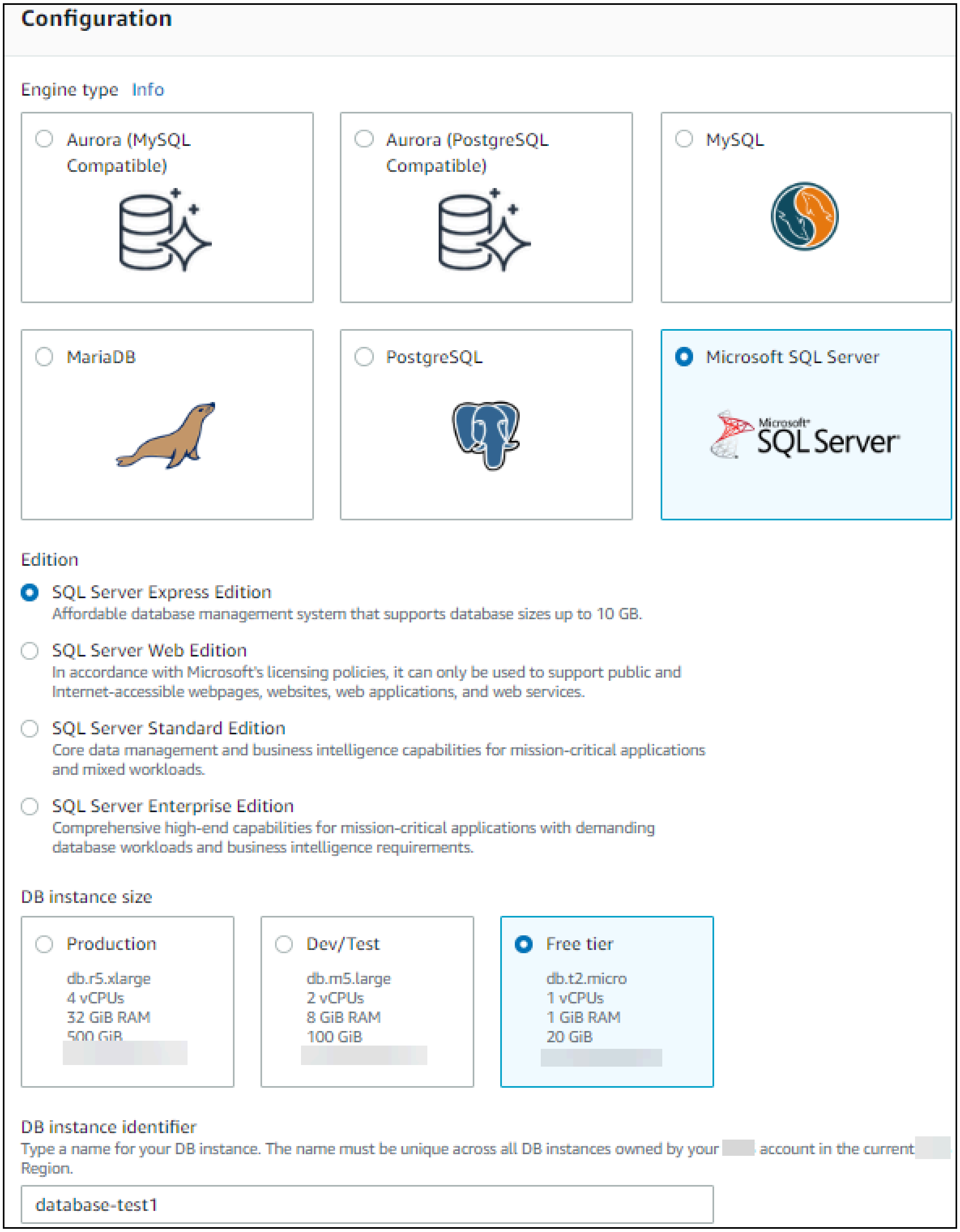
使用 Easy create (轻松创建) 创建 Microsoft SQL Server 数据库实例

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在 Amazon RDS 控制台的右上角，选择要在其中创建数据库实例的 Amazon Web Services 区域。
3. 在导航窗格中，选择 Databases (数据库)。
4. 选择 Create database (创建数据库)，并确保已选择 Easy create (轻松创建)。



5. 在配置中，选择 Microsoft SQL Server。
6. 对于版本，选择 SQL Server Express Edition。
7. 对于 数据库实例大小，选择 免费套餐。
8. 对于 DB instance identifier（数据库实例标识符），输入 **`database-test1`**。

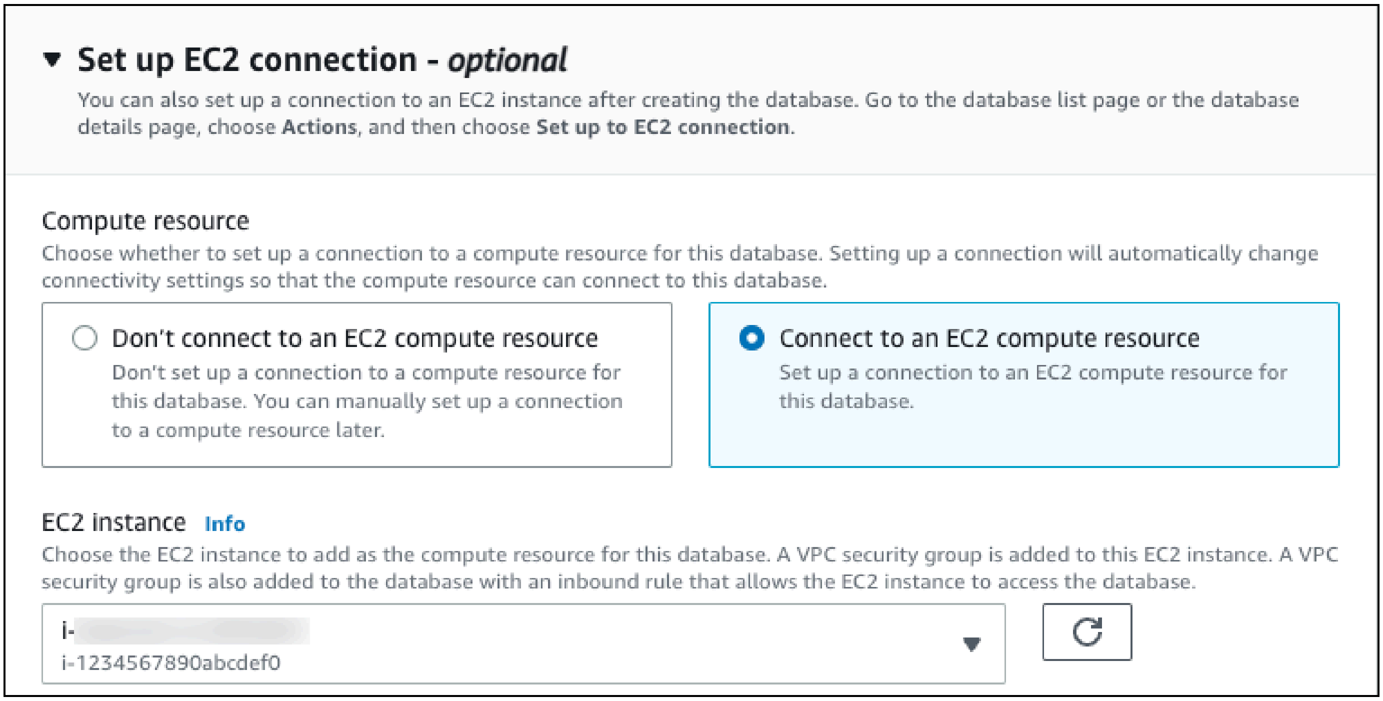
   创建数据库页面应类似于以下图像。



9. 对于主用户名，输入主用户的名称，或者保留原定设置名称。

10. 要设置与您之前创建的 EC2 实例的连接，请打开设置 EC2 连接 - 可选。

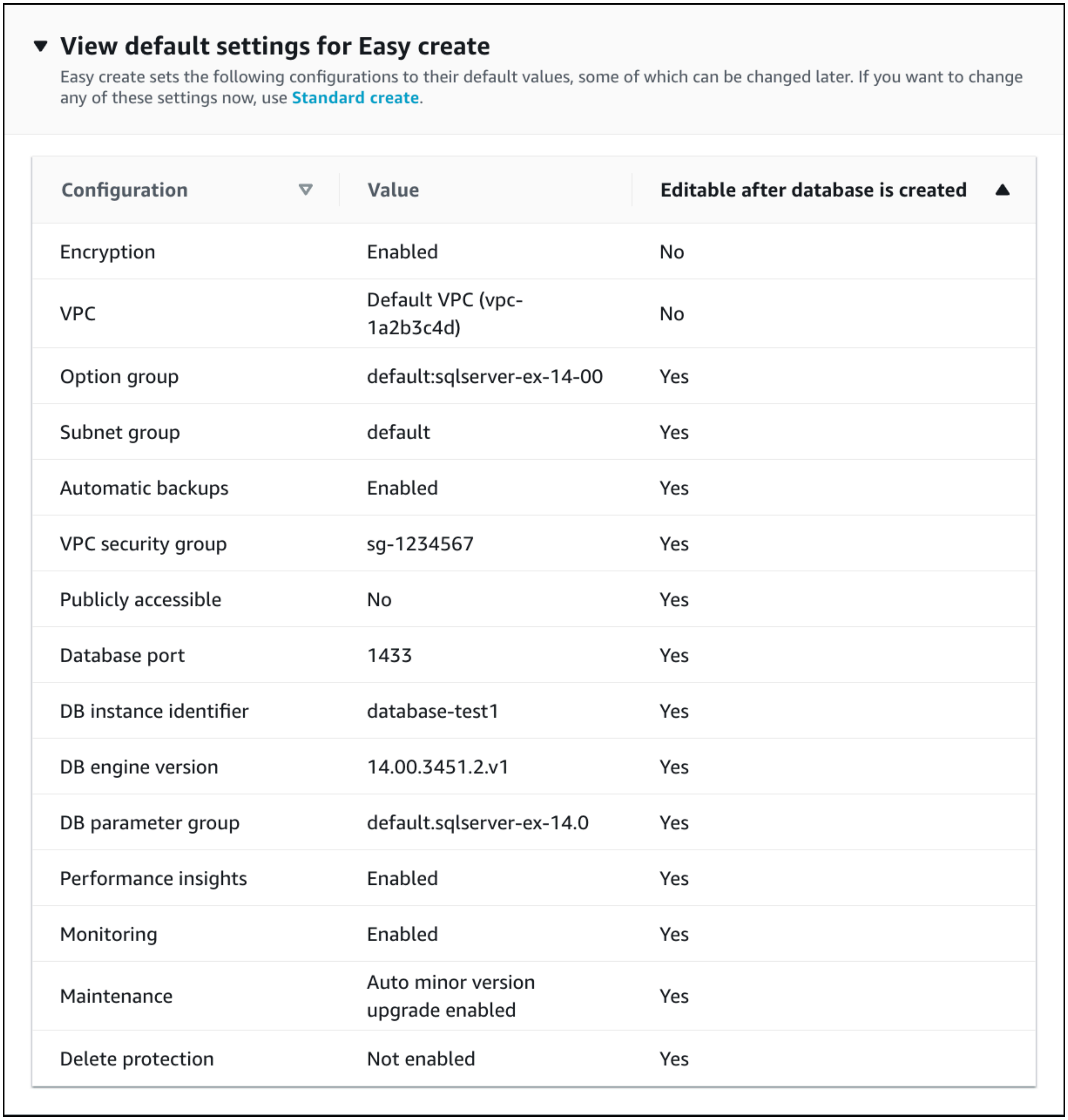
    选择连接到 EC2 计算资源。选择您之前创建的 EC2 实例。



11. 要为数据库实例使用自动生成的主密码，请选择 Auto generate a password (自动生成密码) 框。

    要输入主密码，请清除 Auto generate a password (自动生成密码) 框，然后在 Master password (主密码) 和 Confirm password (确认密码) 中输入相同的密码。

12. 打开查看轻松创建的默认设置。



View default settings for Easy create

Easy create sets the following configurations to their default values, some of which can be changed later. If you want to change any of these settings now, use **Standard create**.

Configuration	Value	Editable after database is created
Encryption	Enabled	No
VPC	Default VPC (vpc-1a2b3c4d)	No
Option group	default:sqlserver-ex-14-00	Yes
Subnet group	default	Yes
Automatic backups	Enabled	Yes
VPC security group	sg-1234567	Yes
Publicly accessible	No	Yes
Database port	1433	Yes
DB instance identifier	database-test1	Yes
DB engine version	14.00.3451.2.v1	Yes
DB parameter group	default.sqlserver-ex-14.0	Yes
Performance insights	Enabled	Yes
Monitoring	Enabled	Yes
Maintenance	Auto minor version upgrade enabled	Yes
Delete protection	Not enabled	Yes

您可以检查在启用 Easy create (轻松创建) 时使用的默认设置。可在创建数据库后编辑列显示在创建数据库后可以更改的选项。

- 如果该列中的某个设置为否，而您想要不同的设置，则可以使用标准创建来创建数据库实例。

- 如果某个设置在该列中为是，您可以使用标准创建来创建数据库实例，也可以在创建后修改数据库实例以更改该设置。

13. 选择创建数据库。

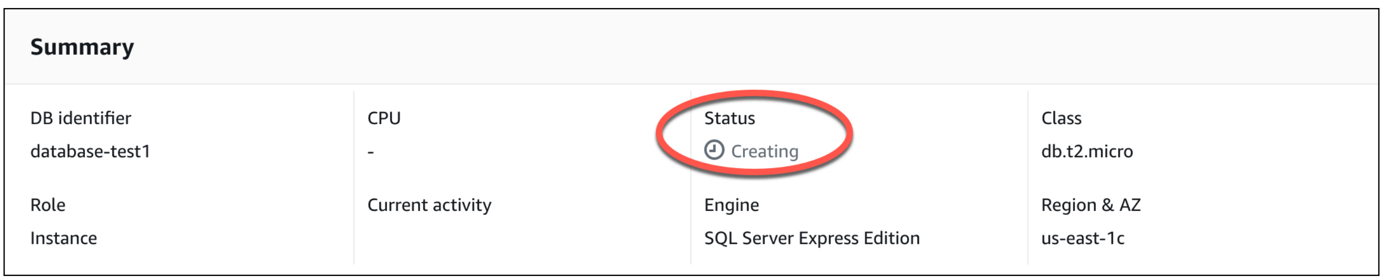
    要查看数据库实例的主用户名和密码，请选择查看凭证详细信息。

    您可以使用显示的用户名和密码，以主用户身份连接到数据库实例。

> **Important**
>
> 您无法再次查看主用户密码。如果您不记录它，您可能需要更改它。
> 如果需要在数据库实例可用后更改主用户密码，则可以修改数据库实例以执行此操作。有关修改 数据库实例的更多信息，请参阅 修改 Amazon RDS 数据库实例。

14. 在数据库列表中，选择新的 SQL Server 数据库实例的名称以显示其详细信息。

    数据库实例具有正在创建状态，直到此实例就绪可供使用。



    当状态变为 Available（可用）时，您便可以连接到该数据库实例。根据数据库实例类和存储量，新实例可能需要等待 20 分钟时间才可用。

（可选）使用 Amazon CloudFormation 创建 VPC、EC2 实例和 SQL Server 实例

您可以使用 Amazon CloudFormation 通过将基础设施视为代码来配置 Amazon 资源，而无需使用控制台创建 VPC、EC2 实例和 SQL Server 实例。为了帮助您将 Amazon 资源组织成更小、更易于管理的单元，您可以使用 Amazon CloudFormation 嵌套堆栈功能。有关更多信息，请参阅在 Amazon CloudFormation 控制台上创建堆栈和使用嵌套堆栈。

⚠ Important

Amazon CloudFormation 是免费的，但 CloudFormation 创建的资源是实时的。您需要为这些资源支付标准使用费，直到您终止使用它们为止。总费用将应该最少的。有关如何最大限度降低费用的信息，请转至 Amazon Free Tier。

要使用 Amazon CloudFormation 控制台创建资源，请执行以下步骤：

- 下载 CloudFormation 模板
- 使用 CloudFormation 配置资源

下载 CloudFormation 模板

CloudFormation 模板是一个 JSON 或 YAML 文本文件，其中包含有关您希望在堆栈中创建的资源的配置信息。此模板还为您创建 VPC 和堡垒主机以及 RDS 实例。

要下载该模板文件，请打开以下链接 SQL Server CloudFormation 模板。

在 Github 页面中，单击下载原始文件按钮以保存模板 YAML 文件。

使用 CloudFormation 配置资源

ⓘ Note

在开始此过程之前，请确保您的 Amazon Web Services 账户中具有 EC2 实例密钥对。有关更多信息，请参阅 Amazon EC2 密钥对和 Linux 实例。

使用 Amazon CloudFormation 模板时，必须选择正确的参数以确保正确创建资源。按以下步骤操作：

1. 登录到 Amazon Web Services Management Console 并打开 Amazon CloudFormation 控制台 https://console.aws.amazon.com/cloudformation。
2. 选择创建堆栈。
3. 在“指定模板”部分，选择从您的计算机上传模板文件，然后选择下一步。
4. 在指定堆栈详细信息页面上，设置以下参数：
   a. 将堆栈名称设置为 SQLServerTestStack。
   b. 在参数下，通过选择三个可用区来设置可用区。

c. 在 Linux 堡垒主机配置下，在密钥名称中，选择用于登录您的 EC2 实例的密钥对。

d. 在 Linux 堡垒主机配置设置中，将允许的 IP 范围设置为您的 IP 地址。要使用 Secure Shell（SSH）连接到 VPC 中的 EC2 实例，请通过 https://checkip.amazonaws.com 上的服务确定您的公有 IP 地址。IP 地址的示例为 192.0.2.1/32。

> **Warning**
>
> 如果您使用 `0.0.0.0/0` 进行 SSH 访问，则所有 IP 地址可能能够使用 SSH 访问您的公有 EC2 实例。在测试环境下短时间内，此方法尚可接受，但它对于生产环境并不安全。在生产环境中，将仅向特定 IP 地址或地址范围授权使用 SSH 访问您的 EC2 实例。

e. 在数据库常规配置下，将数据库实例类设置为 db.t3.micro。

f. 将数据库名称设置为 **`database-test1`**。

g. 在数据库主用户名中，输入主用户的名称。

h. 在本教程中，将使用 Secrets Manager 管理数据库主用户密码设置为 `false`。

i. 在数据库密码中，设置所选密码。请记住此密码以便在教程中的后续步骤中使用。

j. 在数据库存储配置下，将数据库存储类型设置为 gp2。

k. 在数据库监控配置下，将启用 RDS 性能详情设置为 false。

l. 将所有其他设置保留为默认值。单击下一步继续。

5. 在配置堆栈选项页面中，保留所有默认选项。单击下一步继续。
6. 在查看堆栈页面中，在检查数据库和 Linux 堡垒主机选项后，选择提交。

堆栈创建过程完成后，查看名为 BastionStack 和 RDSNS 的堆栈，记下连接到数据库所需的信息。有关更多信息，请参阅在 Amazon Web Services Management Console 上查看 Amazon CloudFormation 堆栈数据和资源。

连接到 SQL Server 数据库实例

在以下过程中，您将使用 Microsoft SQL Server Management Studio（SSMS）连接数据库实例。

使用 SSMS 连接到 RDS for SQL Server 数据库实例

1. 找到您的数据库实例的端点（DNS 名称）和端口号。

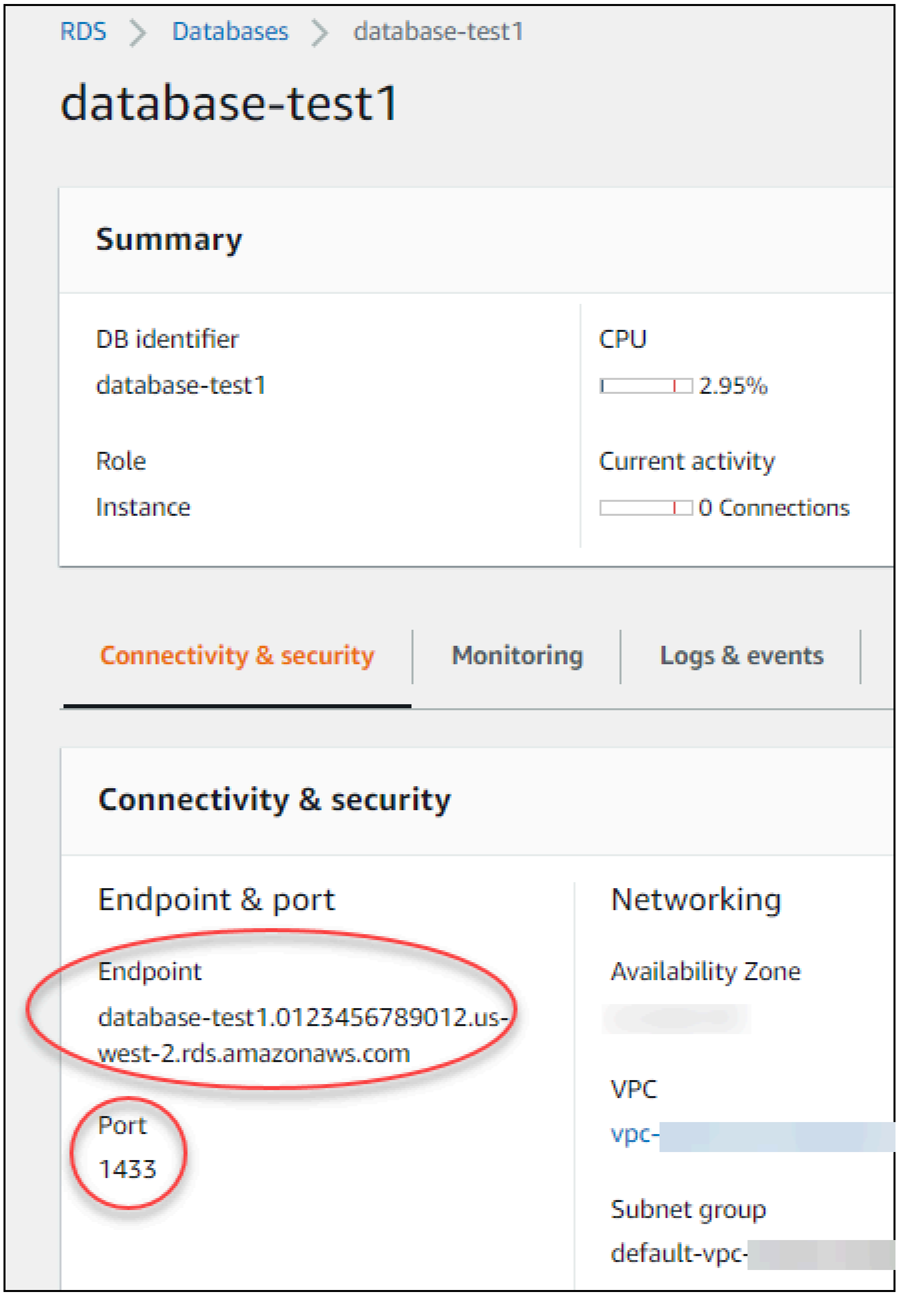
   a. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。

b. 在 Amazon RDS 控制台的右上角，选择数据库实例的 Amazon Web Services 区域。

c. 在导航窗格中，选择 Databases（数据库）。

d. 选择 SQL Server 数据库实例名称以显示其详细信息。

e. 在 Connectivity (连接) 选项卡上，复制端点。另请注意端口号。您需要端点和端口号才能连接到数据库实例。



2. 按照《适用于 Windows 实例的 Amazon EC2 用户指南》的[连接到您的 Microsoft Windows 实例]中的步骤，连接到您之前创建的 EC2 实例。

3. 安装 Microsoft 的 SQL Server Management Studio（SSMS）客户端。

要将 SSMS 的独立版本下载到您的 EC2 实例，请参阅 Microsoft 文档中的下载 SQL Server Management Studio（SSMS）。

a. 使用“开始”菜单打开 Internet Explorer。

b. 使用 Internet Explorer 下载并安装 SSMS 的独立版本。如果系统提示您该站点不可信，请将该站点添加到可信站点列表中。

4. 启动 SQL Server Management Studio（SSMS）。

   显示连接至服务器对话框。

5. 为您的示例数据库实例提供以下信息：

   a. 对于 Server type，选择 Database Engine。

   b. 对于 Server name (服务器名称)，请输入 DNS 名称，后跟逗号和端口号（默认端口为 1433）。例如，您的服务器名称应如下所示：

```
database-test1.0123456789012.us-west-2.rds.amazonaws.com,1433
```

   c. 对于 Authentication，选择 SQL Server Authentication。

   d. 对于登录名，输入您选择用于示例数据库实例的用户名。这也称作主用户名。

   e. 对于 Password (密码)，输入您之前为示例数据库实例选择的密码。这也称作主用户密码。

6. 选择 Connect。

   稍等几分钟后，SSMS 将连接到数据库实例。出于安全考虑，最佳做法是使用加密连接。仅当客户端和服务器位于同一 VPC 中，并且网络受信任时，才会使用未加密的 SQL Server 连接。有关使用加密连接的信息，请参阅将 SSL 与 Microsoft SQL Server 数据库实例结合使用

有关连接到 Microsoft SQL Server 数据库实例的更多信息，请参阅连接到 Microsoft SQL Server 数据库实例。

有关连接问题的信息，请参阅无法连接到 Amazon RDS 数据库实例。

探索示例 SQL Server 数据库实例

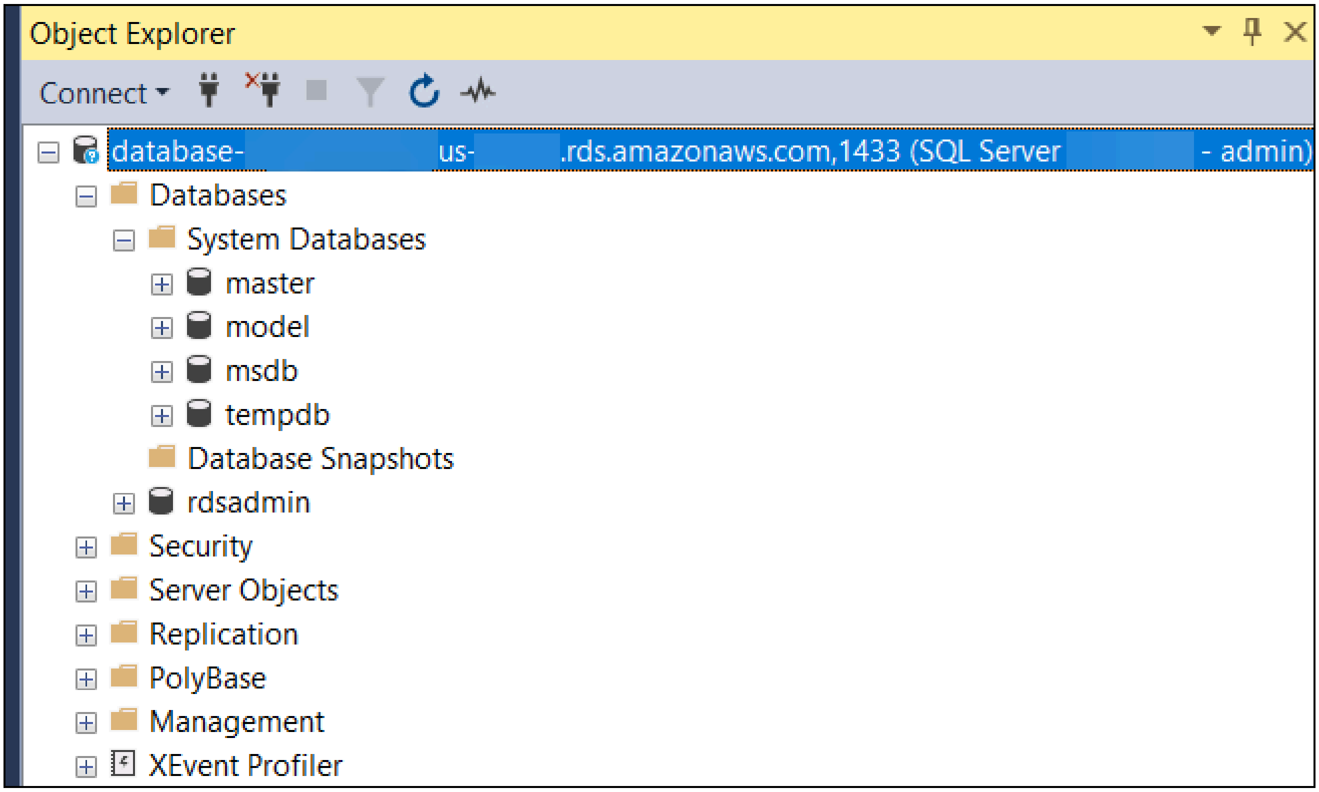
您可以使用 Microsoft SQL Server Management Studio（SSMS）探索您的示例数据库实例。

使用 SSMS 探索数据库实例

1. 您的 SQL Server 数据库实例 附带 SQL Server 的标准内置系统数据库 (master、model、msdb 和 tempdb)。要了解系统数据库，请执行以下操作：

   a. 在 SSMS 的 View 菜单上，选择 Object Explorer。

   b. 展开数据库实例，展开 Databases (数据库)，然后展开 System Databases (系统数据库)，如下所示。



   您的 SQL Server 数据库实例还附带一个名为 `rdsadmin` 的数据库。Amazon RDS 使用此数据库存储它用于管理您的数据库的对象。`rdsadmin` 数据库还包括存储过程，您可以运行它们来执行高级任务。

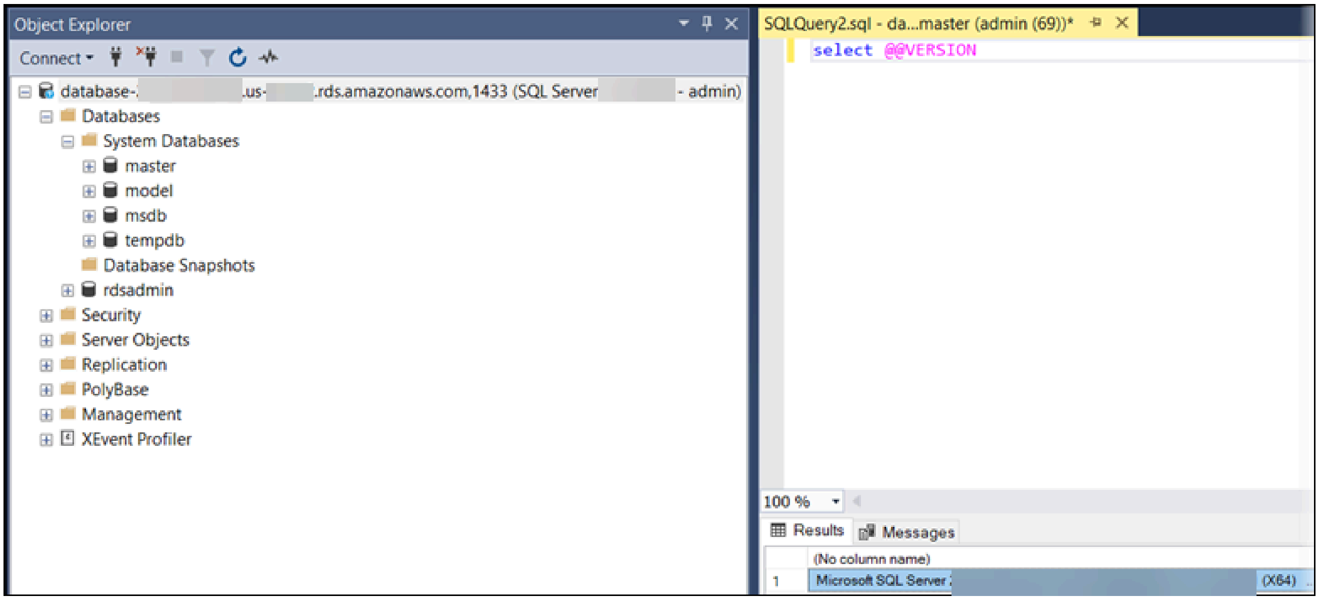
2. 开始创建您自己的数据库并像往常一样对数据库实例和数据库运行查询。要对示例数据库实例运行测试查询，请执行以下操作：

   a. 在 SSMS 中，在文件菜单上，指向新建，然后选择使用当前连接进行查询。

   b. 输入以下 SQL 查询：

```
select @@VERSION
```

c. 运行查询。SSMS 会返回 SQL Server 版的 Amazon RDS 数据库实例。



删除 EC2 实例和数据库实例

在连接到并浏览您创建的示例 EC2 实例和数据库实例之后，删除它们，以便不再为其付费。

如果您使用 Amazon CloudFormation 创建了资源，请跳过此步骤，转至下一步。

删除 EC2 实例

1. 登录到 Amazon Web Services Management Console 并打开 Amazon EC2 控制台（https://console.aws.amazon.com/ec2/）。
2. 在导航窗格中，选择实例。
3. 选择 EC2 实例，然后依次选择实例状态、终止实例。
4. 当系统提示您确认时，选择终止。

有关删除 EC2 实例的更多信息，请参阅《Windows 实例的用户指南》中的终止实例。

删除无最终数据库快照的数据库实例

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。
3. 选择要删除的数据库实例。

4. 对于 Actions，选择 Delete。
5. 清除是否创建最终快照？和保留自动备份。
6. 完成确认并选择删除。

（可选）删除使用 CloudFormation 创建的 EC2 实例和数据库实例

如果您使用 Amazon CloudFormation 创建了资源，请在连接并浏览 EC2 实例和数据库实例示例之后，删除 CloudFormation 堆栈，以便不再为其付费。

删除 CloudFormation 资源

1. 打开 Amazon CloudFormation 控制台。
2. 在 CloudFormationconsole 的堆栈页面上，选择根堆栈（名称不是 VPCStack、BastionStack 或 RDSNS 的堆栈）。
3. 选择删除。
4. 提示进行确认时，选择删除堆栈。

有关如何删除 CloudFormation 中堆栈的信息，请参阅《Amazon CloudFormation 用户指南》中的在 Amazon CloudFormation 控制台上删除堆栈。

（可选）将您的数据库实例连接到 Lambda 函数

您也可以将您的 RDS for SQL Server 数据库实例连接到 Lambda 无服务器计算资源。Lambda 函数支持在不预调配或管理基础设施的情况下运行代码。Lambda 函数还支持自动响应任何规模的代码执行请求，从每天十几个事件到每秒数百个事件。有关更多信息，请参阅 自动连接 Lambda 函数和数据库实例。

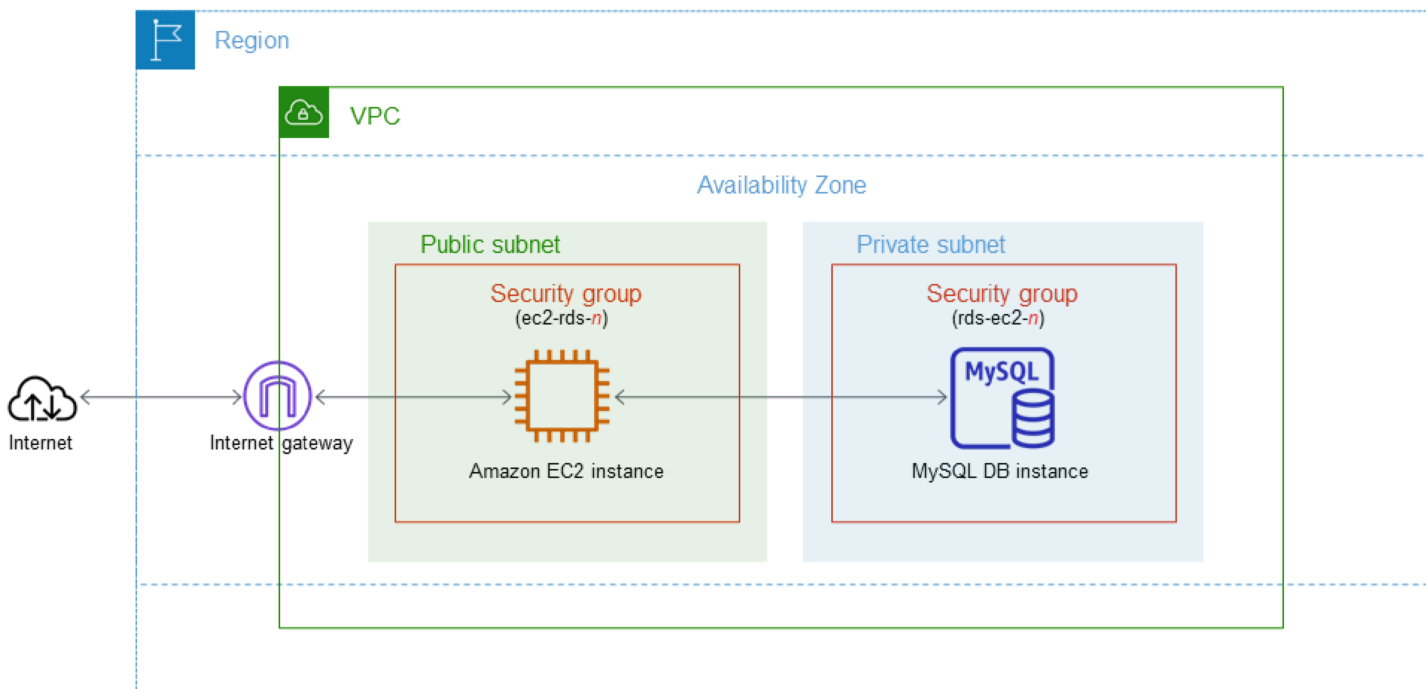
创建并连接到 MySQL 数据库实例

本教程创建一个 EC2 实例和一个 RDS for MySQL 数据库实例。本教程向您展示如何使用标准 MySQL 客户端从 EC2 实例访问数据库实例。作为最佳实践，本教程在虚拟私有云（VPC）中创建了私有数据库实例。在大多数情况下，同一 VPC 中的其他资源（例如 EC2 实例）可以访问数据库实例，但 VPC 之外的资源无法访问该实例。

完成本教程后，VPC 的每个可用区中都有一个公有和私有子网。在一个可用区中，EC2 实例在公有子网中，数据库实例在私有子网中。

> **Important**
>
> 创建 Amazon 账户并不会收费；但是，在完成本教程过程中使用 Amazon 资源可能会产生费用。完成本教程后，如果不再需要这些资源，可以将其删除。

下图显示了教程完成时的配置。



本教程可帮助您通过以下方法之一创建资源：

1. 使用 Amazon Web Services Management Console - 创建 MySQL 数据库实例 和 创建 EC2 实例
2. 使用 Amazon CloudFormation 创建数据库实例和 EC2 实例 - （可选）使用 Amazon CloudFormation 创建 VPC、EC2 实例和 MySQL 实例

第一种方法使用轻松创建，通过 Amazon Web Services Management Console 创建私有 MySQL 数据库实例。您可以仅指定数据库引擎类型、数据库实例大小和数据库实例标识符。轻松创建为其他配置选项使用默认设置。

如果改用标准创建，则在创建数据库实例时可以指定更多配置选项。这些选项包括可用性、安全性、备份和维护设置。要创建公有数据库实例，必须使用标准创建。有关信息，请参阅创建 Amazon RDS 数据库实例。

主题

先决条件

在开始之前，请完成以下各节中的步骤：

- 注册 Amazon Web Services 账户
- 保护 IAM 用户

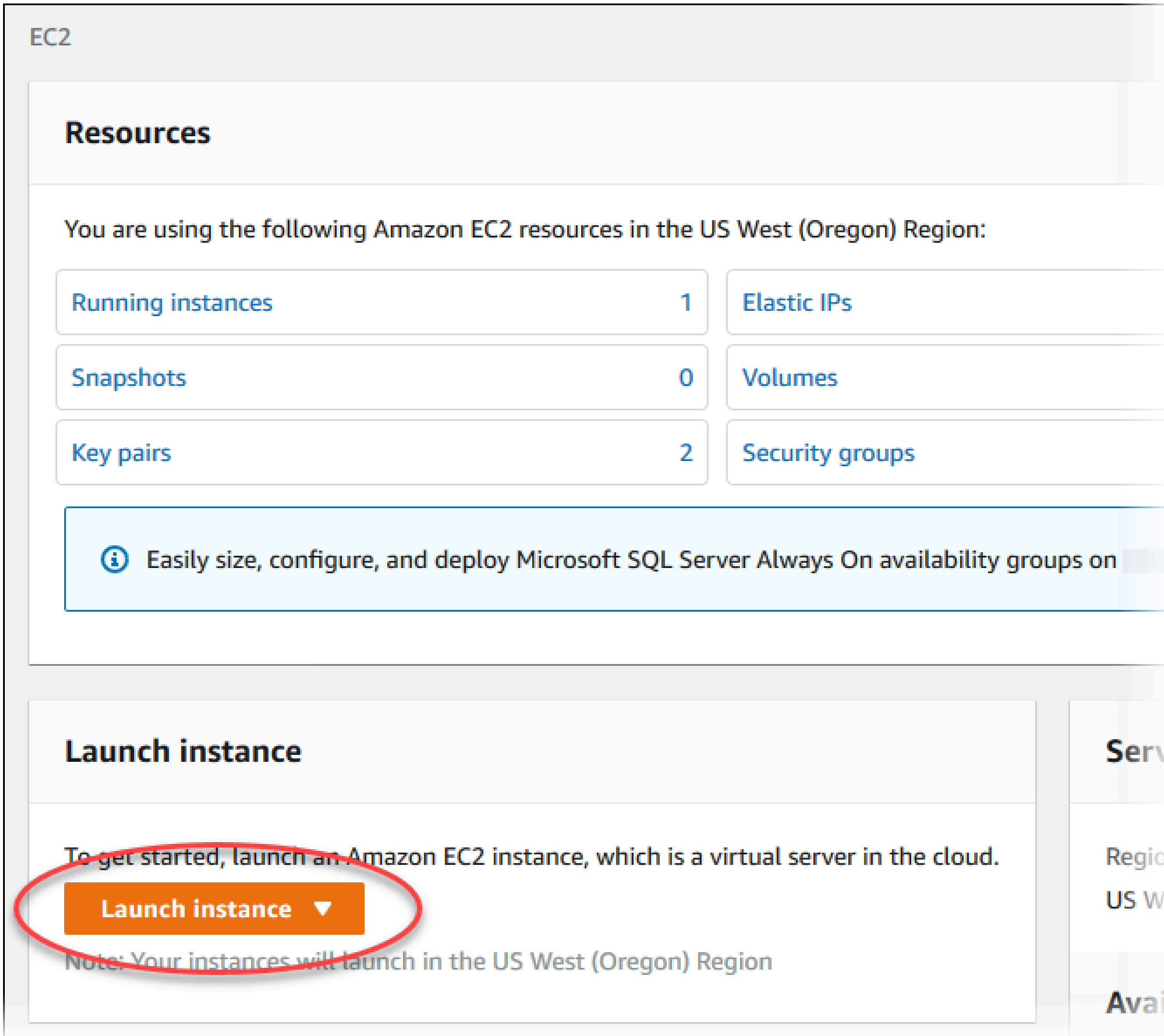
创建 EC2 实例

创建将用于连接到数据库的 Amazon EC2 实例。

创建 EC2 实例

1. 登录到 Amazon Web Services Management Console 并打开 Amazon EC2 控制台（https://console.aws.amazon.com/ec2/）。
2. 在 Amazon Web Services Management Console的右上角，选择要在其中创建 EC2 实例的 Amazon Web Services 区域。

3. 选择 EC2 控制面板，然后选择启动实例，如下图所示。

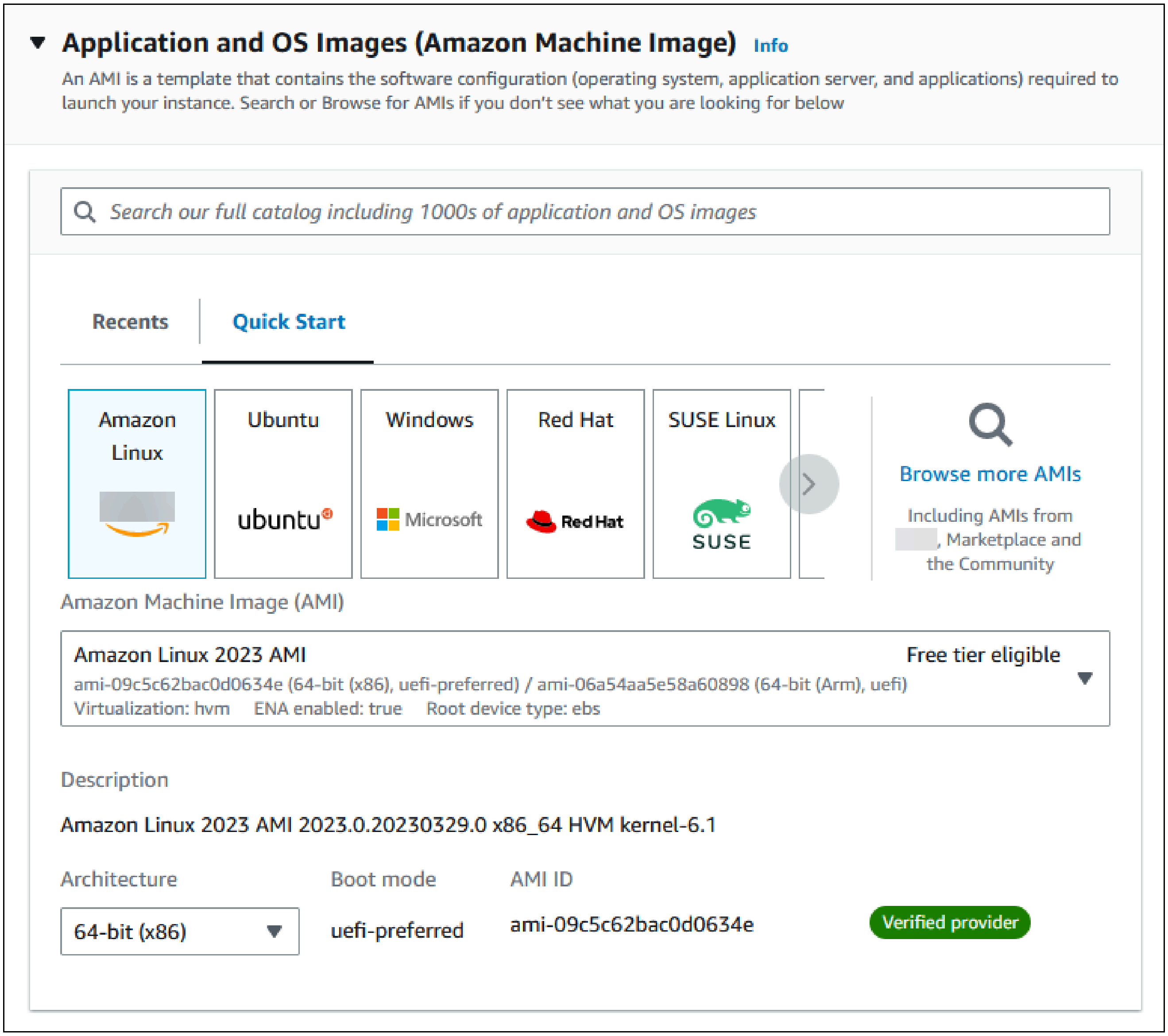


启动实例页面打开。

4. 在启动实例页面上选择以下设置。

   a. 在 Name and tags（名称和标签）下，对于 Name（名称），输入 **`ec2-database-connect`**。

   b. 在应用程序和操作系统映像（Amazon 机器映像）下，选择 Amazon Linux，然后选择 Amazon Linux 2023 AMI。对于其他选项，保留默认选择。



c. 在 Instance type（实例类型）下，选择 t2.micro。

d. 在 Key pair (login) [密钥对（登录）] 下，选择 Key pair name（密钥对名称）以使用现有密钥对。要为 Amazon EC2 实例创建新的密钥对，请选择 Create new key pair（创建新的密钥对），然后使用 Create key pair（创建密钥对）窗口来创建它。

   有关创建新的密钥对的更多信息，请参阅《Amazon EC2 用户指南》中的创建密钥对。

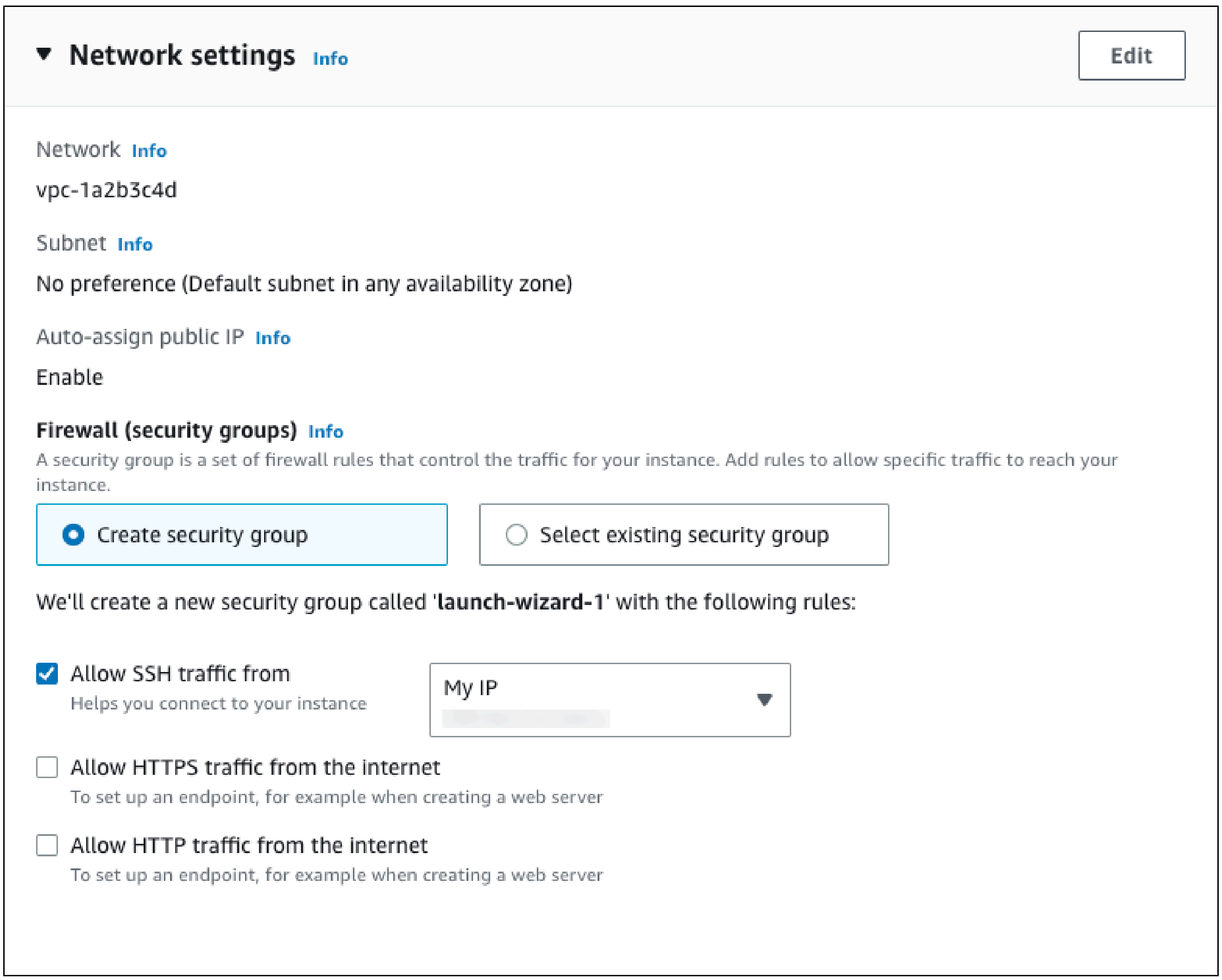
e. 对于网络设置中的允许 SSH 流量，选择 EC2 实例的 SSH 连接来源。

   如果显示的 IP 地址对于 SSH 连接是正确的，您可以选择 My IP（我的 IP）。否则，您可以确定要用来通过 Secure Shell（SSH）连接到 VPC 中的 EC2 实例的 IP 地址。要确定您的公有 IP 地址，请在新的浏览器窗口或标签页中，使用 https://checkip.amazonaws.com 上的服务。IP 地址的示例为 192.0.2.1/32。

在许多情况下，您可能通过互联网服务提供商（ISP）进行连接，或者在不使用静态 IP 地址的情况下从防火墙之后进行连接。如果是这样，请确保确定客户端计算机使用的 IP 地址范围。

> **Warning**
>
> 如果您使用 `0.0.0.0/0` 进行 SSH 访问，则所有 IP 地址可能能够使用 SSH 访问您的公有 EC2 实例。在测试环境下短时间内，此方法尚可接受，但它对于生产环境并不安全。在生产环境中，将仅向特定 IP 地址或地址范围授权使用 SSH 访问您的 EC2 实例。

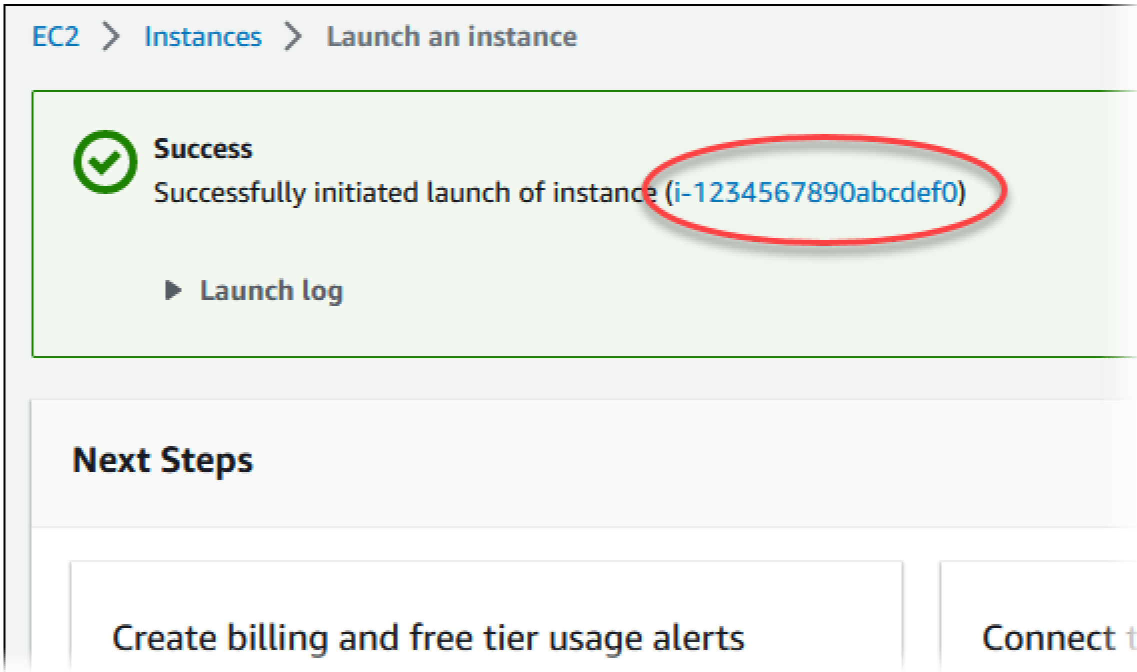
下图显示了网络设置部分的示例。



f. 对于其余部分保留默认值。

g. 查看摘要面板中您的 EC2 实例配置的摘要，当您准备好后，选择启动实例。

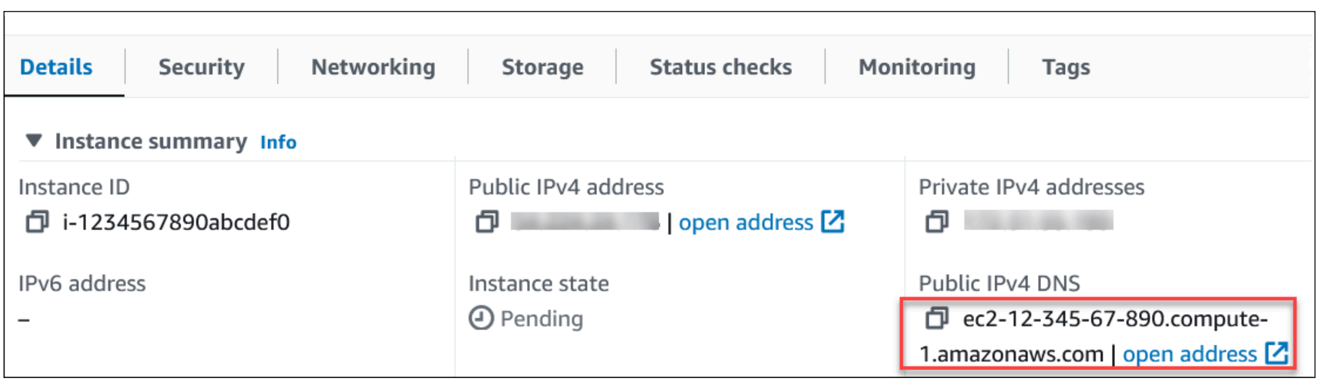
5. 在启动状态页面上，记下新 EC2 实例的标识符，例如：`i-1234567890abcdef0`。



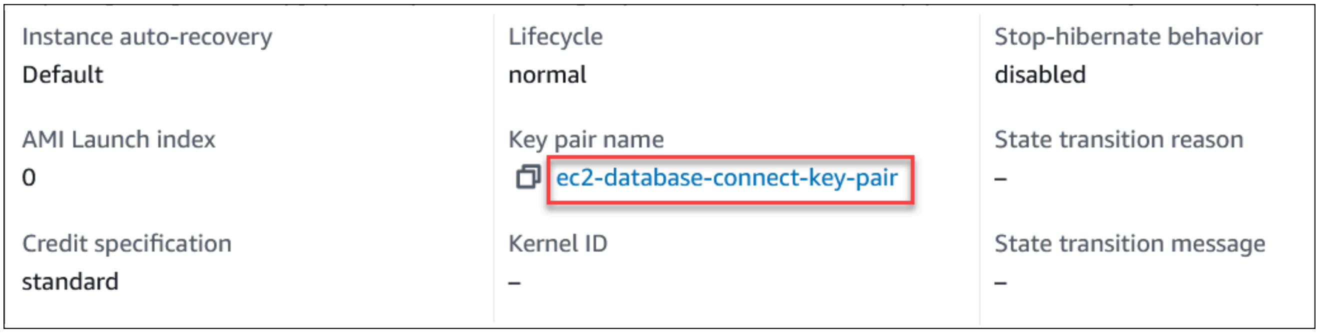
6. 选择 EC2 实例标识符以打开 EC2 实例列表，然后选择您的 EC2 实例。

7. 在详细信息选项卡中，记下使用 SSH 进行连接时所需的以下值：

a. 在实例摘要中，记下公有 IPv4 DNS 的值。



b. 在实例详细信息中，记下密钥对名称的值。



8. 等待 EC2 实例的实例状态变为正在运行，然后继续。

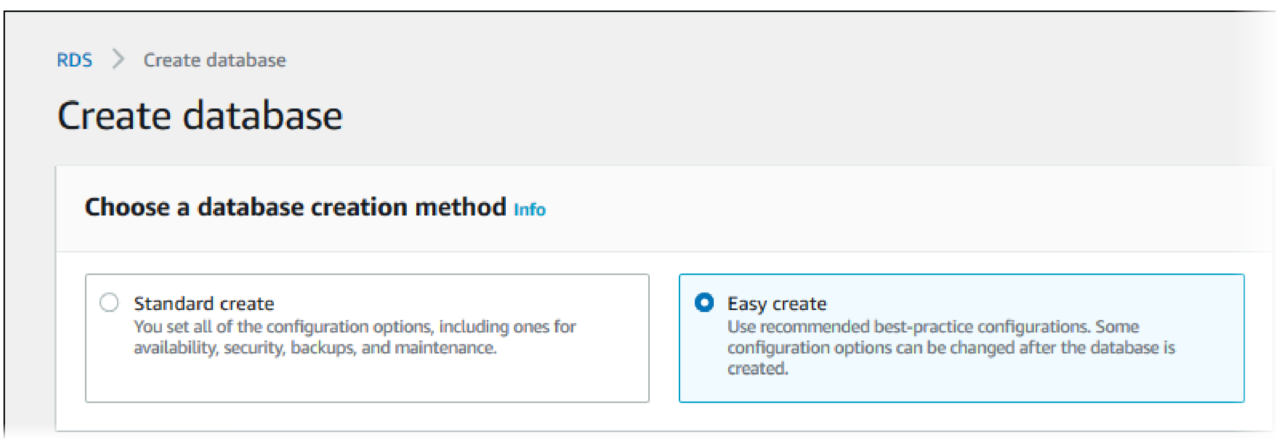
创建 MySQL 数据库实例

Amazon RDS 的基本构建基块是数据库实例。这是您将要在其中运行 MySQL 数据库的环境。

在该示例中，您使用轻松创建创建一个数据库实例，它运行带有 db.t3.micro 数据库实例类的 MySQL 数据库引擎。

在启用“Easy create (轻松创建)”的情况下创建 MySQL 数据库实例

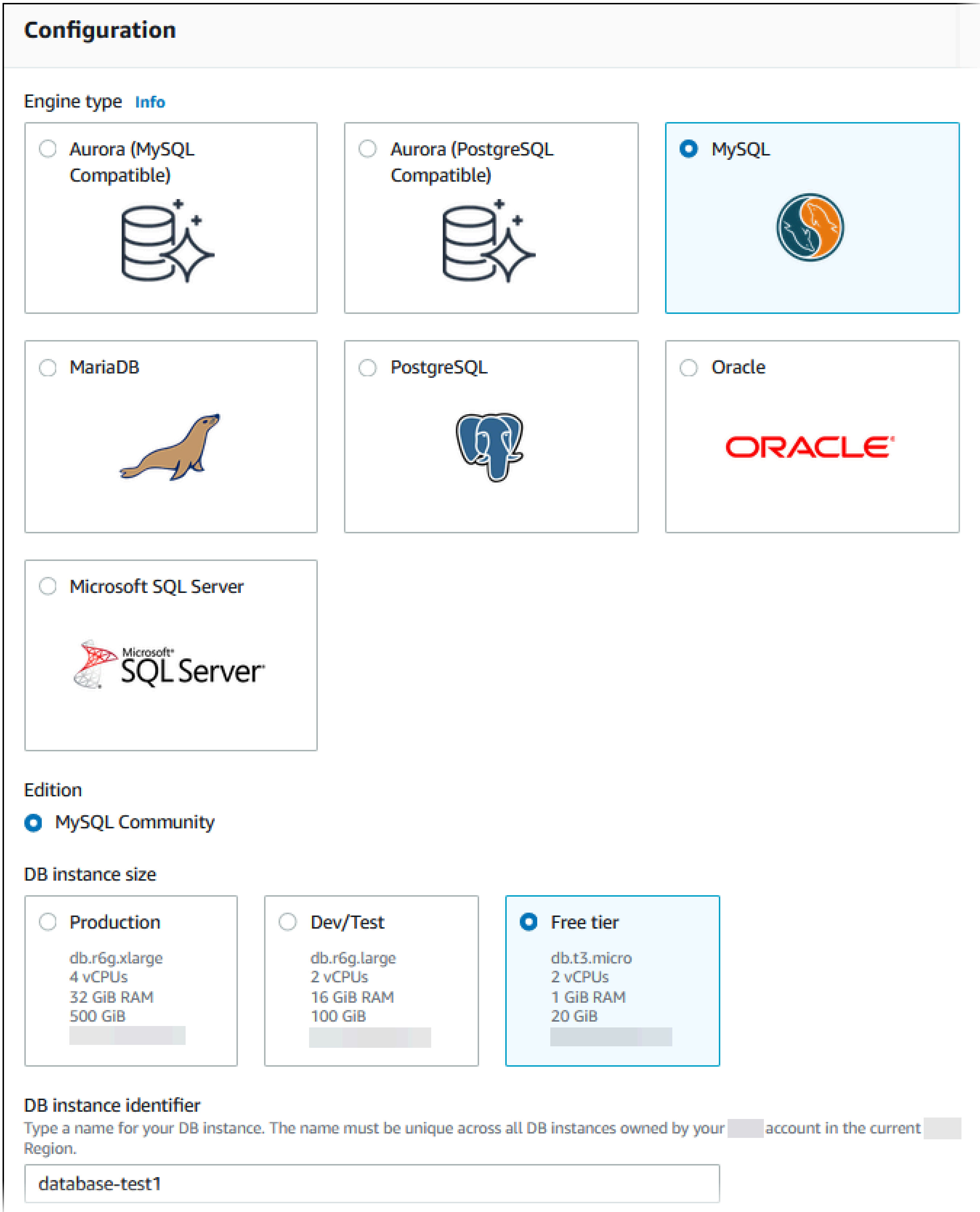
1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在 Amazon RDS 控制台的右上角，选择以前用于 EC2 实例的 Amazon Web Services 区域。
3. 在导航窗格中，选择 Databases (数据库)。
4. 选择 Create database (创建数据库)，并确保已选择 Easy create (轻松创建)。



5. 在配置中，选择 MySQL。
6. 对于数据库实例大小，选择免费套餐。
7. 对于 DB instance identifier（数据库实例标识符），输入 **database-test1**。

8. 对于主用户名，输入主用户的名称，或者保留原定设置名称。

   创建数据库页面应类似于以下图像。

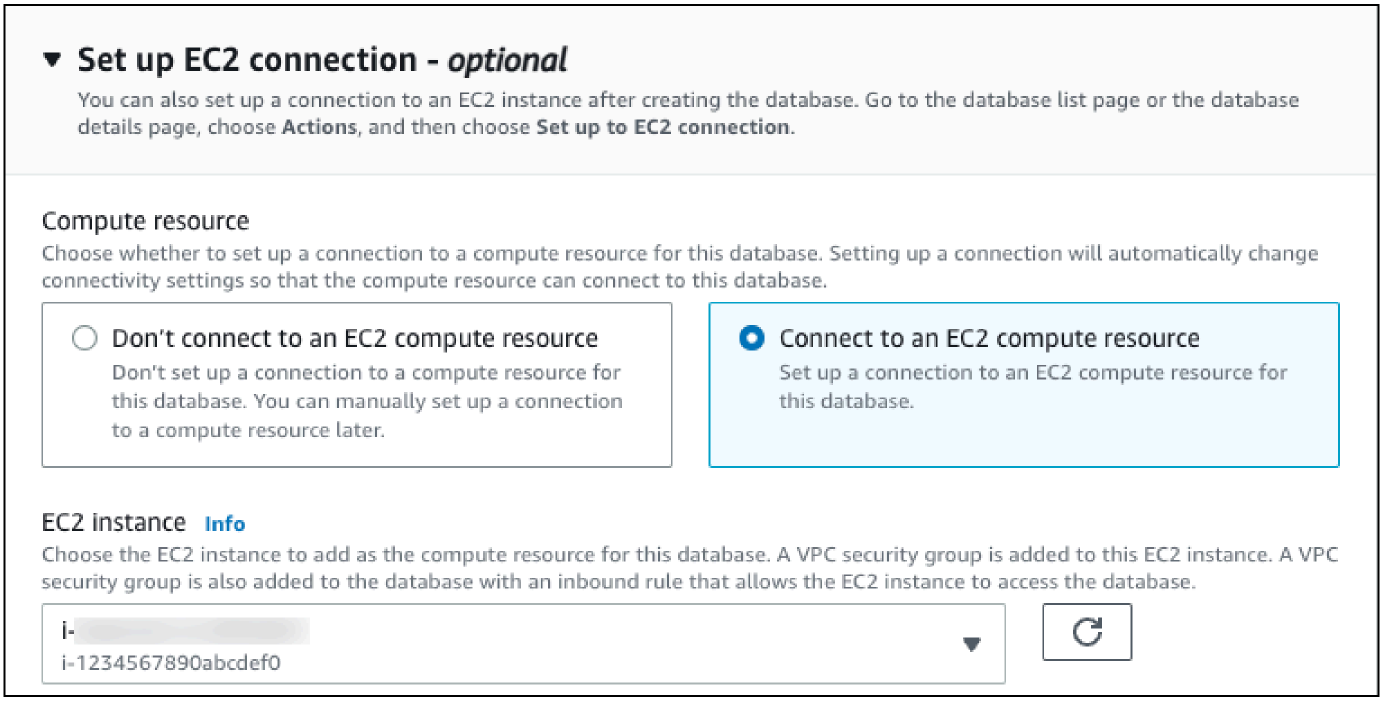


9. 要为数据库实例使用自动生成的主密码，请选择自动生成密码。

要输入主密码，请确保清除自动生成密码，然后在主密码和确认密码中输入相同的密码。

10. 要设置与您之前创建的 EC2 实例的连接，请打开设置 EC2 连接 - 可选。

    选择连接到 EC2 计算资源。选择您之前创建的 EC2 实例。



11. （可选）打开查看轻松创建的默认设置。

View default settings for Easy create

Easy create sets the following configurations to their default values, some of which can be changed later. If you want to change any of these settings now, use **Standard create**.

Configuration	Value	Editable after database is created
Encryption	Enabled	No
VPC	Default VPC (vpc-1a2b3c4d)	No
Option group	default:mysql-8-0	Yes
Subnet group	default	Yes
Automatic backups	Enabled	Yes
VPC security group	sg-0cc53de1b4d1763cf	Yes
Publicly accessible	No	Yes
Database port	3306	Yes
DB instance identifier	database-test1	Yes
DB engine version	8.0.28	Yes
DB parameter group	default.mysql8.0	Yes
Performance insights	Enabled	Yes
Monitoring	Enabled	Yes
Maintenance	Auto minor version upgrade enabled	Yes
Delete protection	Not enabled	Yes

您可以检查在启用 Easy create (轻松创建) 时使用的默认设置。可在创建数据库后编辑列显示在创建数据库后可以更改的选项。

- 如果该列中的某个设置为否，而您想要不同的设置，则可以使用标准创建来创建数据库实例。

- 如果某个设置在该列中为是，您可以使用标准创建来创建数据库实例，也可以在创建后修改数据库实例以更改该设置。

12. 选择创建数据库。

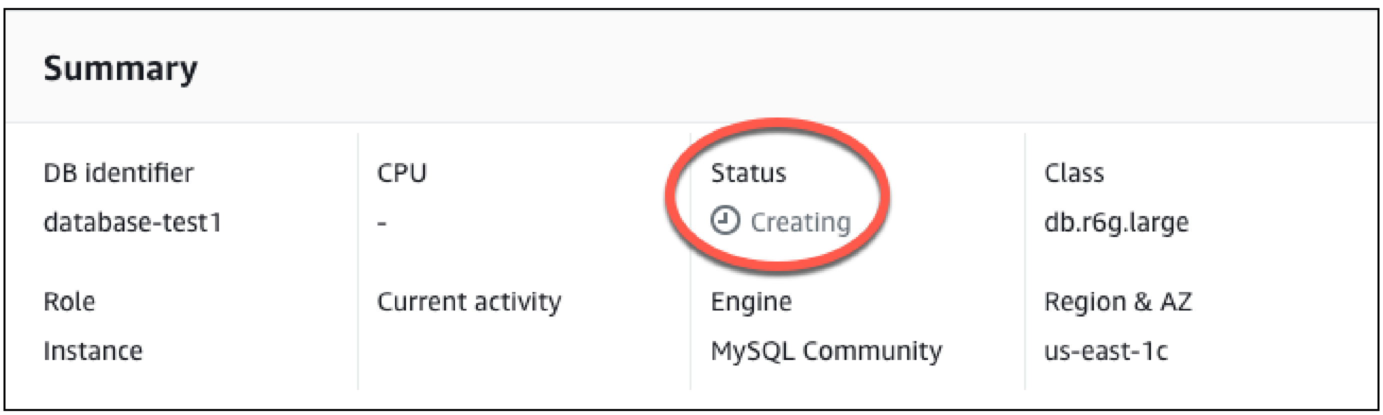
    要查看数据库实例的主用户名和密码，请选择查看凭证详细信息。

    您可以使用显示的用户名和密码，以主用户身份连接到数据库实例。

> **Important**
>
> 您无法再次查看主用户密码。如果您不记录它，您可能需要更改它。
> 如果需要在数据库实例可用后更改主用户密码，则可以修改数据库实例以执行此操作。有关修改 数据库实例的更多信息，请参阅 修改 Amazon RDS 数据库实例。

13. 在数据库列表中，选择新的 MySQL 数据库实例的名称以显示其详细信息。

    数据库实例具有正在创建状态，直到此实例就绪可供使用。



    当状态变为 Available（可用）时，您便可以连接到该数据库实例。根据数据库实例类和存储量，新实例可能需要等待 20 分钟时间才可用。

（可选）使用 Amazon CloudFormation 创建 VPC、EC2 实例和 MySQL 实例

您可以使用 Amazon CloudFormation 通过将基础设施视为代码来配置 Amazon 资源，而无需使用控制台创建 VPC、EC2 实例和 MySQL 实例。为了帮助您将 Amazon 资源组织成更小、更易于管理的单元，您可以使用 Amazon CloudFormation 嵌套堆栈功能。有关更多信息，请参阅在 Amazon CloudFormation 控制台上创建堆栈和使用嵌套堆栈。

> **Important**
>
> Amazon CloudFormation 是免费的，但 CloudFormation 创建的资源是实时的。您需要为这些资源支付标准使用费，直到您终止使用它们为止。总费用将应该最少的。有关如何最大限度降低费用的信息，请转至 Amazon Free Tier。

要使用 Amazon CloudFormation 控制台创建资源，请执行以下步骤：

- 下载 CloudFormation 模板
- 使用 CloudFormation 配置资源

下载 CloudFormation 模板

CloudFormation 模板是一个 JSON 或 YAML 文本文件，其中包含有关您希望在堆栈中创建的资源的配置信息。此模板还为您创建 VPC 和堡垒主机以及 RDS 实例。

要下载模板文件，请打开以下链接 MySQL CloudFormation 模板。

在 Github 页面中，单击 Download raw file 按钮以保存模板 YAML 文件。

使用 CloudFormation 配置资源

> **Note**
>
> 在开始此过程之前，请确保您的 Amazon Web Services 账户中具有 EC2 实例密钥对。有关更多信息，请参阅 Amazon EC2 密钥对和 Linux 实例。

使用 Amazon CloudFormation 模板时，必须选择正确的参数以确保正确创建资源。按以下步骤操作：

1. 登录到 Amazon Web Services Management Console 并打开 Amazon CloudFormation 控制台 https://console.aws.amazon.com/cloudformation。
2. 选择创建堆栈。
3. 在“指定模板”部分，选择从您的计算机上传模板文件，然后选择下一步。
4. 在指定堆栈详细信息页面上，设置以下参数：
   a. 将堆栈名称设置为 MySQLTestStack。

b. 在参数下，通过选择三个可用区来设置可用区。

c. 在 Linux 堡垒主机配置下，在密钥名称中，选择用于登录您的 EC2 实例的密钥对。

d. 在 Linux 堡垒主机配置设置中，将允许的 IP 范围设置为您的 IP 地址。要使用 Secure Shell（SSH）连接到 VPC 中的 EC2 实例，请通过 https://checkip.amazonaws.com 上的服务确定您的公有 IP 地址。IP 地址的示例为 192.0.2.1/32。

> **Warning**
>
> 如果您使用 `0.0.0.0/0` 进行 SSH 访问，则所有 IP 地址可能能够使用 SSH 访问您的公有 EC2 实例。在测试环境下短时间内，此方法尚可接受，但它对于生产环境并不安全。在生产环境中，将仅向特定 IP 地址或地址范围授权使用 SSH 访问您的 EC2 实例。

e. 在数据库常规配置下，将数据库实例类设置为 db.t3.micro。

f. 将数据库名称设置为 **`database-test1`**。

g. 在数据库主用户名中，输入主用户的名称。

h. 在本教程中，将使用 Secrets Manager 管理数据库主用户密码设置为 `false`。

i. 在数据库密码中，设置所选密码。请记住此密码以便在教程中的后续步骤中使用。

j. 在数据库存储配置下，将数据库存储类型设置为 gp2。

k. 在数据库监控配置下，将启用 RDS 性能详情设置为 false。

l. 将所有其他设置保留为默认值。单击下一步继续。

5. 在配置堆栈选项页面中，保留所有默认选项。单击下一步继续。

6. 在查看堆栈页面中，在检查数据库和 Linux 堡垒主机选项后，选择提交。

堆栈创建过程完成后，查看名为 BastionStack 和 RDSNS 的堆栈，记下连接到数据库所需的信息。有关更多信息，请参阅在 Amazon Web Services Management Console 上查看 Amazon CloudFormation 堆栈数据和资源。

连接到 MySQL 数据库实例

您可以使用任何标准 SQL 客户端应用程序连接到数据库实例。在该示例中，您使用 mysql 命令行客户端连接到 MySQL 数据库实例。

连接到 MySQL 数据库实例

1. 找到您的数据库实例的端点（DNS 名称）和端口号。

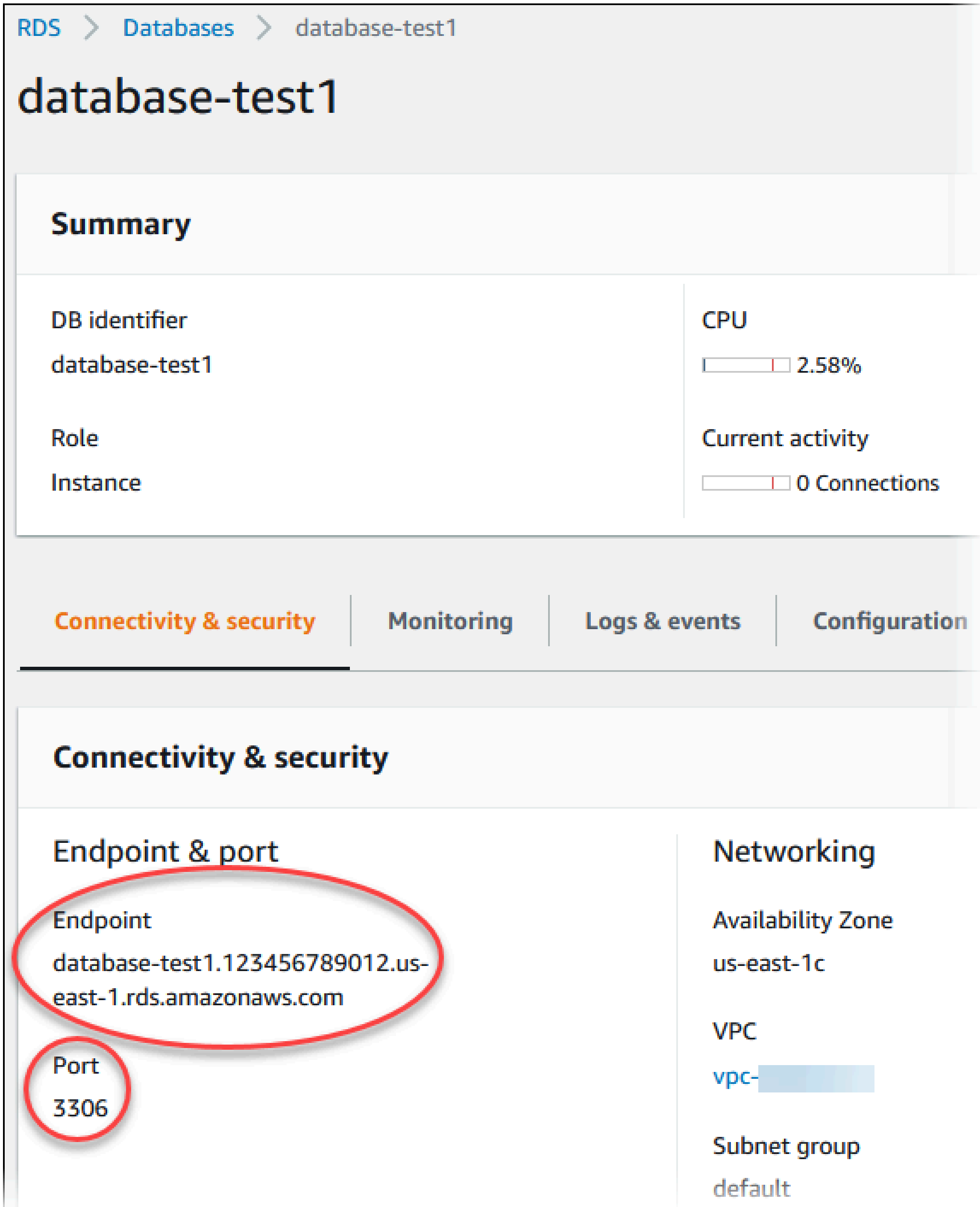
a. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。

b. 在 Amazon RDS 控制台的右上角，选择数据库实例的 Amazon Web Services 区域。

c. 在导航窗格中，选择 Databases（数据库）。

d. 选择 MySQL 数据库实例名称以显示其详细信息。

e. 在连接和安全性选项卡上，复制端点。另请注意端口号。您需要端点和端口号才能连接到数据库实例。



2. 按照《Amazon EC2 用户指南》中的[连接到 Linux 实例]中的步骤操作，连接到您之前创建的 EC2 实例。

   我们建议您使用 SSH 连接到 EC2 实例。如果 SSH 客户端实用程序安装在 Windows、Linux 或 Mac 上，则可以使用以下命令格式连接到该实例：

```
ssh -i location_of_pem_file ec2-user@ec2-instance-public-dns-name
```

例如，假设在 Linux 上 `ec2-database-connect-key-pair.pem` 存储在 `/dir1` 中，而 EC2 实例的公有 IPv4 DNS 为 `ec2-12-345-678-90.compute-1.amazonaws.com`。SSH 命令将如下所示：

```
ssh -i /dir1/ec2-database-connect-key-pair.pem ec2-user@ec2-12-345-678-90.compute-1.amazonaws.com
```

3. 通过更新 EC2 实例上的软件，获取最新的错误修复和安全更新。要执行此操作，请使用以下命令。

> **Note**
>
> `-y` 选项安装更新时不提示确认。要在安装前检查更新，请忽略该选项。

```
sudo dnf update -y
```

4. 要在 Amazon Linux 2023 上安装 MariaDB 中的 mysql 命令行客户端，请运行以下命令：

```
sudo dnf install mariadb105
```

5. 连接到 MySQL 数据库实例。例如，输入以下命令。此操作可让您使用 MySQL 客户端连接到 MySQL 数据库实例。

将 *`endpoint`* 替换为数据库实例端点（DNS 名称），并替换用于 *`admin`* 的主用户名。提示输入密码时，提供所使用的主密码。

```
mysql -h endpoint -P 3306 -u admin -p
```

在输入用户的密码后，您应该会看到类似于以下内容的输出。

```
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 3082
Server version: 8.0.28 Source distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.
```

```
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MySQL [(none)]>
```

有关连接到数据库实例的更多信息，请参阅连接到 MySQL 数据库实例。如果您无法连接到数据库实例，请参阅 无法连接到 Amazon RDS 数据库实例。

出于安全考虑，最佳做法是使用加密连接。仅当客户端和服务器位于同一 VPC 中，并且网络受信任时，才会使用未加密的 MySQL 连接。有关使用加密连接的信息，请参阅 从 MySQL 命令行客户端使用 SSL/TLS 连接到 Amazon RDS 上的 MySQL 数据库实例（已加密）。

6. 运行 SQL 命令。

   例如，以下 SQL 命令显示了当前日期和时间：

```
SELECT CURRENT_TIMESTAMP;
```

删除 EC2 实例和数据库实例

在连接到并浏览您创建的示例 EC2 实例和数据库实例之后，删除它们，以便不再为其付费。

如果您使用 Amazon CloudFormation 创建了资源，请跳过此步骤，转至下一步。

删除 EC2 实例

1. 登录到 Amazon Web Services Management Console 并打开 Amazon EC2 控制台（https://console.aws.amazon.com/ec2/）。
2. 在导航窗格中，选择实例。
3. 选择 EC2 实例，然后依次选择实例状态、终止实例。
4. 当系统提示您确认时，选择终止。

有关删除 EC2 实例的更多信息，请参阅《Amazon EC2 用户指南》中的终止实例。

删除无最终数据库快照的数据库实例

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。

3. 选择要删除的数据库实例。
4. 对于 Actions，选择 Delete。
5. 清除是否创建最终快照？和保留自动备份。
6. 完成确认并选择删除。

（可选）删除使用 CloudFormation 创建的 EC2 实例和数据库实例

如果您使用 Amazon CloudFormation 创建了资源，请在连接并浏览 EC2 实例和数据库实例示例之后，删除 CloudFormation 堆栈，以便不再为其付费。

删除 CloudFormation 资源

1. 打开 Amazon CloudFormation 控制台。
2. 在 CloudFormationconsole 的堆栈页面上，选择根堆栈（名称不是 VPCStack、BastionStack 或 RDSNS 的堆栈）。
3. 选择删除。
4. 提示进行确认时，选择删除堆栈。

有关如何删除 CloudFormation 中堆栈的信息，请参阅《Amazon CloudFormation 用户指南》中的在 Amazon CloudFormation 控制台上删除堆栈。

（可选）将您的数据库实例连接到 Lambda 函数

您也可以将您的 RDS for MySQL 数据库实例连接到 Lambda 无服务器计算资源。Lambda 函数支持在不预调配或管理基础设施的情况下运行代码。Lambda 函数还支持自动响应任何规模的代码执行请求，从每天十几个事件到每秒数百个事件。有关更多信息，请参阅 自动连接 Lambda 函数和数据库实例。

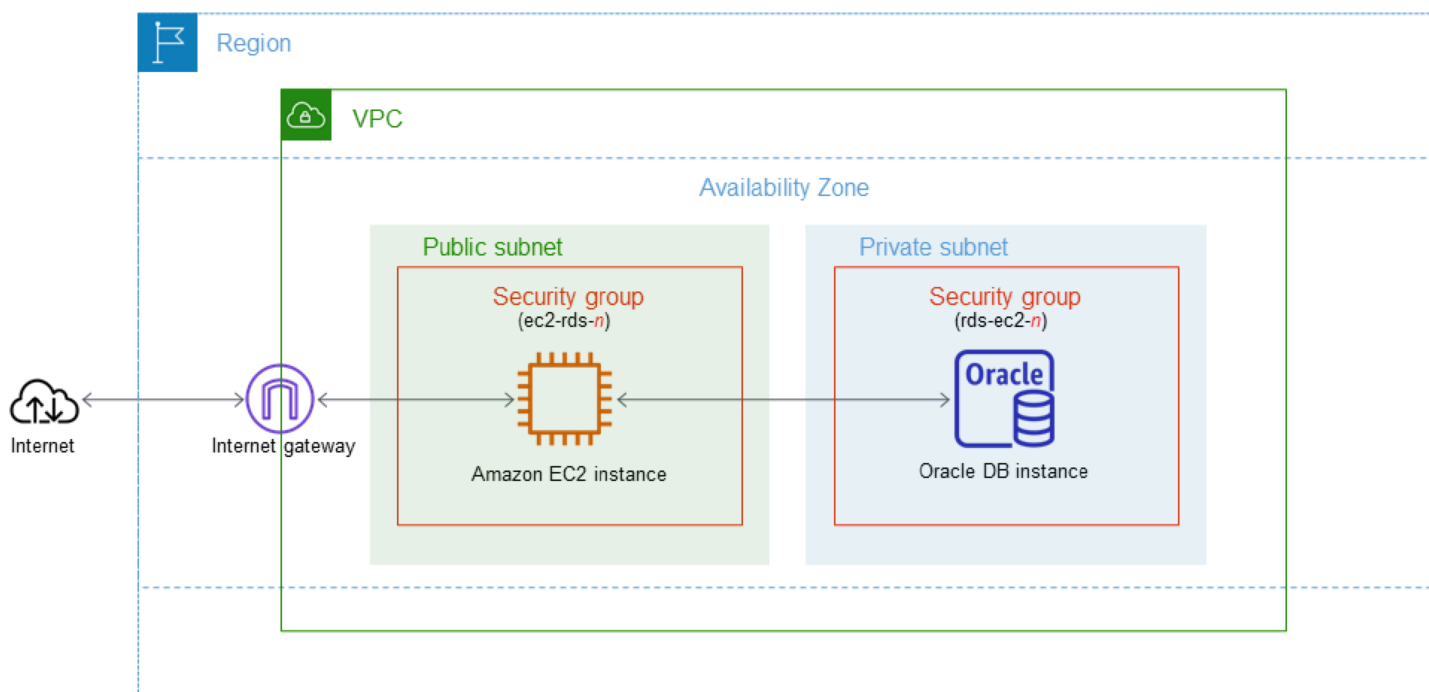
创建并连接到 Oracle 数据库实例

本教程创建一个 EC2 实例和一个 RDS for Oracle 数据库实例。本教程向您展示如何使用标准 Oracle 客户端从 EC2 实例访问数据库实例。作为最佳实践，本教程在虚拟私有云（VPC）中创建了私有数据库实例。在大多数情况下，同一 VPC 中的其他资源（例如 EC2 实例）可以访问数据库实例，但 VPC 之外的资源无法访问该实例。

完成本教程后，VPC 的每个可用区中都有一个公有和私有子网。在一个可用区中，EC2 实例在公有子网中，数据库实例在私有子网中。

> **Important**
>
> 创建 Amazon 账户并不会收费；但是，在完成本教程过程中使用 Amazon 资源可能会产生费用。完成本教程后，如果不再需要这些资源，可以将其删除。

下图显示了教程完成时的配置。



本教程可帮助您通过以下方法之一创建资源：

1. 使用 Amazon Web Services Management Console - 步骤 2：创建 Oracle 数据库实例 和 步骤 1：创建 EC2 实例

2. 使用 Amazon CloudFormation 创建数据库实例和 EC2 实例 - （可选）使用 Amazon CloudFormation 创建 VPC、EC2 实例和 Oracle 数据库实例

第一种方法使用轻松创建，通过 Amazon Web Services Management Console 创建私有 Oracle 数据库实例。您可以仅指定数据库引擎类型、数据库实例大小和数据库实例标识符。轻松创建为其他配置选项使用默认设置。

如果改用标准创建，则在创建数据库实例时可以指定更多配置选项。这些选项包括可用性、安全性、备份和维护设置。要创建公有数据库实例，必须使用标准创建。有关信息，请参阅创建 Amazon RDS 数据库实例。

主题

先决条件

在开始之前，请完成以下各节中的步骤：

- 注册 Amazon Web Services 账户
- 保护 IAM 用户

步骤 1：创建 EC2 实例

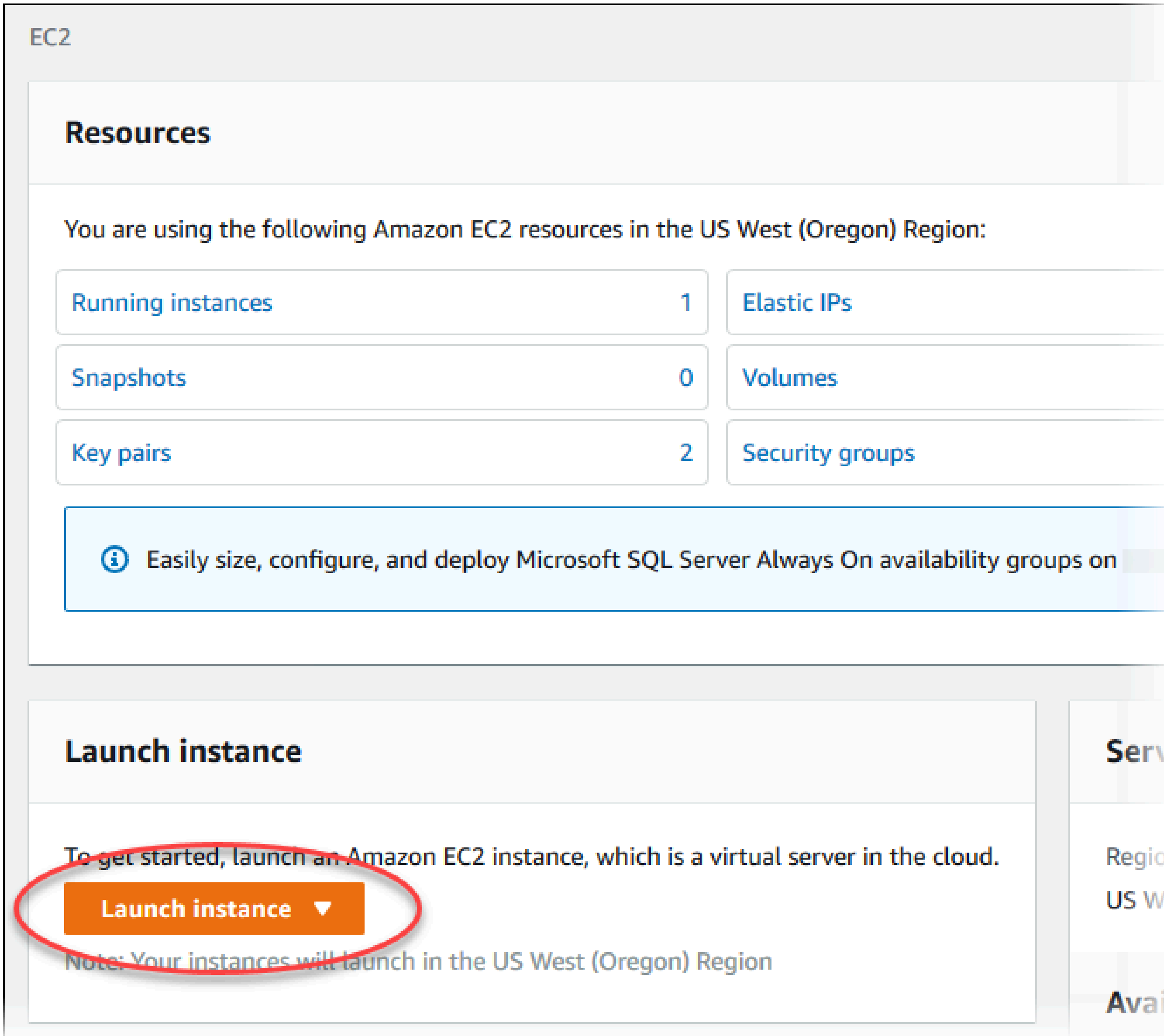
创建将用于连接到数据库的 Amazon EC2 实例。

创建 EC2 实例

1. 登录到 Amazon Web Services Management Console 并打开 Amazon EC2 控制台（https://console.aws.amazon.com/ec2/）。

2. 在 Amazon Web Services Management Console的右上角，选择要在其中创建 EC2 实例的 Amazon Web Services 区域。

3. 选择 EC2 控制面板，然后选择启动实例，如下图所示。

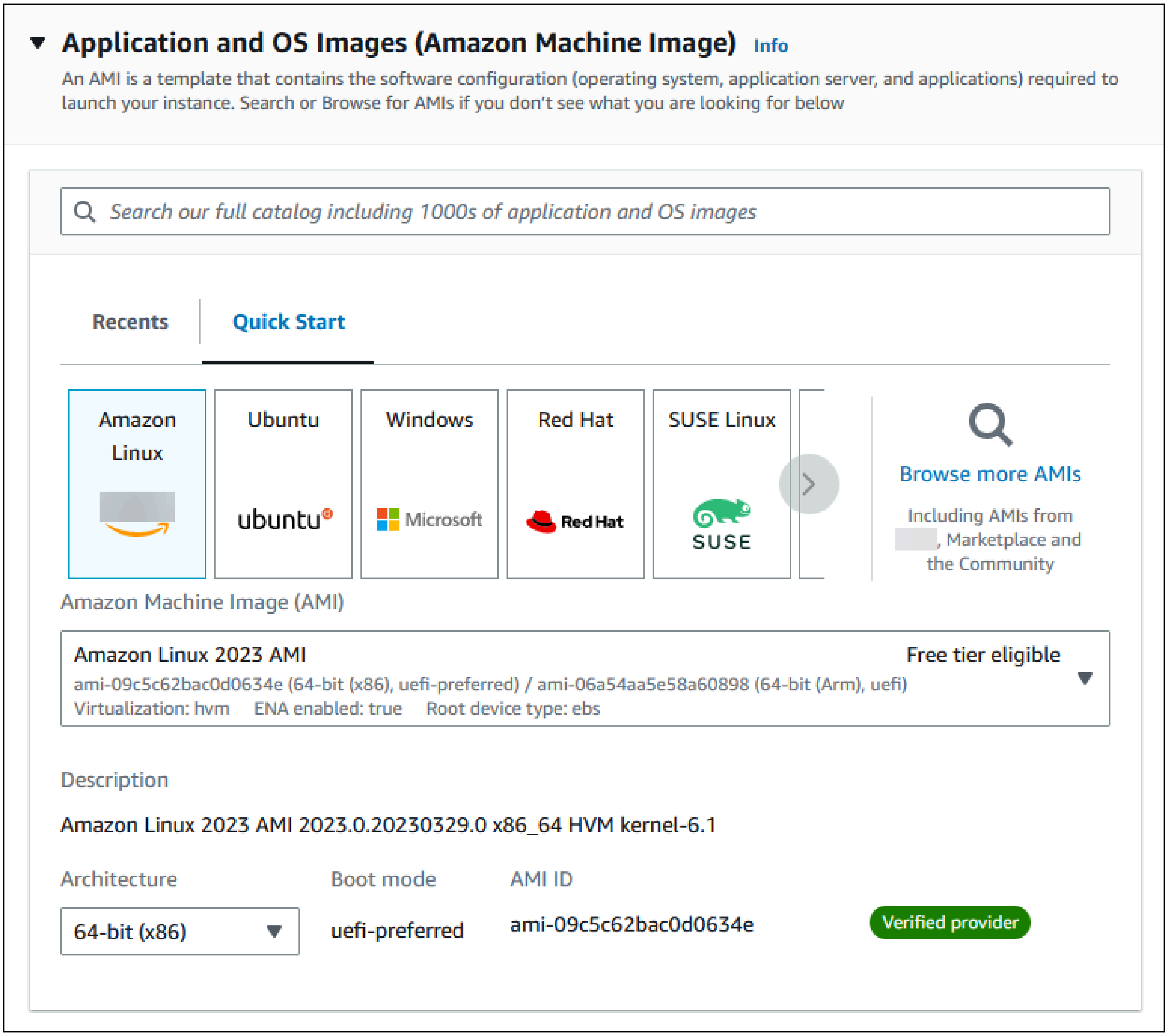


启动实例页面打开。

4. 在启动实例页面上选择以下设置。

   a. 在 Name and tags（名称和标签）下，对于 Name（名称），输入 **ec2-database-connect**。

   b. 在应用程序和操作系统映像（Amazon 机器映像）下，选择 Amazon Linux，然后选择 Amazon Linux 2023 AMI。对于其他选项，保留默认选择。



c. 在 Instance type（实例类型）下，选择 t2.micro。

d. 在 Key pair (login) [密钥对（登录）] 下，选择 Key pair name（密钥对名称）以使用现有密钥对。要为 Amazon EC2 实例创建新的密钥对，请选择 Create new key pair（创建新的密钥对），然后使用 Create key pair（创建密钥对）窗口来创建它。

   有关创建新的密钥对的更多信息，请参阅《Amazon EC2 用户指南》中的[创建密钥对](#)。

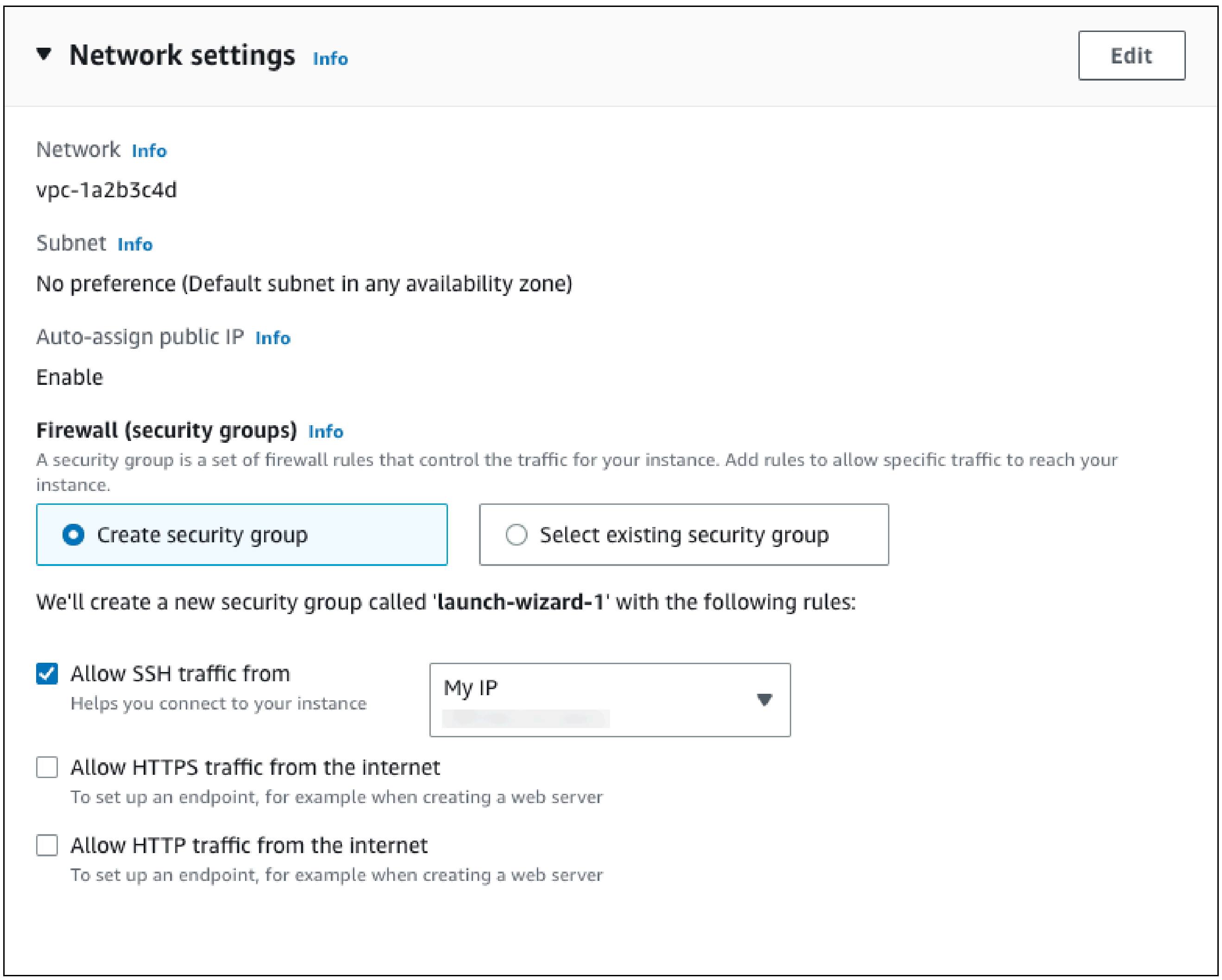
e. 对于网络设置中的允许 SSH 流量，选择 EC2 实例的 SSH 连接来源。

   如果显示的 IP 地址对于 SSH 连接是正确的，您可以选择 My IP（我的 IP）。否则，您可以确定要用来通过 Secure Shell（SSH）连接到 VPC 中的 EC2 实例的 IP 地址。要确定您的公有 IP 地址，请在新的浏览器窗口或标签页中，使用 https://checkip.amazonaws.com 上的服务。IP 地址的示例为 192.0.2.1/32。

在许多情况下，您可能通过互联网服务提供商（ISP）进行连接，或者在不使用静态 IP 地址的情况下从防火墙之后进行连接。如果是这样，请确保确定客户端计算机使用的 IP 地址范围。

> **Warning**
>
> 如果您使用 `0.0.0.0/0` 进行 SSH 访问，则所有 IP 地址可能能够使用 SSH 访问您的公有 EC2 实例。在测试环境下短时间内，此方法尚可接受，但它对于生产环境并不安全。在生产环境中，将仅向特定 IP 地址或地址范围授权使用 SSH 访问您的 EC2 实例。

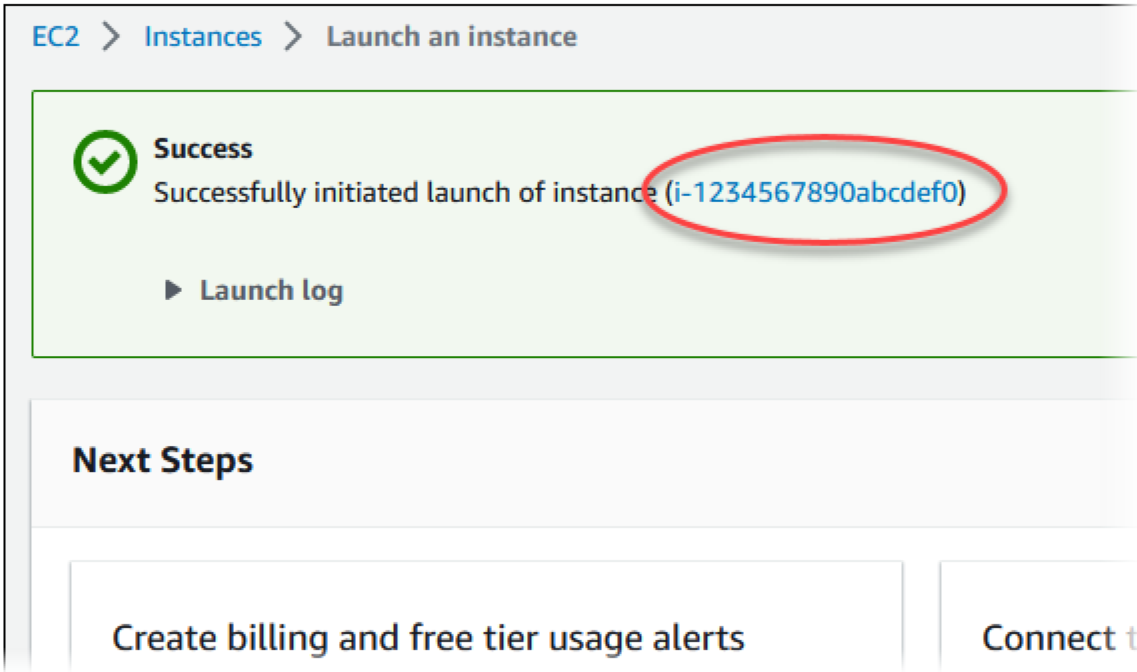
下图显示了网络设置部分的示例。



f. 对于其余部分保留默认值。

g. 查看摘要面板中您的 EC2 实例配置的摘要，当您准备好后，选择启动实例。

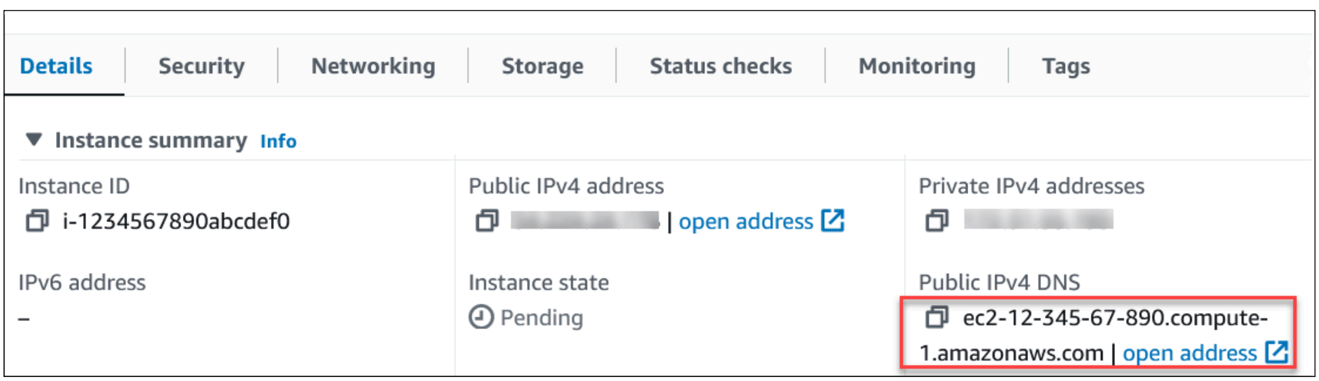
5. 在启动状态页面上，记下新 EC2 实例的标识符，例如：`i-1234567890abcdef0`。



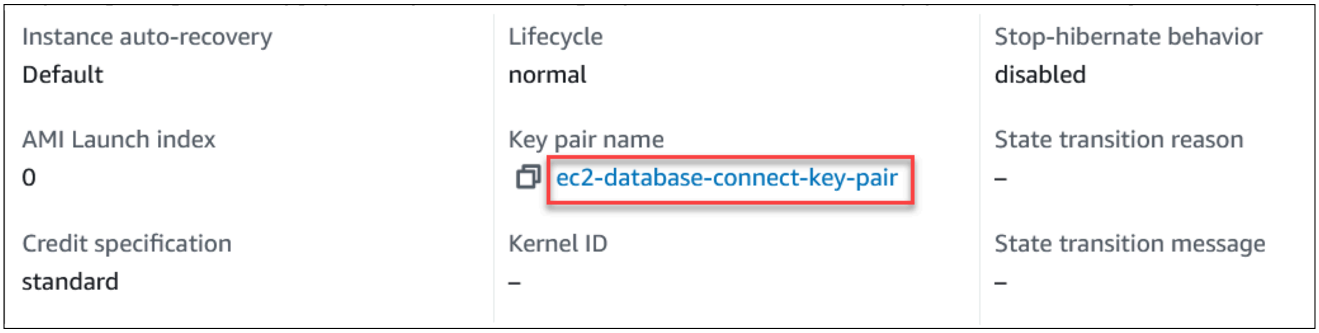
6. 选择 EC2 实例标识符以打开 EC2 实例列表，然后选择您的 EC2 实例。

7. 在详细信息选项卡中，记下使用 SSH 进行连接时所需的以下值：

a. 在实例摘要中，记下公有 IPv4 DNS 的值。



b. 在实例详细信息中，记下密钥对名称的值。



8. 等待 EC2 实例的实例状态变为正在运行，然后继续。

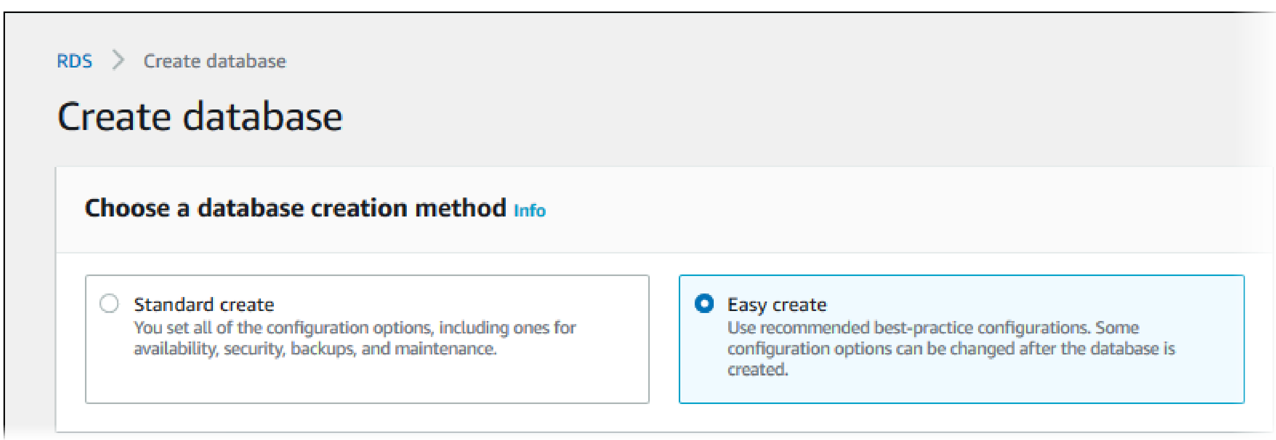
步骤 2：创建 Oracle 数据库实例

Amazon RDS 的基本构建基块是数据库实例。这是您将要在其中运行 Oracle 数据库的环境。

在该示例中，您使用轻松创建来创建一个数据库实例，它运行带有 db.m5.large 数据库实例类的 Oracle 数据库引擎。

使用“轻松创建”创建 Oracle 数据库实例

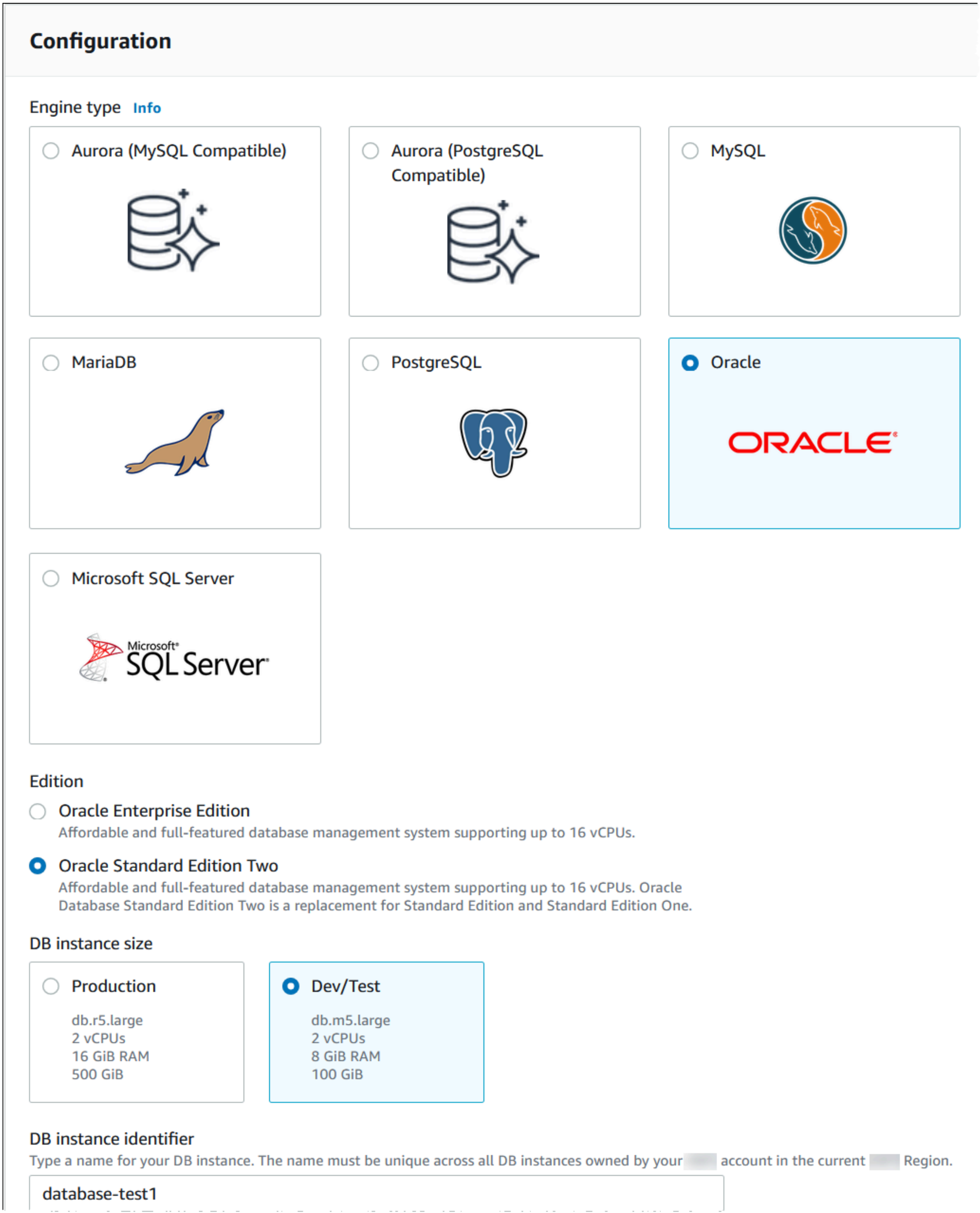
1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在 Amazon RDS 控制台的右上角，选择要在其中创建数据库实例的 Amazon Web Services 区域。
3. 在导航窗格中，选择 Databases (数据库)。
4. 选择 Create database (创建数据库)，并确保已选择 Easy create (轻松创建)。



5. 在配置中，选择 Oracle。
6. 对于数据库实例大小，选择设备/测试。

7. 对于 DB instance identifier（数据库实例标识符），输入 **`database-test1`**。
8. 对于主用户名，输入主用户的名称，或者保留原定设置名称。

   创建数据库页面应类似于以下图像。

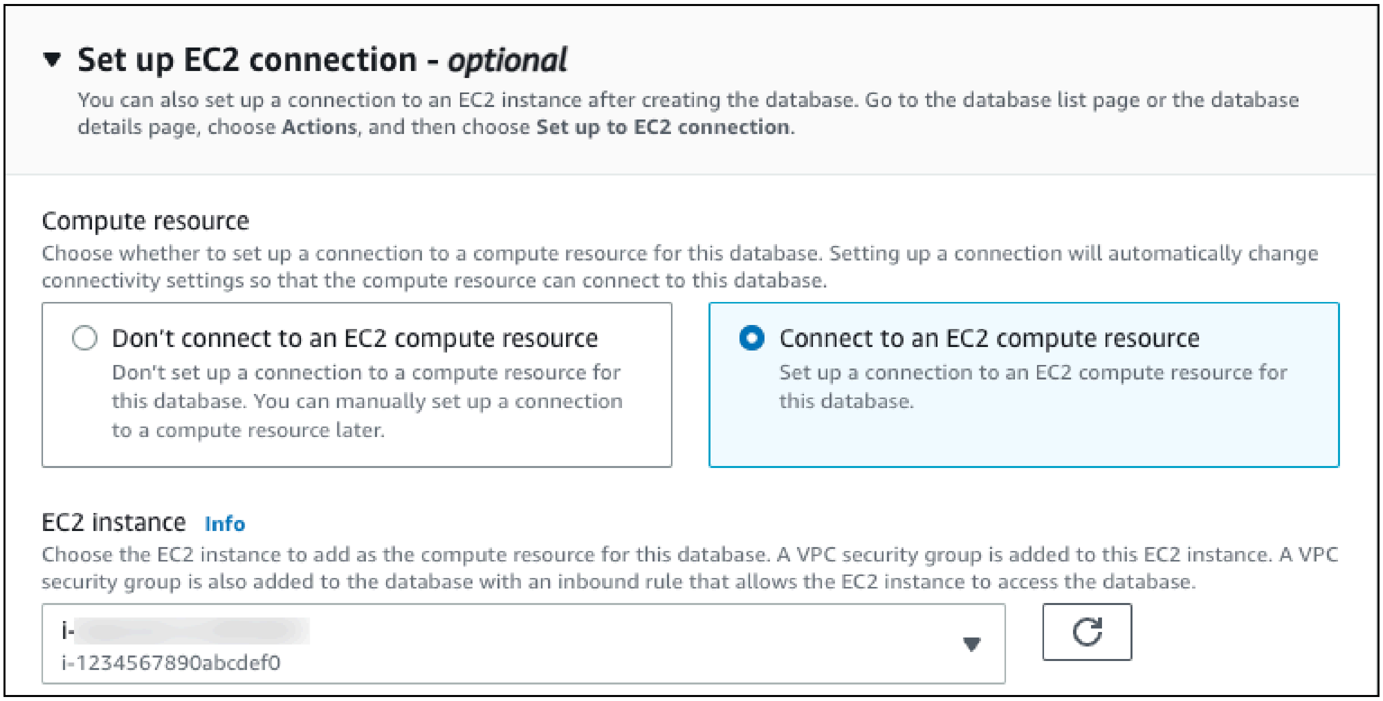

Configuration
Engine type Info
Aurora (MySQL Compatible)
Aurora (PostgreSQL Compatible)
MySQL
MariaDB
PostgreSQL
Oracle
ORACLE
Microsoft SQL Server
Microsoft SQL Server
Edition
Oracle Enterprise Edition
Affordable and full-featured database management system supporting up to 16 vCPUs.
Oracle Standard Edition Two
Affordable and full-featured database management system supporting up to 16 vCPUs. Oracle Database Standard Edition Two is a replacement for Standard Edition and Standard Edition One.
DB instance size
Production
db.r5.large
2 vCPUs
16 GiB RAM
500 GiB
Dev/Test
db.m5.large
2 vCPUs
8 GiB RAM
100 GiB
DB instance identifier
Type a name for your DB instance. The name must be unique across all DB instances owned by your account in the current Region.
database-test1

9. 要为数据库实例使用自动生成的主密码，请选择自动生成密码。

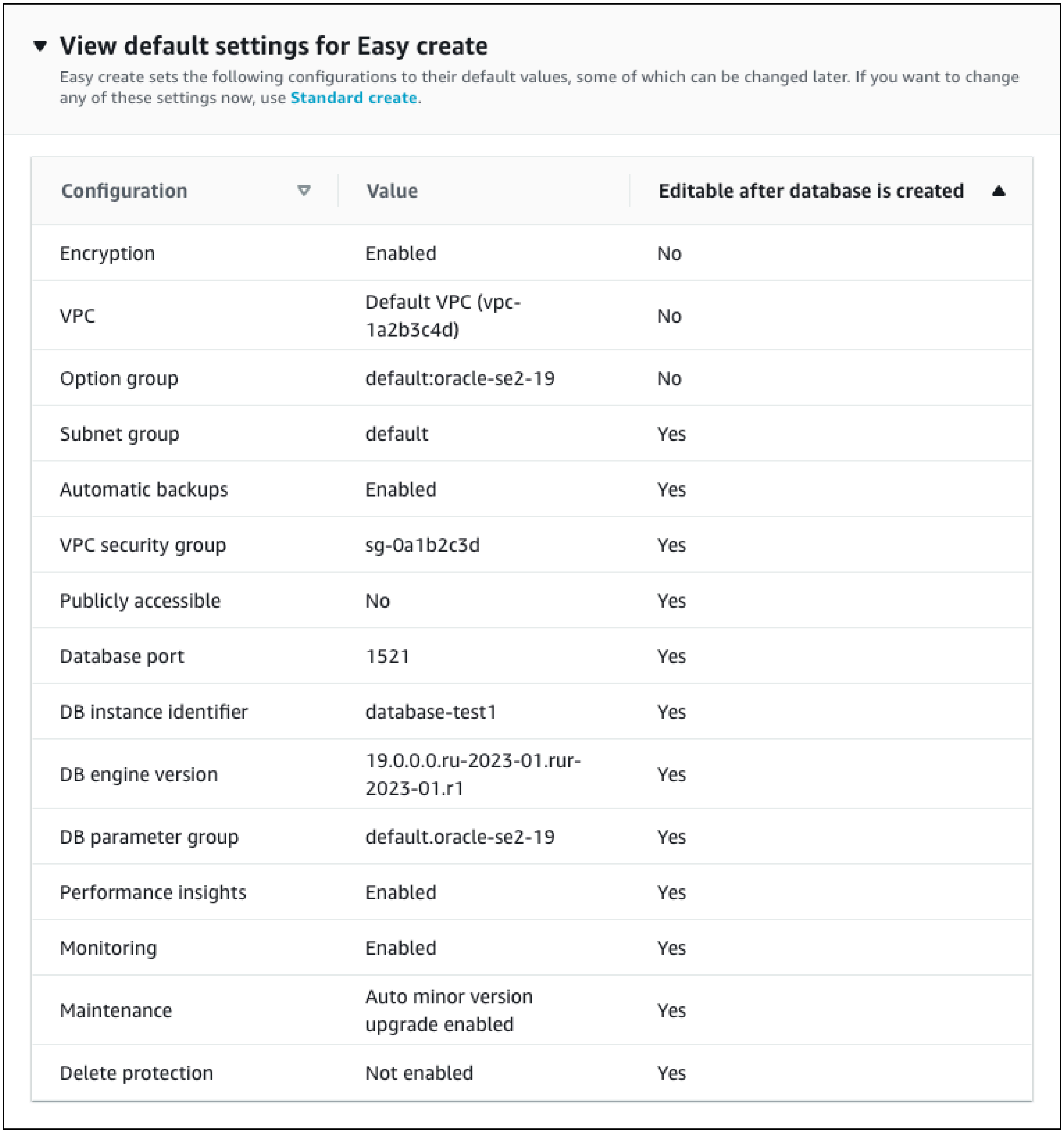
   要输入主密码，请确保清除自动生成密码，然后在主密码和确认密码中输入相同的密码。

10. 要设置与您之前创建的 EC2 实例的连接，请打开设置 EC2 连接 - 可选。

    选择连接到 EC2 计算资源。选择您之前创建的 EC2 实例。



11. 打开查看轻松创建的默认设置。



View default settings for Easy create

Easy create sets the following configurations to their default values, some of which can be changed later. If you want to change any of these settings now, use **Standard create**.

Configuration	Value	Editable after database is created
Encryption	Enabled	No
VPC	Default VPC (vpc-1a2b3c4d)	No
Option group	default:oracle-se2-19	No
Subnet group	default	Yes
Automatic backups	Enabled	Yes
VPC security group	sg-0a1b2c3d	Yes
Publicly accessible	No	Yes
Database port	1521	Yes
DB instance identifier	database-test1	Yes
DB engine version	19.0.0.0.ru-2023-01.rur-2023-01.r1	Yes
DB parameter group	default.oracle-se2-19	Yes
Performance insights	Enabled	Yes
Monitoring	Enabled	Yes
Maintenance	Auto minor version upgrade enabled	Yes
Delete protection	Not enabled	Yes

您可以检查在启用 Easy create (轻松创建) 时使用的默认设置。可在创建数据库后编辑列显示在创建数据库后可以更改的选项。

- 如果该列中的某个设置为否，而您想要不同的设置，则可以使用标准创建来创建数据库实例。

- 如果某个设置在该列中为是，您可以使用标准创建来创建数据库实例，也可以在创建后修改数据库实例以更改该设置。

12. 选择创建数据库。

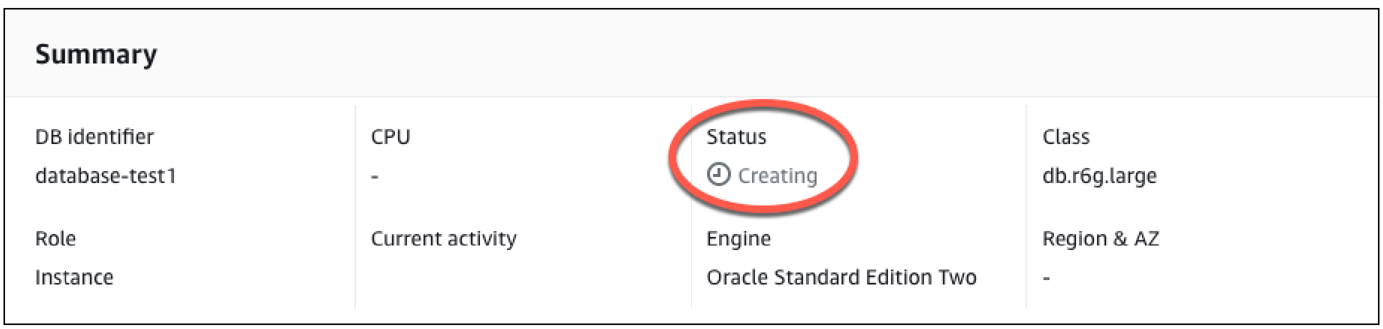
    要查看数据库实例的主用户名和密码，请选择查看凭证详细信息。

    您可以使用显示的用户名和密码，以主用户身份连接到数据库实例。

    > **Important**
    >
    > 您无法再次查看主用户密码。如果您不记录它，您可能需要更改它。
    > 如果需要在数据库实例可用后更改主用户密码，则可以修改数据库实例以执行此操作。有关修改 数据库实例的更多信息，请参阅 修改 Amazon RDS 数据库实例。

13. 在数据库列表中，选择新的 Oracle 数据库实例的名称以显示其详细信息。

    数据库实例具有正在创建状态，直到此实例就绪可供使用。



    当状态变为 Available（可用）时，您便可以连接到该数据库实例。根据数据库实例类和存储量，新实例可能需要等待 20 分钟时间才可用。在创建数据库实例时，您可以继续执行下一步并创建 EC2 实例。

（可选）使用 Amazon CloudFormation 创建 VPC、EC2 实例和 Oracle 数据库实例

您可以使用 Amazon CloudFormation 通过将基础设施视为代码来配置 Amazon 资源，而无需使用控制台创建 VPC、EC2 实例和 Oracle 数据库实例。为了帮助您将 Amazon 资源组织成更小、更易于管理的单元，您可以使用 Amazon CloudFormation 嵌套堆栈功能。有关更多信息，请参阅在 Amazon CloudFormation 控制台上创建堆栈和使用嵌套堆栈。

> **Important**
>
> Amazon CloudFormation 是免费的，但 CloudFormation 创建的资源是实时的。您需要为这些资源支付标准使用费，直到您终止使用它们为止。总费用将应该最少的。有关如何最大限度降低费用的信息，请转至 Amazon Free Tier。

要使用 Amazon CloudFormation 控制台创建资源，请执行以下步骤：

- 步骤 1：下载 CloudFormation 模板
- 步骤 2：使用 CloudFormation 配置资源

下载 CloudFormation 模板

CloudFormation 模板是一个 JSON 或 YAML 文本文件，其中包含有关您希望在堆栈中创建的资源的配置信息。此模板还为您创建 VPC 和堡垒主机以及 RDS 实例。

要下载模板文件，请打开以下链接 Oracle CloudFormation 模板。

在 Github 页面中，单击 Download raw file 按钮以保存模板 YAML 文件。

使用 CloudFormation 配置资源

> **Note**
>
> 在开始此过程之前，请确保您的 Amazon Web Services 账户中具有 EC2 实例密钥对。有关更多信息，请参阅 Amazon EC2 密钥对和 Linux 实例。

使用 Amazon CloudFormation 模板时，必须选择正确的参数以确保正确创建资源。按以下步骤操作：

1. 登录到 Amazon Web Services Management Console 并打开 Amazon CloudFormation 控制台 https://console.aws.amazon.com/cloudformation。
2. 选择创建堆栈。
3. 在“指定模板”部分，选择从您的计算机上传模板文件，然后选择下一步。
4. 在指定堆栈详细信息页面上，设置以下参数：

   a. 将堆栈名称设置为 OracleTestStack。

b. 在参数下，通过选择三个可用区来设置可用区。

c. 在 Linux 堡垒主机配置下，在密钥名称中，选择用于登录您的 EC2 实例的密钥对。

d. 在 Linux 堡垒主机配置设置中，将允许的 IP 范围设置为您的 IP 地址。要使用 Secure Shell（SSH）连接到 VPC 中的 EC2 实例，请通过 https://checkip.amazonaws.com 上的服务确定您的公有 IP 地址。IP 地址的示例为 192.0.2.1/32。

> **Warning**
>
> 如果您使用 `0.0.0.0/0` 进行 SSH 访问，则所有 IP 地址可能能够使用 SSH 访问您的公有 EC2 实例。在测试环境下短时间内，此方法尚可接受，但它对于生产环境并不安全。在生产环境中，将仅向特定 IP 地址或地址范围授权使用 SSH 访问您的 EC2 实例。

e. 在数据库常规配置下，将数据库实例类设置为 db.t3.micro。

f. 将数据库名称设置为 **`database-test1`**。

g. 在数据库主用户名中，输入主用户的名称。

h. 在本教程中，将使用 Secrets Manager 管理数据库主用户密码设置为 `false`。

i. 在数据库密码中，设置所选密码。请记住此密码以便在教程中的后续步骤中使用。

j. 在数据库存储配置下，将数据库存储类型设置为 gp2。

k. 在数据库监控配置下，将启用 RDS 性能详情设置为 false。

l. 将所有其他设置保留为默认值。单击下一步继续。

5. 在配置堆栈选项页面中，保留所有默认选项。单击下一步继续。

6. 在查看堆栈页面中，在检查数据库和 Linux 堡垒主机选项后，选择提交。

堆栈创建过程完成后，查看名为 BastionStack 和 RDSNS 的堆栈，记下连接到数据库所需的信息。有关更多信息，请参阅在 Amazon Web Services Management Console 上查看 Amazon CloudFormation 堆栈数据和资源。

步骤 3：将 SQL 客户端连接到 Oracle 数据库实例

您可以使用任何标准 SQL 客户端应用程序连接到数据库实例。在该示例中，您使用 Oracle 命令行客户端连接到 Oracle 数据库实例。

连接到 Oracle 数据库实例

1. 找到您的数据库实例的端点（DNS 名称）和端口号。

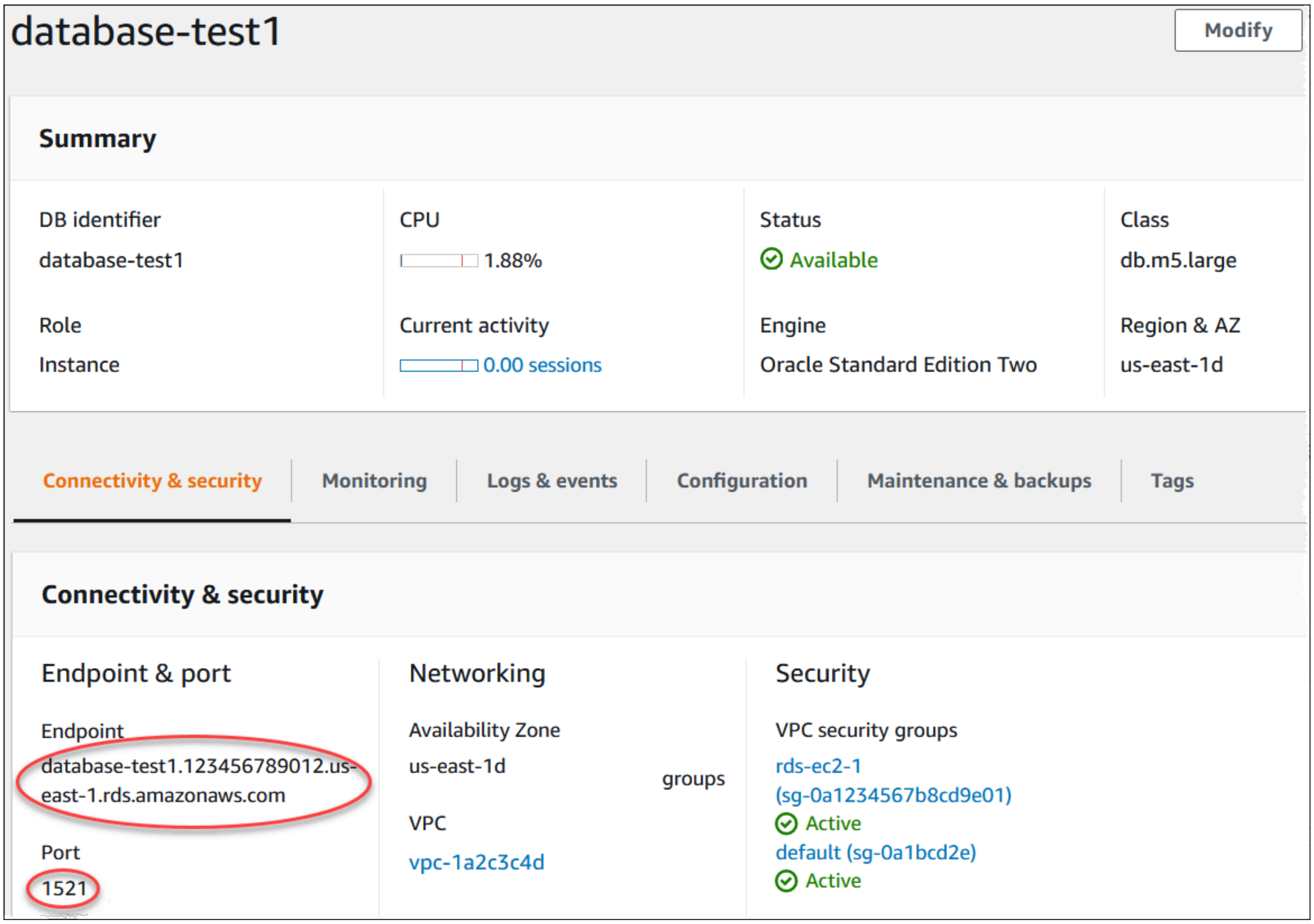
a. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。

b. 在 Amazon RDS 控制台的右上角，选择数据库实例的 Amazon Web Services 区域。

c. 在导航窗格中，选择 Databases（数据库）。

d. 选择 Oracle 数据库实例名称以显示其详细信息。

e. 在连接和安全性选项卡上，复制端点。另请注意端口号。您需要端点和端口号才能连接到数据库实例。



2. 按照《Amazon EC2 用户指南》中的连接到 Linux 实例中的步骤操作，连接到您之前创建的 EC2 实例。

我们建议您使用 SSH 连接到 EC2 实例。如果 SSH 客户端实用程序安装在 Windows、Linux 或 Mac 上，则可以使用以下命令格式连接到该实例：

```
ssh -i location_of_pem_file ec2-user@ec2-instance-public-dns-name
```

例如，假设在 Linux 上 ec2-database-connect-key-pair.pem 存储在 /dir1 中，而 EC2 实例的公有 IPv4 DNS 为 ec2-12-345-678-90.compute-1.amazonaws.com。SSH 命令将如下所示：

```
ssh -i /dir1/ec2-database-connect-key-pair.pem ec2-
user@ec2-12-345-678-90.compute-1.amazonaws.com
```

3. 通过更新 EC2 实例上的软件，获取最新的错误修复和安全更新。为此，请使用以下命令。

   > **Note**
   >
   > -y 选项安装更新时不提示确认。要在安装前检查更新，请忽略该选项。

   ```
   sudo dnf update -y
   ```

4. 在 Web 浏览器中，转到 https://www.oracle.com/database/technologies/instant-client/linux-x86-64-downloads.html。
5. 对于网页上显示的最新数据库版本，请复制 Instant Client 基本软件包和 SQL*Plus 软件包的 .rpm 链接（不是 .zip 链接）。例如，以下链接适用于 Oracle Database 21.9 版本：
   - https://download.oracle.com/otn_software/linux/instantclient/219000/oracle-instantclient-basic-21.9.0.0.0-1.el8.x86_64.rpm
   - https://download.oracle.com/otn_software/linux/instantclient/219000/oracle-instantclient-sqlplus-21.9.0.0.0-1.el8.x86_64.rpm
6. 在 SSH 会话中，运行 wget 命令以从您在上一步中获得的链接下载 .rpm 文件。以下示例下载 Oracle Database 21.9 版本的 .rpm 文件：

   ```
   wget https://download.oracle.com/otn_software/linux/instantclient/219000/oracle-
   instantclient-basic-21.9.0.0.0-1.el8.x86_64.rpm
   wget https://download.oracle.com/otn_software/linux/instantclient/219000/oracle-
   instantclient-sqlplus-21.9.0.0.0-1.el8.x86_64.rpm
   ```

7. 通过运行 dnf 命令安装软件包，如下所示：

   ```
   sudo dnf install oracle-instantclient-*.rpm
   ```

8. 启动 SQL**Pus，然后连接到 Oracle 数据库实例。例如，输入以下命令。

将 *oracle-db-instance-endpoint* 替换为数据库实例端点（DNS 名称），并将 *admin* 替换为您使用的主用户名。对于 Oracle 使用轻松创建时，数据库名称为 DATABASE。提示输入密码时，提供所使用的主密码。

```
sqlplus admin@oracle-db-instance-endpoint:1521/DATABASE
```

在输入用户的密码后，您应该会看到类似于以下内容的输出。

```
SQL*Plus: Release 21.0.0.0.0 - Production on Wed Mar 1 16:41:28 2023
Version 21.9.0.0.0

Copyright (c) 1982, 2022, Oracle.  All rights reserved.

Enter password:
Last Successful login time: Wed Mar 01 2023 16:30:52 +00:00

Connected to:
Oracle Database 19c Standard Edition 2 Release 19.0.0.0.0 - Production
Version 19.18.0.0.0

SQL>
```

有关连接到 RDS for Oracle 数据库实例的更多信息，请参阅连接到 Oracle 数据库实例。如果您无法连接到数据库实例，请参阅 无法连接到 Amazon RDS 数据库实例。

出于安全考虑，最佳做法是使用加密连接。仅当客户端和服务器位于同一 VPC 中，并且网络受信任时，才会使用未加密的 Oracle 连接。有关使用加密连接的信息，请参阅 保护 Oracle 数据库实例连接安全。

9. 运行 SQL 命令。

   例如，以下 SQL 命令显示了当前日期：

```
SELECT SYSDATE FROM DUAL;
```

步骤 4：删除 EC2 实例和数据库实例

在连接到并浏览您创建的示例 EC2 实例和数据库实例之后，删除它们，以便不再为其付费。

如果您使用 Amazon CloudFormation 创建了资源，请跳过此步骤，转至下一步。

删除 EC2 实例

1. 登录到 Amazon Web Services Management Console 并打开 Amazon EC2 控制台（https://console.aws.amazon.com/ec2/）。
2. 在导航窗格中，选择实例。
3. 选择 EC2 实例，然后依次选择实例状态、终止实例。
4. 当系统提示您确认时，选择终止。

有关删除 EC2 实例的更多信息，请参阅《Amazon EC2 用户指南》中的终止实例。

删除无最终数据库快照的数据库实例

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。
3. 选择要删除的数据库实例。
4. 对于 Actions，选择 Delete。
5. 清除是否创建最终快照？和保留自动备份。
6. 完成确认并选择删除。

（可选）删除使用 CloudFormation 创建的 EC2 实例和数据库实例

如果您使用 Amazon CloudFormation 创建了资源，请在连接并浏览 EC2 实例和数据库实例示例之后，删除 CloudFormation 堆栈，以便不再为其付费。

删除 CloudFormation 资源

1. 打开 Amazon CloudFormation 控制台。
2. 在 CloudFormationconsole 的堆栈页面上，选择根堆栈（名称不是 VPCStack、BastionStack 或 RDSNS 的堆栈）。
3. 选择删除。
4. 提示进行确认时，选择删除堆栈。

有关如何删除 CloudFormation 中堆栈的信息，请参阅《Amazon CloudFormation 用户指南》中的在 Amazon CloudFormation 控制台上删除堆栈。

（可选）将您的数据库实例连接到 Lambda 函数

您也可以将您的 RDS for Oracle 数据库实例连接到 Lambda 无服务器计算资源。Lambda 函数支持在不预调配或管理基础设施的情况下运行代码。Lambda 函数还支持自动响应任何规模的代码执行请求，从每天十几个事件到每秒数百个事件。有关更多信息，请参阅 自动连接 Lambda 函数和数据库实例。

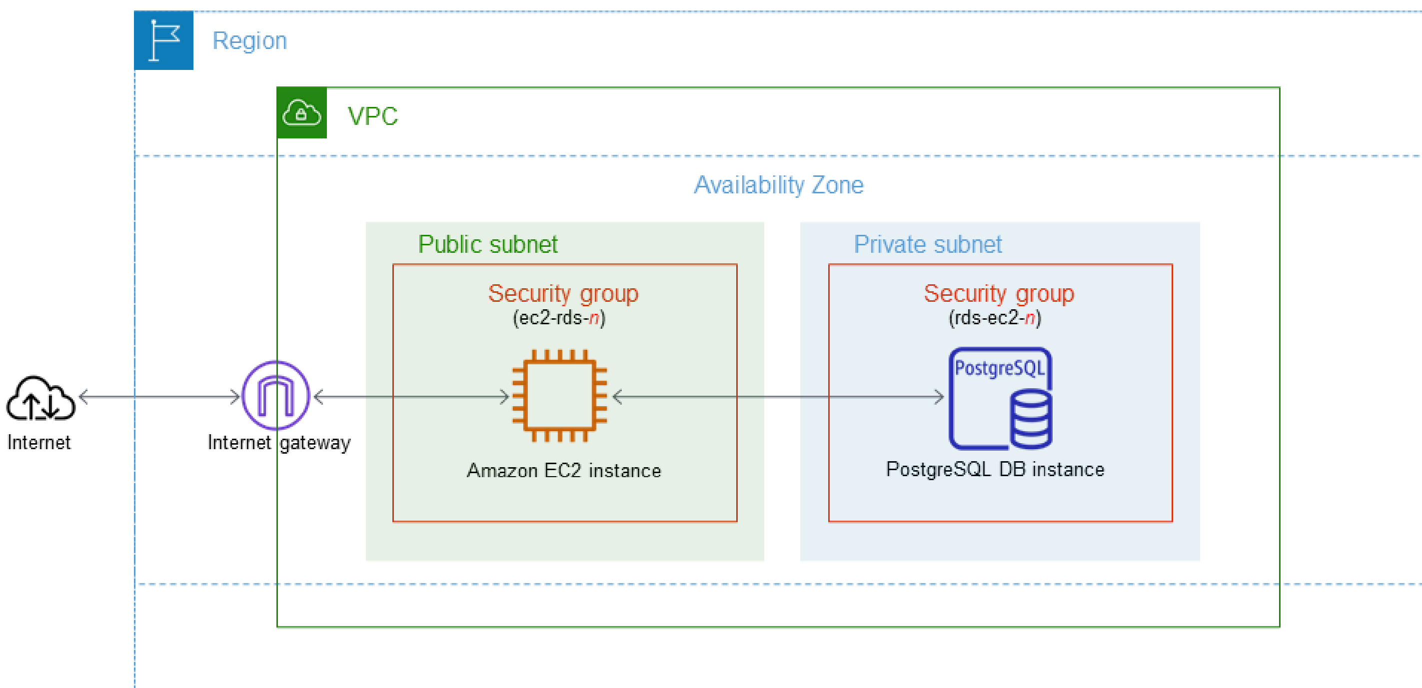
创建 PostgreSQL 数据库实例并连接到该实例

本教程创建一个 EC2 实例和一个 RDS for PostgreSQL 数据库实例。本教程向您展示如何使用标准 PostgreSQL 客户端从 EC2 实例访问数据库实例。作为最佳实践，本教程在虚拟私有云（VPC）中创建了私有数据库实例。在大多数情况下，同一 VPC 中的其他资源（例如 EC2 实例）可以访问数据库实例，但 VPC 之外的资源无法访问该实例。

完成本教程后，VPC 的每个可用区中都有一个公有和私有子网。在一个可用区中，EC2 实例在公有子网中，数据库实例在私有子网中。

> **Important**
>
> 创建 Amazon 账户并不会收费；但是，在完成本教程过程中使用 Amazon 资源可能会产生费用。完成本教程后，如果不再需要这些资源，可以将其删除。

下图显示了教程完成时的配置。



本教程可帮助您通过以下方法之一创建资源：

1. 使用 Amazon Web Services Management Console - [创建 EC2 实例](#) 和 [创建 PostgreSQL 数据库实例](#)

2. 使用 Amazon CloudFormation 创建数据库实例和 EC2 实例 - （可选）使用 Amazon CloudFormation 创建 VPC、EC2 实例和 PostgreSQL 实例

第一种方法使用轻松创建，通过 Amazon Web Services Management Console 创建私有 PostgreSQL 数据库实例。您可以仅指定数据库引擎类型、数据库实例大小和数据库实例标识符。轻松创建为其他配置选项使用默认设置。

如果改用标准创建，则在创建数据库实例时可以指定更多配置选项。这些选项包括可用性、安全性、备份和维护设置。要创建公有数据库实例，必须使用标准创建。有关信息，请参阅创建 Amazon RDS 数据库实例。

主题

先决条件

在开始之前，请完成以下各节中的步骤：

- 注册 Amazon Web Services 账户
- 保护 IAM 用户

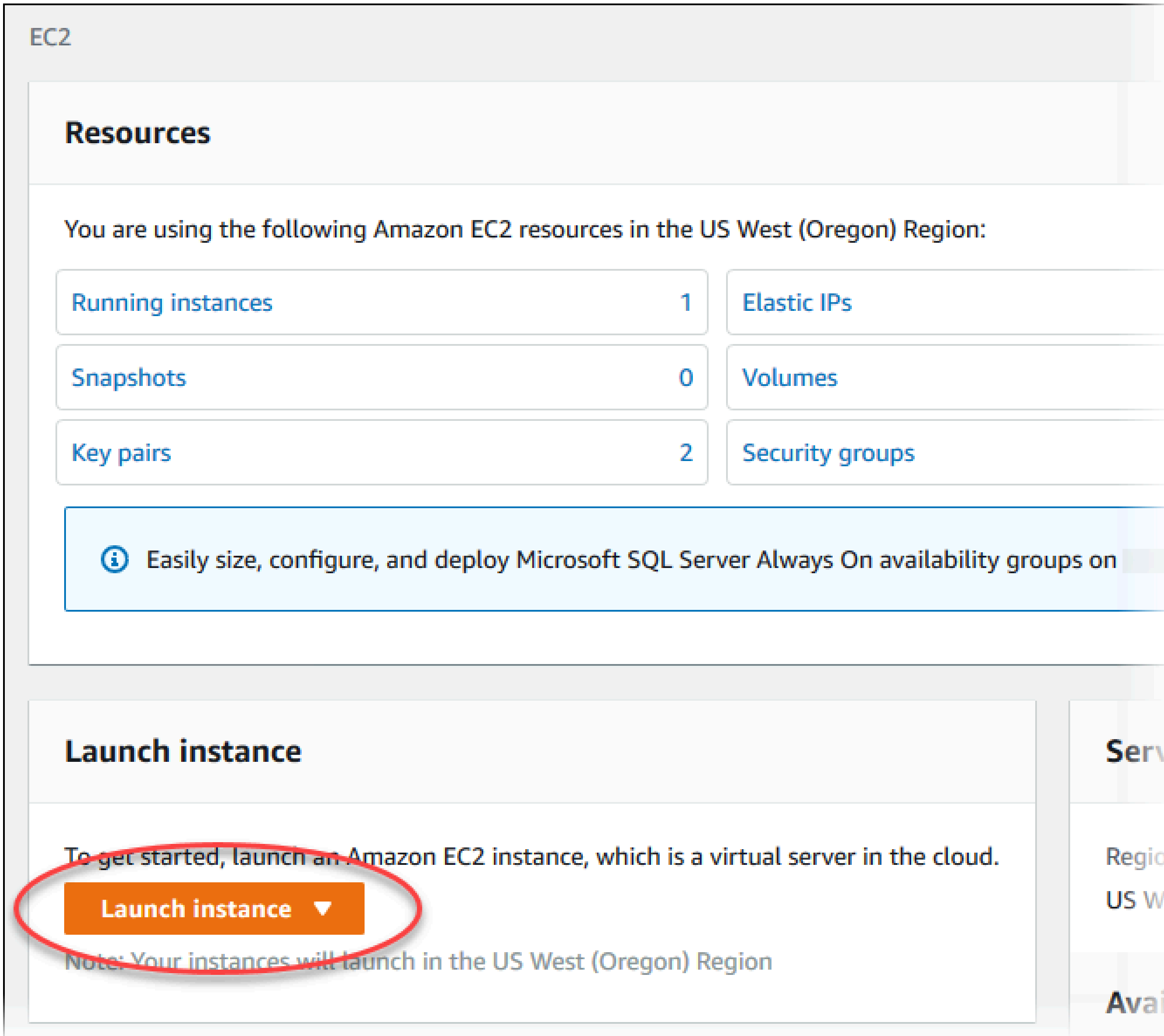
创建 EC2 实例

创建将用于连接到数据库的 Amazon EC2 实例。

创建 EC2 实例

1. 登录到 Amazon Web Services Management Console 并打开 Amazon EC2 控制台（https://console.aws.amazon.com/ec2/）。

2. 在 Amazon Web Services Management Console的右上角，选择要在其中创建 EC2 实例的 Amazon Web Services 区域。
3. 选择 EC2 控制面板，然后选择启动实例，如下图所示。

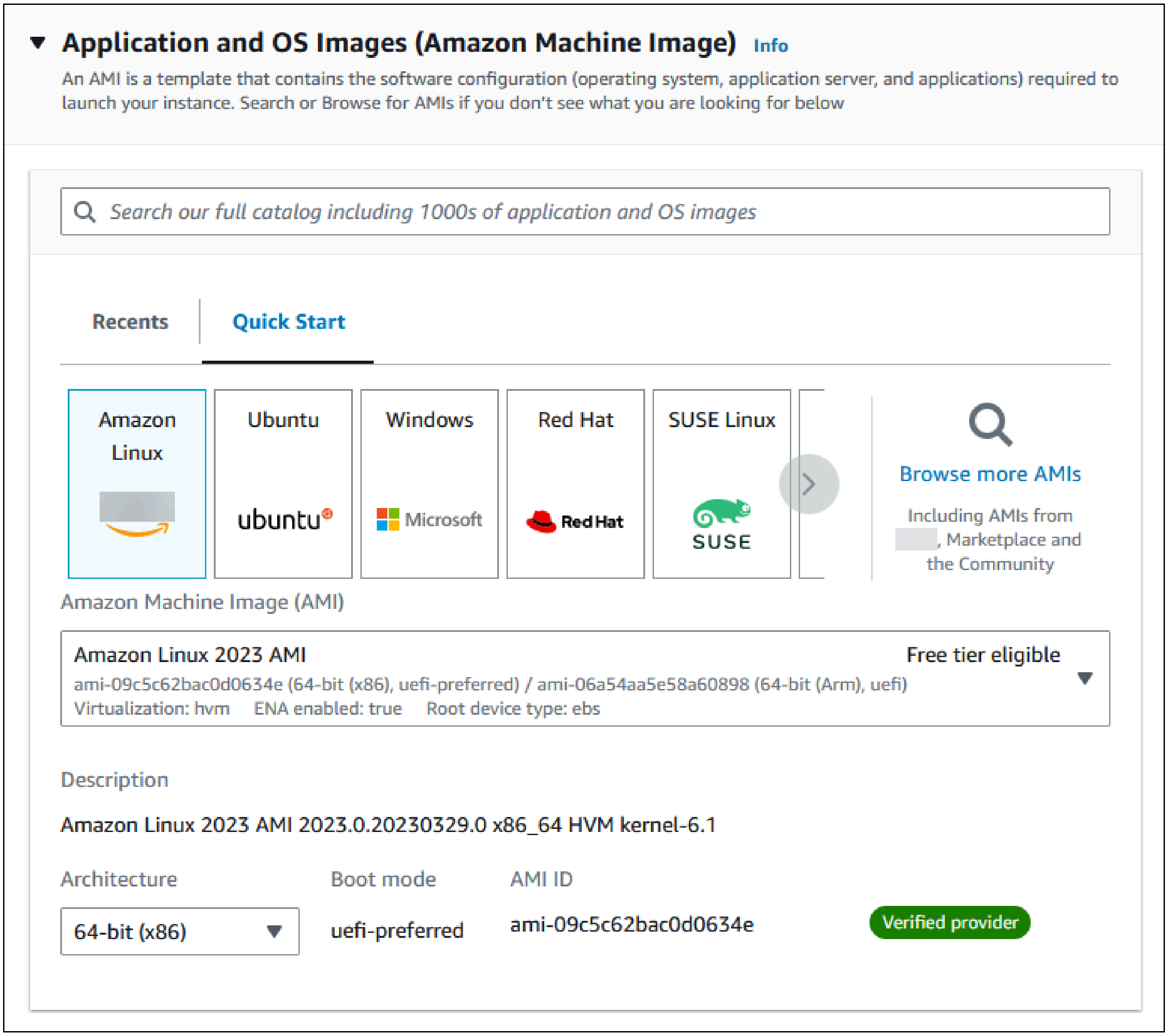


启动实例页面打开。

4. 在启动实例页面上选择以下设置。

   a. 在 Name and tags（名称和标签）下，对于 Name（名称），输入 **`ec2-database-connect`**。

   b. 在应用程序和操作系统映像（Amazon 机器映像）下，选择 Amazon Linux，然后选择 Amazon Linux 2023 AMI。对于其他选项，保留默认选择。



c. 在 Instance type（实例类型）下，选择 t2.micro。

d. 在 Key pair (login) [密钥对（登录）] 下，选择 Key pair name（密钥对名称）以使用现有密钥对。要为 Amazon EC2 实例创建新的密钥对，请选择 Create new key pair（创建新的密钥对），然后使用 Create key pair（创建密钥对）窗口来创建它。

   有关创建新的密钥对的更多信息，请参阅《Amazon EC2 用户指南》中的[创建密钥对]。

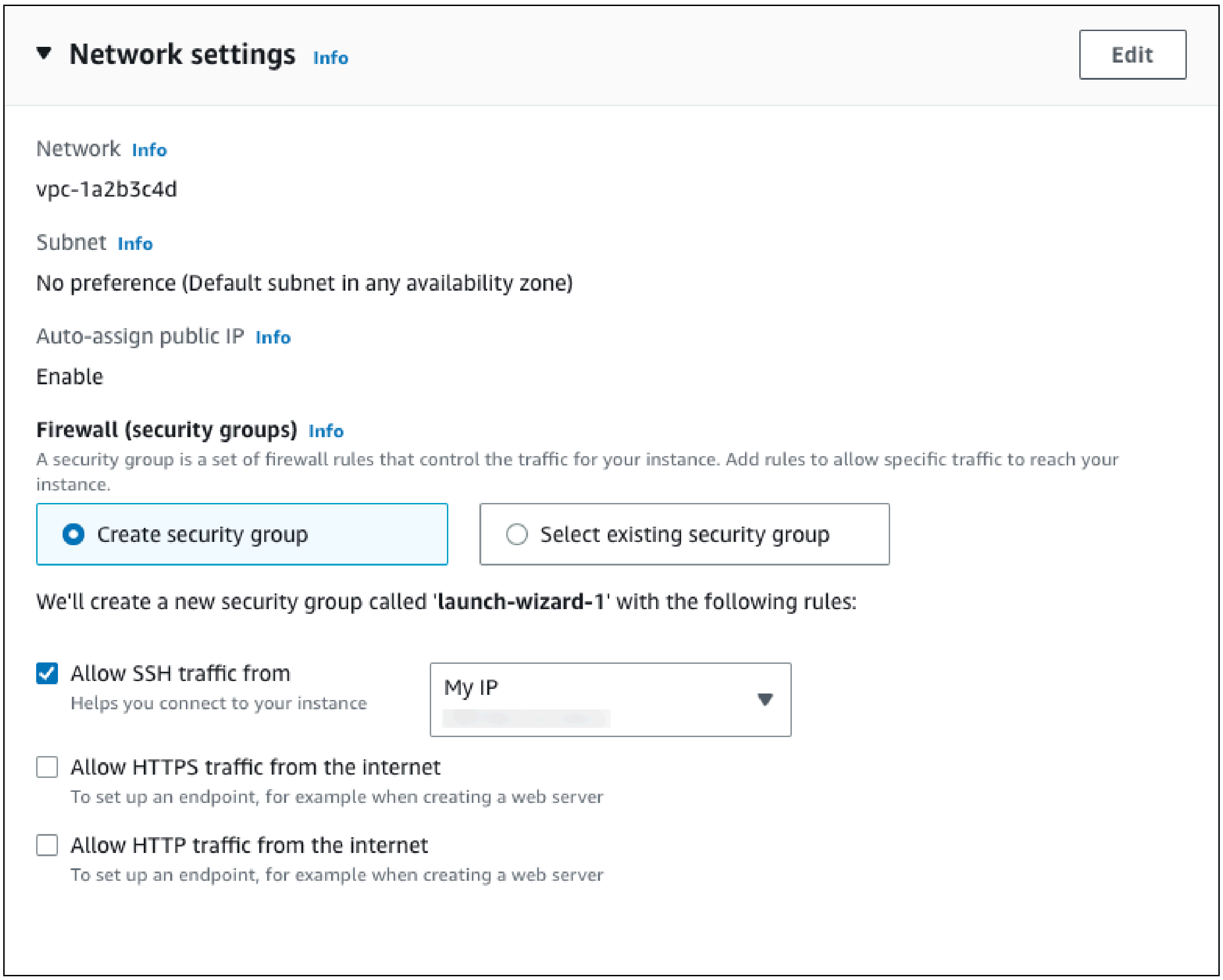
e. 对于网络设置中的允许 SSH 流量，选择 EC2 实例的 SSH 连接来源。

   如果显示的 IP 地址对于 SSH 连接是正确的，您可以选择 My IP（我的 IP）。否则，您可以确定要用来通过 Secure Shell（SSH）连接到 VPC 中的 EC2 实例的 IP 地址。要确定您的公有 IP 地址，请在新的浏览器窗口或标签页中，使用 https://checkip.amazonaws.com 上的服务。IP 地址的示例为 192.0.2.1/32。

在许多情况下，您可能通过互联网服务提供商（ISP）进行连接，或者在不使用静态 IP 地址的情况下从防火墙之后进行连接。如果是这样，请确保确定客户端计算机使用的 IP 地址范围。

> **Warning**
>
> 如果您使用 `0.0.0.0/0` 进行 SSH 访问，则所有 IP 地址可能能够使用 SSH 访问您的公有 EC2 实例。在测试环境下短时间内，此方法尚可接受，但它对于生产环境并不安全。在生产环境中，将仅向特定 IP 地址或地址范围授权使用 SSH 访问您的 EC2 实例。

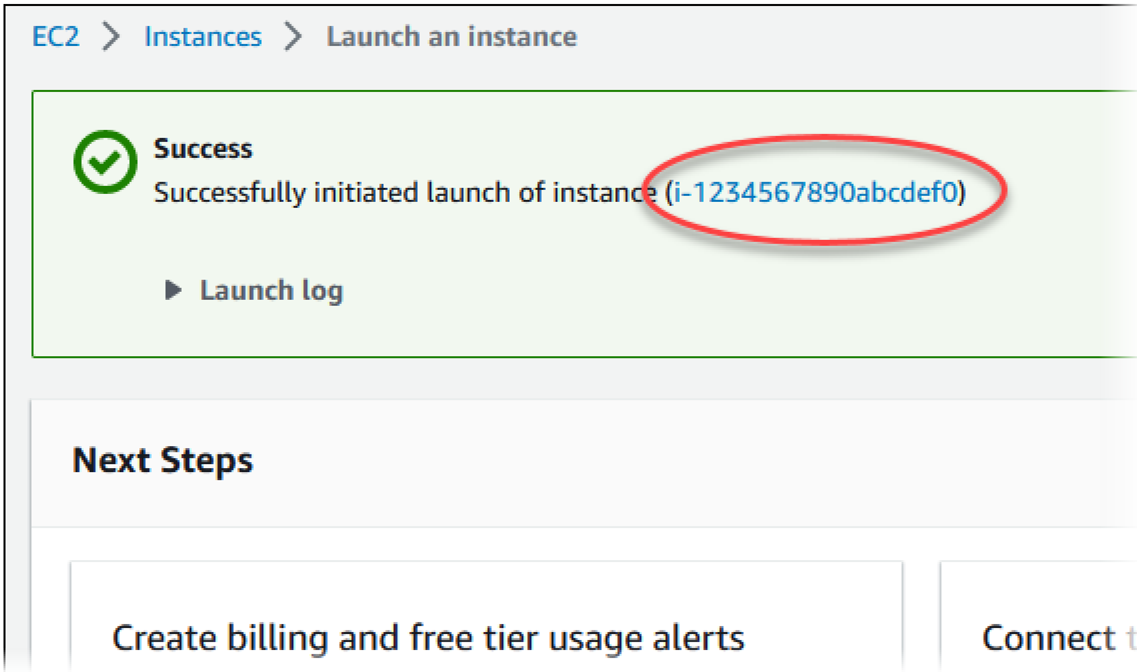
下图显示了网络设置部分的示例。



f. 对于其余部分保留默认值。

g. 查看摘要面板中您的 EC2 实例配置的摘要，当您准备好后，选择启动实例。

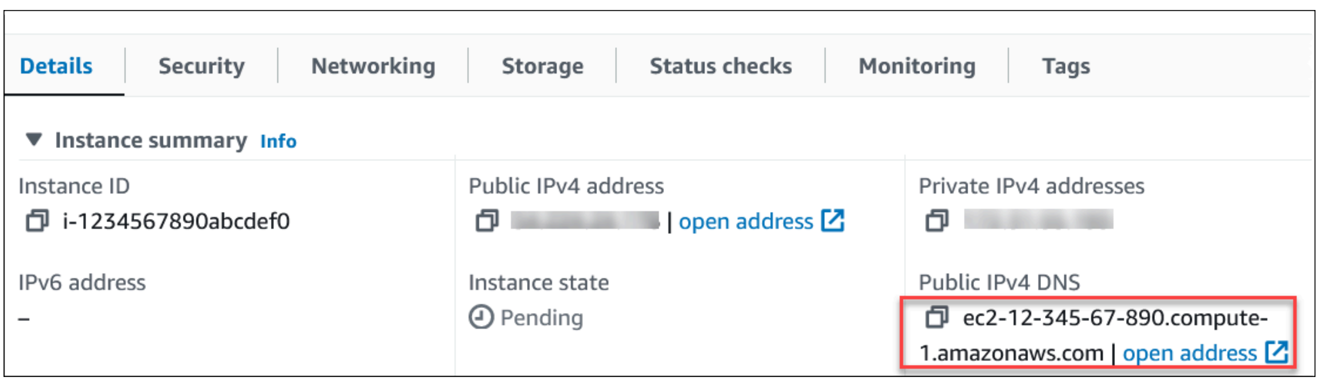
5. 在启动状态页面上，记下新 EC2 实例的标识符，例如：`i-1234567890abcdef0`。



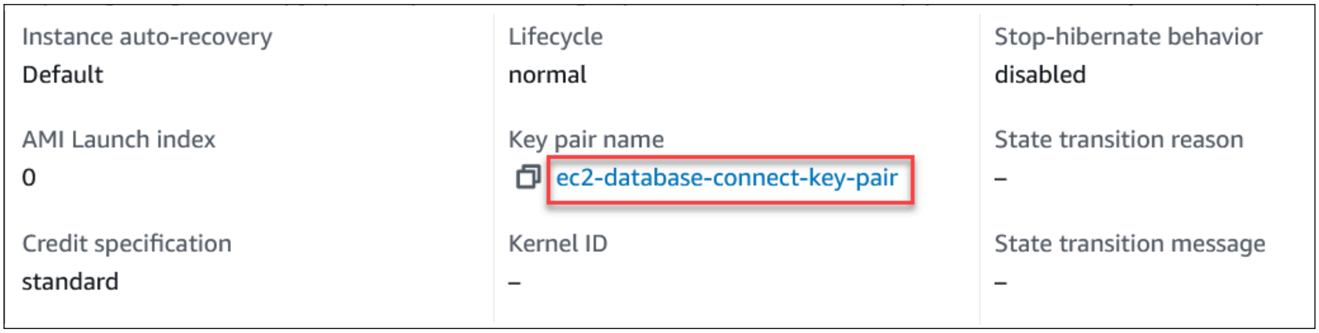
6. 选择 EC2 实例标识符以打开 EC2 实例列表，然后选择您的 EC2 实例。

7. 在详细信息选项卡中，记下使用 SSH 进行连接时所需的以下值：

a. 在实例摘要中，记下公有 IPv4 DNS 的值。



b. 在实例详细信息中，记下密钥对名称的值。



8. 等待 EC2 实例的实例状态变为正在运行，然后继续。

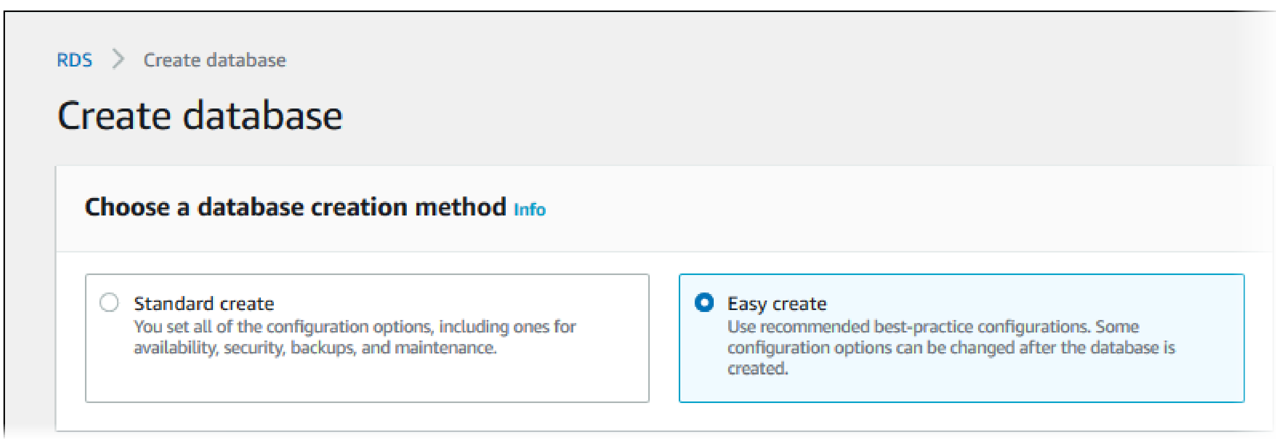
创建 PostgreSQL 数据库实例

Amazon RDS 的基本构建基块是数据库实例。这是您将要在其中运行 PostgreSQL 数据库的环境。

在该示例中，您使用轻松创建来创建一个数据库实例，该实例运行带有 db.t3.micro 数据库实例类的 PostgreSQL 数据库引擎。

在启用“Easy create (轻松创建)”的情况下创建 PostgreSQL 数据库实例

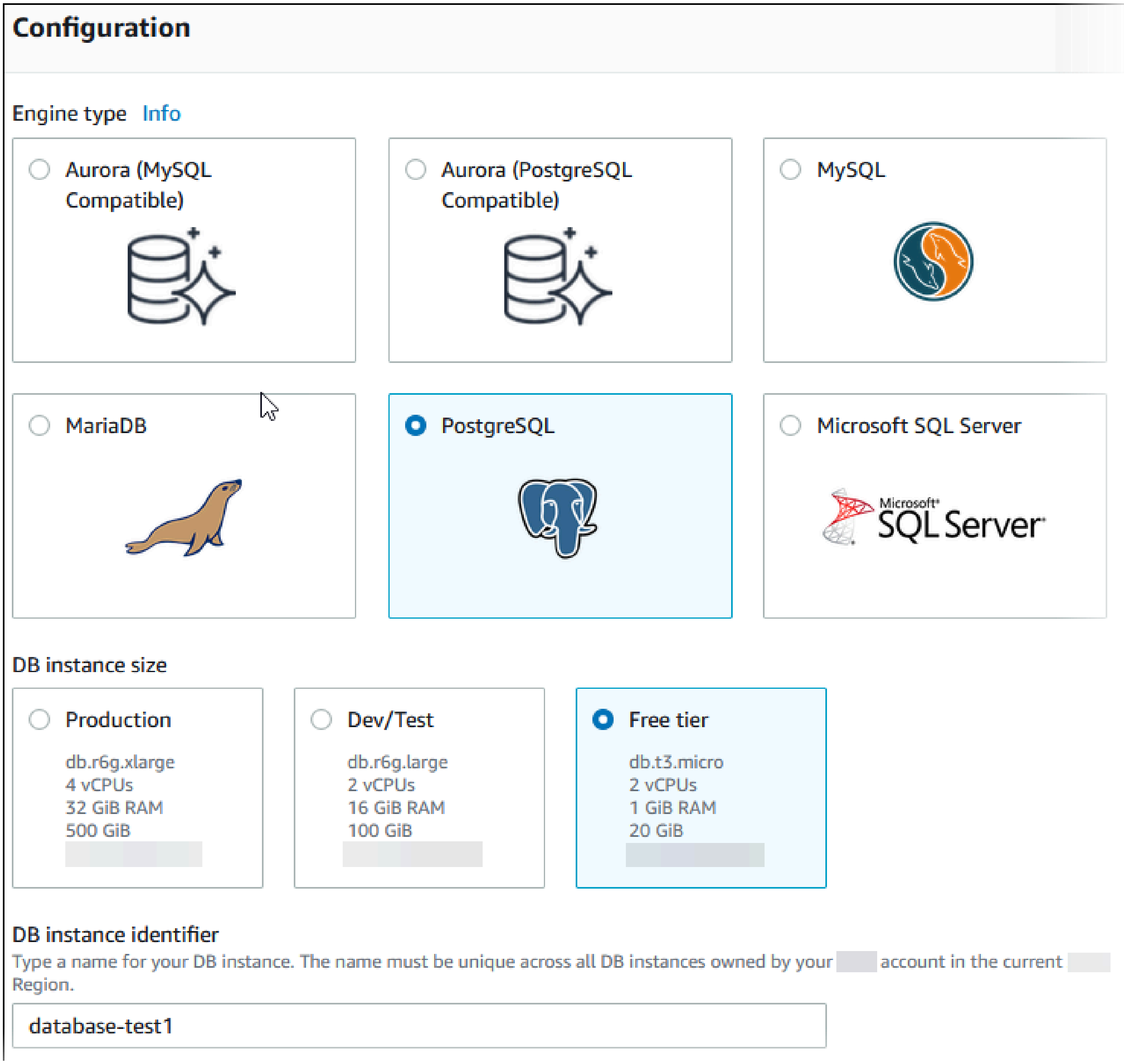
1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在 Amazon RDS 控制台的右上角，选择要在其中创建数据库实例的Amazon区域。
3. 在导航窗格中，选择 Databases (数据库)。
4. 选择 Create database (创建数据库)，并确保已选择 Easy create (轻松创建)。



5. 在配置中，选择 PostgreSQL。
6. 对于数据库实例大小，选择免费套餐。
7. 对于 DB instance identifier（数据库实例标识符），输入 **database-test1**。

8. 对于主用户名，输入主用户的名称，或者保留默认名称（**postgres**）。

   创建数据库页面应类似于以下图像。

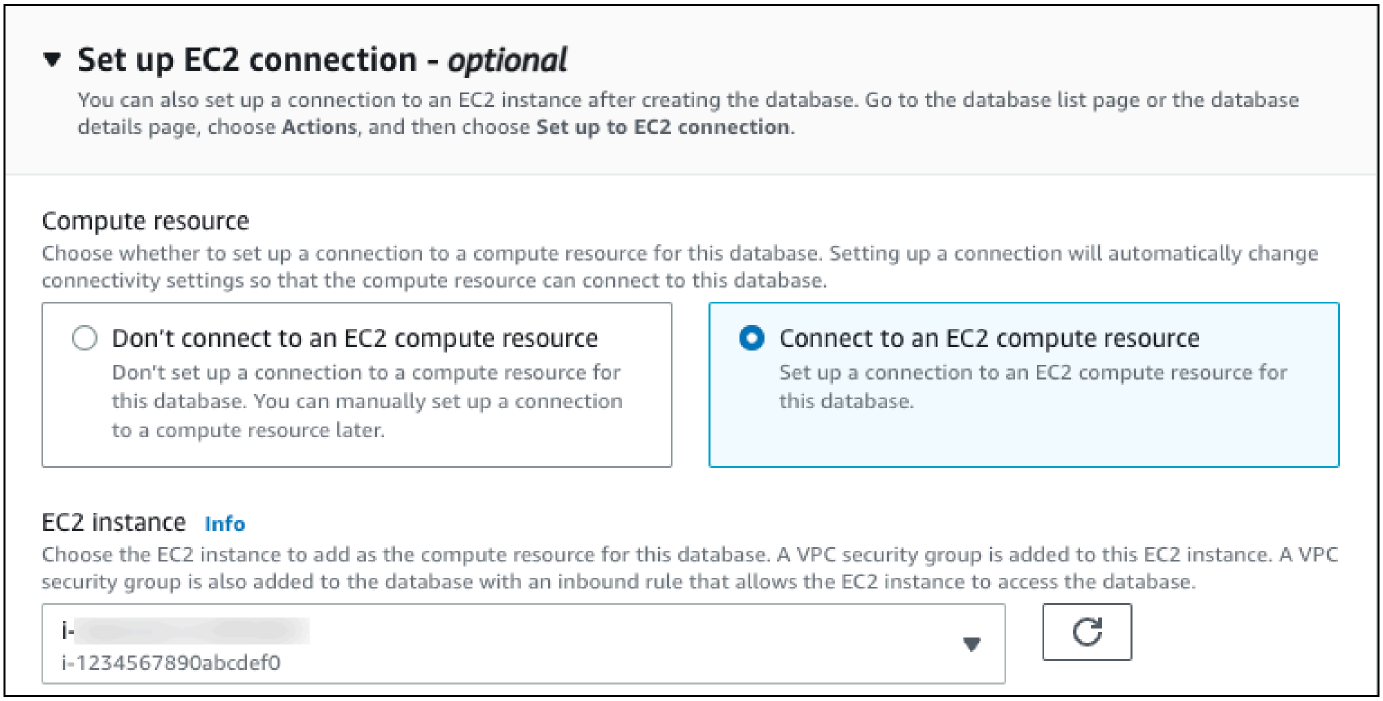


9. 要为数据库实例使用自动生成的主密码，请选择自动生成密码。

   要输入主密码，请确保清除自动生成密码，然后在主密码和确认密码中输入相同的密码。

10. 要设置与您之前创建的 EC2 实例的连接，请打开设置 EC2 连接 - 可选。

    选择连接到 EC2 计算资源。选择您之前创建的 EC2 实例。



11. 打开查看轻松创建的默认设置。

View default settings for Easy create

Easy create sets the following configurations to their default values, some of which can be changed later. If you want to change any of these settings now, use **Standard create**.

Configuration	Value	Editable after database is created
Encryption	Enabled	No
VPC	Default VPC (vpc-1a2b3c4d)	No
Option group	default:postgres-14	No
Subnet group	default	Yes
Automatic backups	Enabled	Yes
VPC security group	sg-1234567	Yes
Publicly accessible	No	Yes
Database port	5432	Yes
DB instance identifier	database-test1	Yes
DB engine version	14.6	Yes
DB parameter group	default.postgres14	Yes
Performance insights	Enabled	Yes
Monitoring	Enabled	Yes
Maintenance	Auto minor version upgrade enabled	Yes
Delete protection	Not enabled	Yes

您可以检查在启用 Easy create (轻松创建) 时使用的默认设置。可在创建数据库后编辑列显示在创建数据库后可以更改的选项。

- 如果该列中的某个设置为否，而您想要不同的设置，则可以使用标准创建来创建数据库实例。

- 如果某个设置在该列中为是，您可以使用标准创建来创建数据库实例，也可以在创建后修改数据库实例以更改该设置。

12. 选择创建数据库。

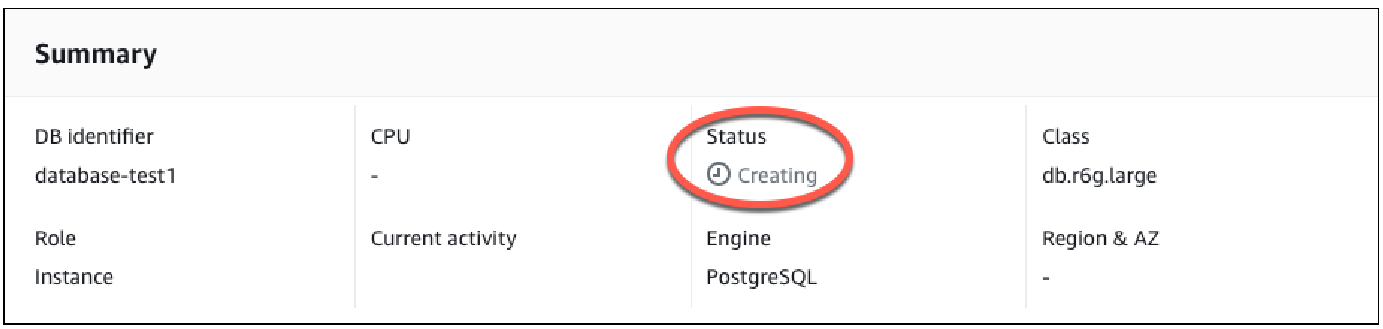
    要查看数据库实例的主用户名和密码，请选择查看凭证详细信息。

    您可以使用显示的用户名和密码，以主用户身份连接到数据库实例。

    > **Important**
    >
    > 您无法再次查看主用户密码。如果您不记录它，您可能需要更改它。
    >
    > 如果需要在数据库实例可用后更改主用户密码，则可以修改数据库实例以执行此操作。有关修改 数据库实例的更多信息，请参阅 修改 Amazon RDS 数据库实例。

13. 在数据库列表中，选择新的 PostgreSQL 数据库实例的名称以显示其详细信息。

    数据库实例具有正在创建状态，直到此实例就绪可供使用。



    当状态变为 Available（可用）时，您便可以连接到该数据库实例。根据数据库实例类和存储量，新实例可能需要等待 20 分钟时间才可用。

（可选）使用 Amazon CloudFormation 创建 VPC、EC2 实例和 PostgreSQL 实例

您可以使用 Amazon CloudFormation 通过将基础设施视为代码来配置 Amazon 资源，而无需使用控制台来创建 VPC、EC2 实例和 PostgreSQL 实例。为了帮助您将 Amazon 资源组织成更小、更易于管理的单元，您可以使用 Amazon CloudFormation 嵌套堆栈功能。有关更多信息，请参阅在 Amazon CloudFormation 控制台上创建堆栈和使用嵌套堆栈。

> **Important**
>
> Amazon CloudFormation 是免费的，但 CloudFormation 创建的资源是实时的。您需要为这些资源支付标准使用费，直到您终止使用它们为止。总费用将应该最少的。有关如何最大限度降低费用的信息，请转至 Amazon Free Tier。

要使用 Amazon CloudFormation 控制台创建资源，请执行以下步骤：

- 下载 CloudFormation 模板
- 使用 CloudFormation 配置资源

下载 CloudFormation 模板

CloudFormation 模板是一个 JSON 或 YAML 文本文件，其中包含有关您希望在堆栈中创建的资源的配置信息。此模板还为您创建 VPC 和堡垒主机以及 RDS 实例。

要下载模板文件，请打开以下链接 PostgreSQL CloudFormation 模板。

在 Github 页面中，单击 Download raw file 按钮以保存模板 YAML 文件。

使用 CloudFormation 配置资源

> **Note**
>
> 在开始此过程之前，请确保您的 Amazon Web Services 账户中具有 EC2 实例密钥对。有关更多信息，请参阅 Amazon EC2 密钥对和 Linux 实例。

使用 Amazon CloudFormation 模板时，必须选择正确的参数以确保正确创建资源。按以下步骤操作：

1. 登录到 Amazon Web Services Management Console 并打开 Amazon CloudFormation 控制台 https://console.aws.amazon.com/cloudformation。
2. 选择创建堆栈。
3. 在“指定模板”部分，选择从您的计算机上传模板文件，然后选择下一步。
4. 在指定堆栈详细信息页面上，设置以下参数：
    a. 将堆栈名称设置为 PostgreSQLTestStack。

b. 在参数下，通过选择三个可用区来设置可用区。

c. 在 Linux 堡垒主机配置下，在密钥名称中，选择用于登录您的 EC2 实例的密钥对。

d. 在 Linux 堡垒主机配置设置中，将允许的 IP 范围设置为您的 IP 地址。要使用 Secure Shell（SSH）连接到 VPC 中的 EC2 实例，请通过 https://checkip.amazonaws.com 上的服务确定您的公有 IP 地址。IP 地址的示例为 192.0.2.1/32。

> **Warning**
>
> 如果您使用 `0.0.0.0/0` 进行 SSH 访问，则所有 IP 地址可能能够使用 SSH 访问您的公有 EC2 实例。在测试环境下短时间内，此方法尚可接受，但它对于生产环境并不安全。在生产环境中，将仅向特定 IP 地址或地址范围授权使用 SSH 访问您的 EC2 实例。

e. 在数据库常规配置下，将数据库实例类设置为 db.t3.micro。

f. 将数据库名称设置为 **`database-test1`**。

g. 在数据库主用户名中，输入主用户的名称。

h. 在本教程中，将使用 Secrets Manager 管理数据库主用户密码设置为 `false`。

i. 在数据库密码中，设置所选密码。请记住此密码以便在教程中的后续步骤中使用。

j. 在数据库存储配置下，将数据库存储类型设置为 gp2。

k. 在数据库监控配置下，将启用 RDS 性能详情设置为 false。

l. 将所有其他设置保留为默认值。单击下一步继续。

5. 在配置堆栈选项页面中，保留所有默认选项。单击下一步继续。

6. 在查看堆栈页面中，在检查数据库和 Linux 堡垒主机选项后，选择提交。

堆栈创建过程完成后，查看名为 BastionStack 和 RDSNS 的堆栈，记下连接到数据库所需的信息。有关更多信息，请参阅在 Amazon Web Services Management Console 上查看 Amazon CloudFormation 堆栈数据和资源。

连接到 PostgreSQL 数据库实例

您可以使用 pgadmin 或 psql 连接到数据库实例。此示例说明如何使用 psql 命令行客户端连接到 PostgreSQL 数据库实例。

使用 psql 连接到 PostgreSQL 数据库实例

1. 找到您的数据库实例的端点（DNS 名称）和端口号。

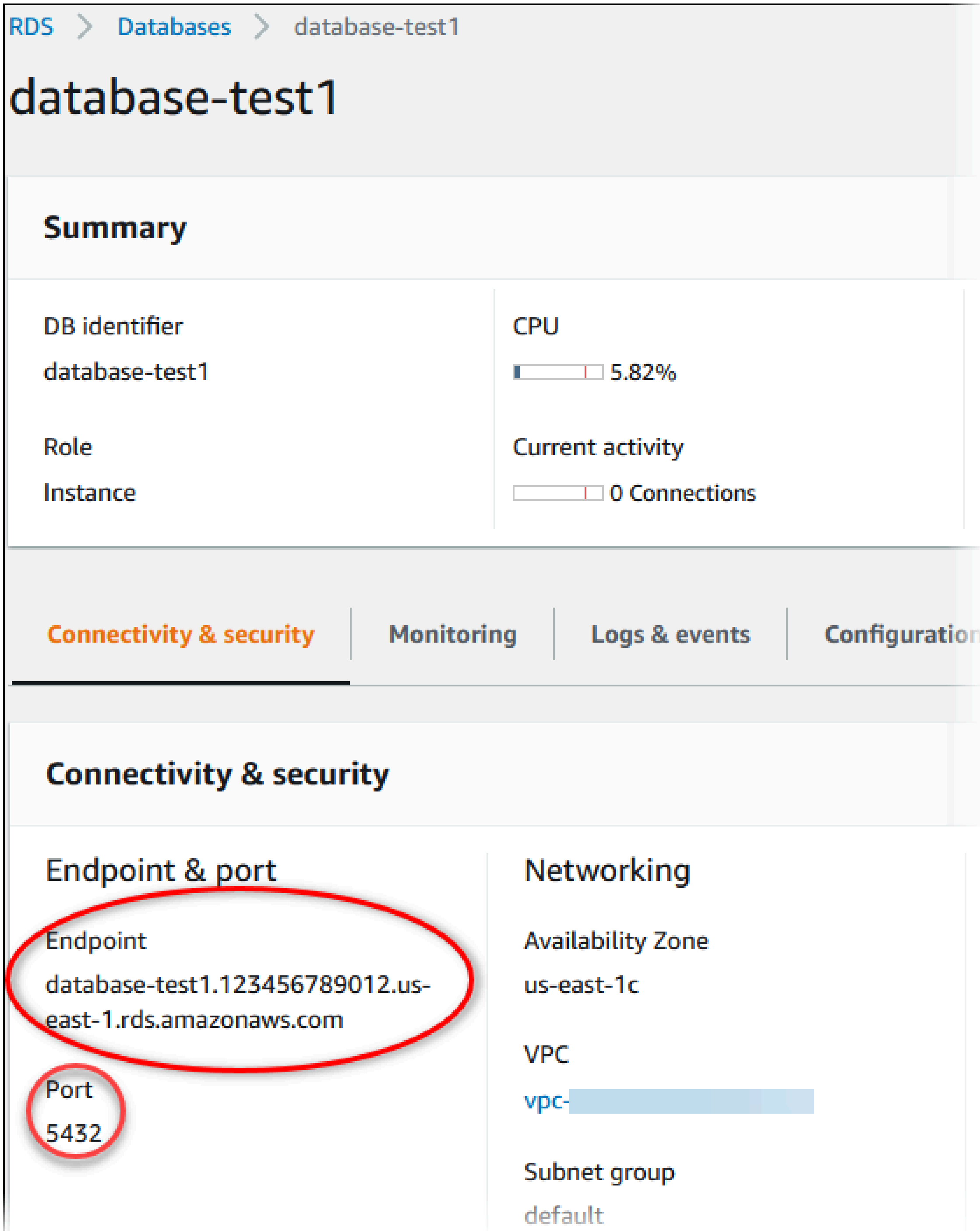
a. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。

b. 在 Amazon RDS 控制台的右上角，选择数据库实例的 Amazon Web Services 区域。

c. 在导航窗格中，选择 Databases（数据库）。

d. 选择 PostgreSQL 数据库实例名称以显示其详细信息。

e. 在连接和安全性选项卡上，复制端点。另请注意端口号。您需要端点和端口号才能连接到数据库实例。



2. 按照《Amazon EC2 用户指南》中的连接到 Linux 实例中的步骤操作，连接到您之前创建的 EC2 实例。

   我们建议您使用 SSH 连接到 EC2 实例。如果 SSH 客户端实用程序安装在 Windows、Linux 或 Mac 上，则可以使用以下命令格式连接到该实例：

```
ssh -i location_of_pem_file ec2-user@ec2-instance-public-dns-name
```

例如，假设在 Linux 上 `ec2-database-connect-key-pair.pem` 存储在 `/dir1` 中，而 EC2 实例的公有 IPv4 DNS 为 `ec2-12-345-678-90.compute-1.amazonaws.com`。SSH 命令将如下所示：

```
ssh -i /dir1/ec2-database-connect-key-pair.pem ec2-
user@ec2-12-345-678-90.compute-1.amazonaws.com
```

3. 通过更新 EC2 实例上的软件，获取最新的错误修复和安全更新。要执行此操作，请使用以下命令。

> **Note**
>
> `-y` 选项安装更新时不提示确认。要在安装前检查更新，请忽略该选项。

```
sudo dnf update -y
```

4. 要在 Amazon Linux 2023 上安装 PostgreSQL 中的 psql 命令行客户端，请运行以下命令：

```
sudo dnf install postgresql15
```

5. 连接到 PostgreSQL 数据库实例。例如，在客户端计算机上的命令提示符处输入以下命令。此操作可让您使用 psql 客户端连接到 PostgreSQL 数据库实例。

用数据库实例端点（DNS 名称）替换 *endpoint*，用要连接到的数据库名称 `--dbname` 替换 *postgres*，并用您使用的主用户名替换 *postgres*。提示输入密码时，提供所使用的主密码。

```
psql --host=endpoint --port=5432 --dbname=postgres --username=postgres
```

在输入用户的密码后，您应该会看到类似于以下内容的输出：

```
psql (14.3, server 14.6)
SSL connection (protocol: TLSv1.2, cipher: ECDHE-RSA-AES256-GCM-SHA384, bits: 256,
 compression: off)
Type "help" for help.
```

```
postgres=>
```

有关连接到 PostgreSQL 数据库实例的更多信息，请参阅连接到运行 PostgreSQL 数据库引擎的数据库实例。如果您无法连接到数据库实例，请参阅 排除与您的 RDS for PostgreSQL 实例的连接故障。

出于安全考虑，最佳做法是使用加密连接。仅当客户端和服务器位于同一 VPC 中，并且网络受信任时，才会使用未加密的 PostgreSQL 连接。有关使用加密连接的信息，请参阅 通过 SSL 连接到 PostgreSQL 数据库实例。

6. 运行 SQL 命令。

   例如，以下 SQL 命令显示了当前日期和时间：

```
SELECT CURRENT_TIMESTAMP;
```

删除 EC2 实例和数据库实例

在连接到并浏览您创建的示例 EC2 实例和数据库实例之后，删除它们，以便不再为其付费。

如果您使用 Amazon CloudFormation 创建了资源，请跳过此步骤，转至下一步。

删除 EC2 实例

1. 登录到 Amazon Web Services Management Console 并打开 Amazon EC2 控制台（https://console.aws.amazon.com/ec2/）。
2. 在导航窗格中，选择实例。
3. 选择 EC2 实例，然后依次选择实例状态、终止实例。
4. 当系统提示您确认时，选择终止。

有关删除 EC2 实例的更多信息，请参阅《Amazon EC2 用户指南》中的终止实例。

删除无最终数据库快照的数据库实例

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。
3. 选择要删除的数据库实例。

4. 对于 Actions，选择 Delete。
5. 清除是否创建最终快照？和保留自动备份。
6. 完成确认并选择删除。

（可选）删除使用 CloudFormation 创建的 EC2 实例和数据库实例

如果您使用 Amazon CloudFormation 创建了资源，请在连接并浏览 EC2 实例和数据库实例示例之后，删除 CloudFormation 堆栈，以便不再为其付费。

删除 CloudFormation 资源

1. 打开 Amazon CloudFormation 控制台。
2. 在 CloudFormationconsole 的堆栈页面上，选择根堆栈（名称不是 VPCStack、BastionStack 或 RDSNS 的堆栈）。
3. 选择删除。
4. 提示进行确认时，选择删除堆栈。

有关如何删除 CloudFormation 中堆栈的信息，请参阅《Amazon CloudFormation 用户指南》中的在 Amazon CloudFormation 控制台上删除堆栈。

（可选）将您的数据库实例连接到 Lambda 函数

您也可以将您的 RDS for PostgreSQL 数据库实例连接到 Lambda 无服务器计算资源。Lambda 函数支持在不预调配或管理基础设施的情况下运行代码。Lambda 函数还支持自动响应任何规模的代码执行请求，从每天十几个事件到每秒数百个事件。有关更多信息，请参阅 自动连接 Lambda 函数和数据库实例。

教程：创建 Web 服务器和 Amazon RDS 数据库实例

本教程说明如何使用 PHP 安装 Apache Web 服务器并创建 MariaDB、MySQL 或 PostgreSQL 数据库。Web 服务器在使用 Amazon Linux 2023 的 Amazon EC2 实例上运行，您可以在 MySQL 或 PostgreSQL 数据库实例之间进行选择。Amazon EC2 实例和数据库实例均在基于 Amazon VPC 服务的 Virtual Private Cloud (VPC) 中运行。

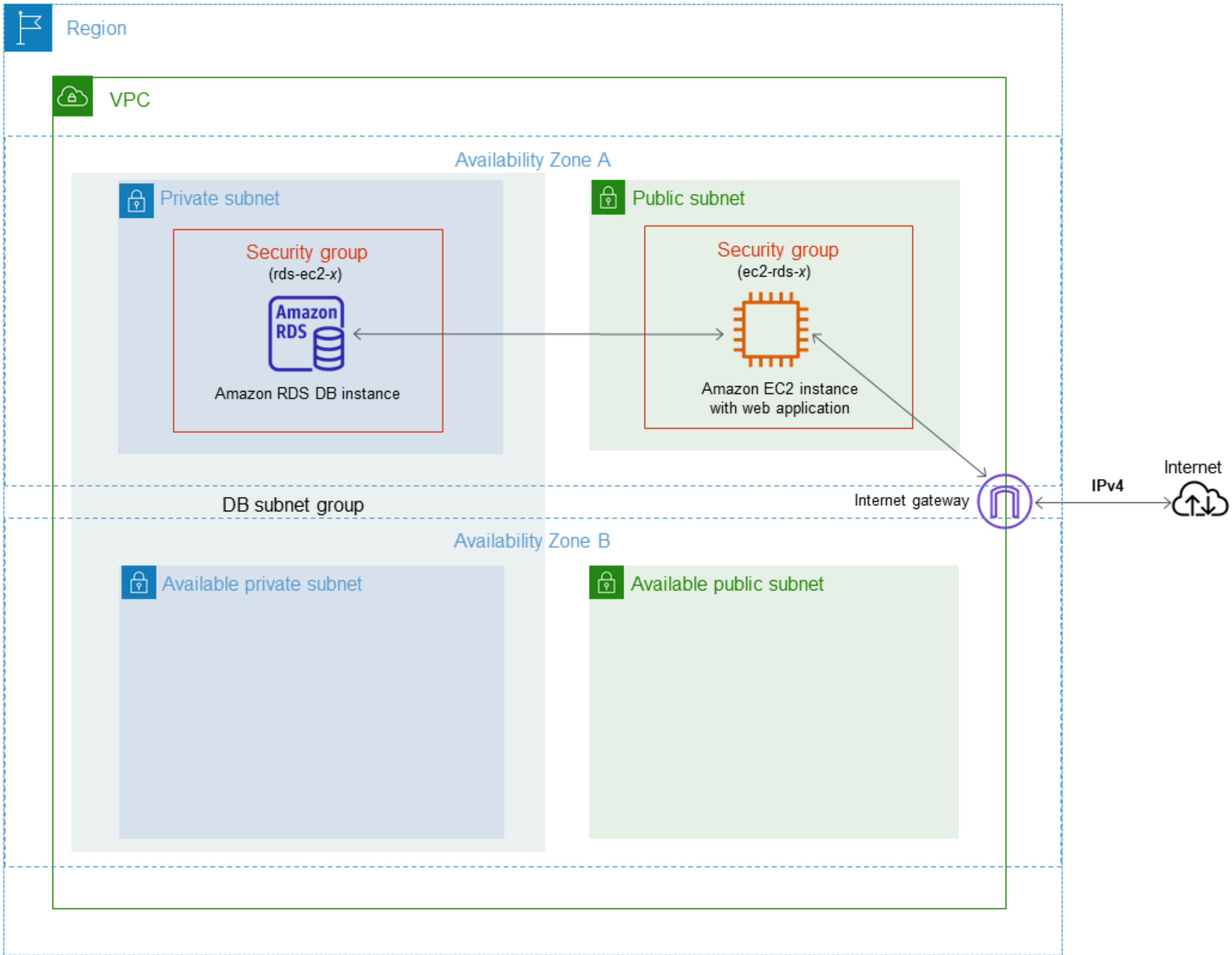
> **Important**
>
> 创建 Amazon 账户并不会收费；但是，在完成本教程过程中使用 Amazon 资源可能会产生费用。完成本教程后，如果不再需要这些资源，可以将其删除。

> **Note**
>
> 本教程适用于 Amazon Linux 2023，可能不适用于其他版本的 Linux。

在下面的教程中，您将创建一个对您的 Amazon Web Services 账户使用默认 VPC、子网和安全组的 EC2 实例。本教程说明如何创建数据库实例，并自动设置与您创建的 EC2 实例的连接。然后，本教程向您展示如何在 EC2 实例上安装 Web 服务器。您使用数据库实例端点，将 Web 服务器连接到 VPC 中的数据库实例。

1. 启动 EC2 实例以连接数据库实例
2. 创建 Amazon RDS 数据库实例
3. 在 EC2 实例上安装 Web 服务器

下图显示了教程完成时的配置。



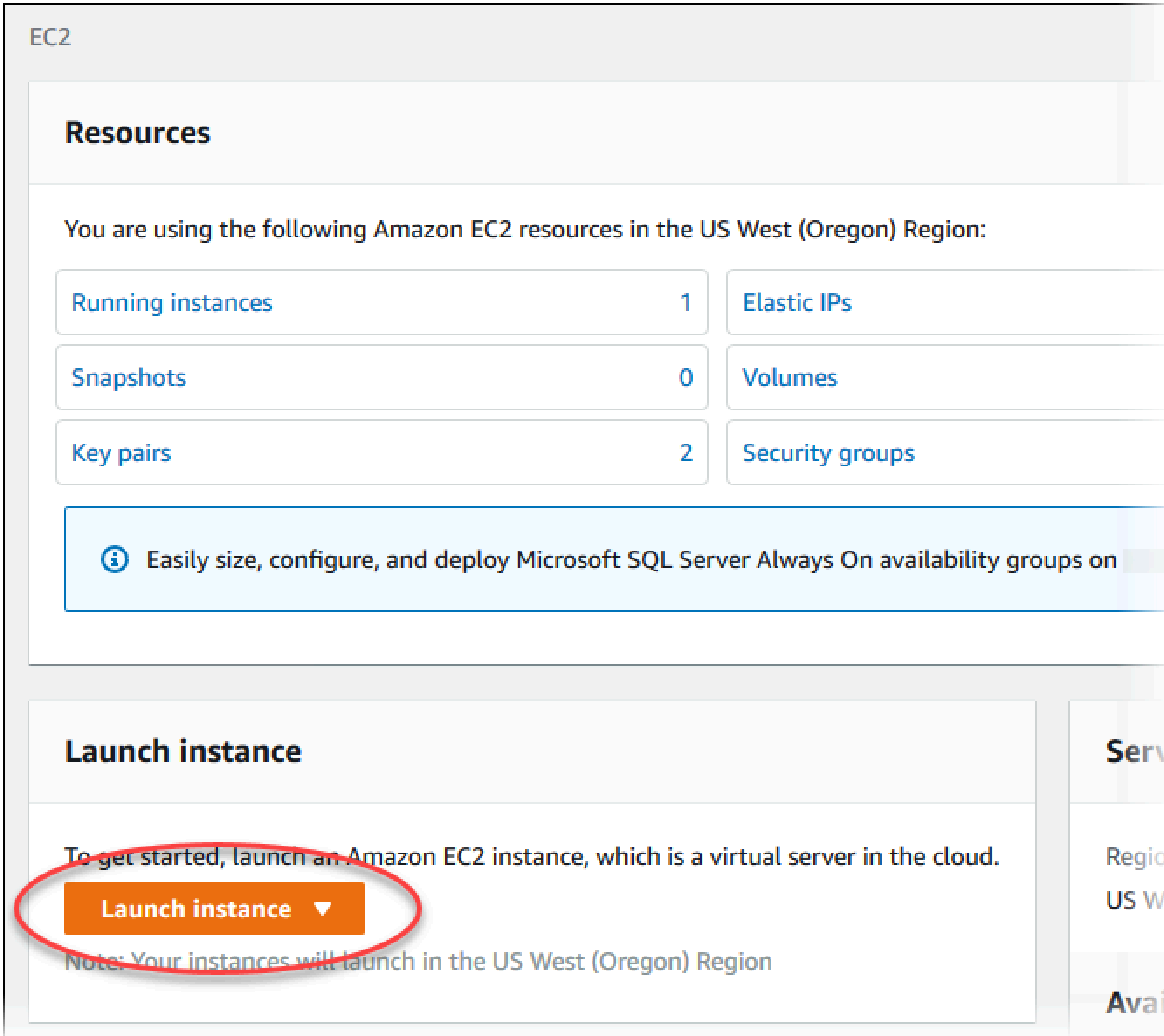
> **Note**
>
> 完成本教程后，VPC 的每个可用区中都有一个公有和私有子网。本教程对您的 Amazon Web Services 账户使用默认 VPC，并自动设置您的 EC2 实例与数据库实例之间的连接。如果您更愿意为此场景配置新的 VPC，请完成教程：创建 VPC 以用于数据库实例（仅限 IPv4）中的任务。

启动 EC2 实例以连接数据库实例

在您的 VPC 的公有子网中创建 Amazon EC2 实例。

启动 EC2 实例

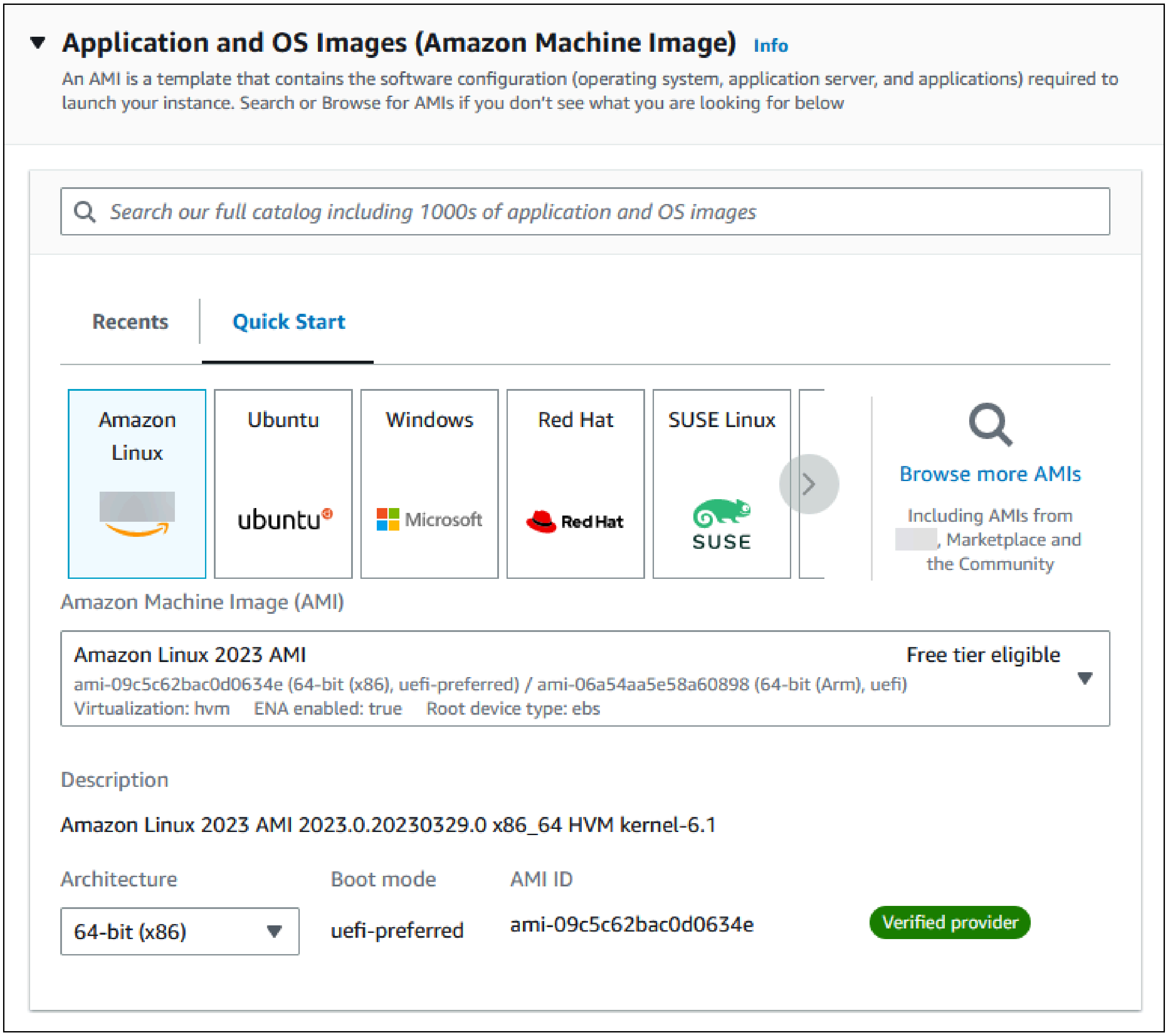
1. 登录 Amazon Web Services Management Console，打开 Amazon EC2 控制台：https://console.aws.amazon.com/ec2/。
2. 在 Amazon Web Services Management Console的右上角，选择要在其中创建 EC2 实例的 Amazon Web Services 区域。
3. 选择 EC2 控制面板，然后选择启动实例，如下所示。



4. 在启动实例页面中选择以下设置。

   a. 在 Name and tags（名称和标签）下，对于 Name（名称），输入 **`tutorial-ec2-instance-web-server`**。

b. 在应用程序和操作系统映像（Amazon 机器映像）下，选择 Amazon Linux，然后选择 Amazon Linux 2023 AMI。对于其他选项保留原定设置值。



c. 在 Instance type（实例类型）下，选择 t2.micro。

d. 在 Key pair (login) [密钥对（登录）] 下，选择 Key pair name（密钥对名称）以使用现有密钥对。要为 Amazon EC2 实例创建新的密钥对，请选择 Create new key pair（创建新的密钥对），然后使用 Create key pair（创建密钥对）窗口来创建它。

有关创建新的密钥对的更多信息，请参阅《Amazon EC2 用户指南》中的创建密钥对。

e. 在 Network settings（网络设置）下，设置以下值并将其他值保留为其原定设置值：

• 对于 Allow SSH traffic from（允许 SSH 流量来自），选择 EC2 实例的 SSH 连接来源。

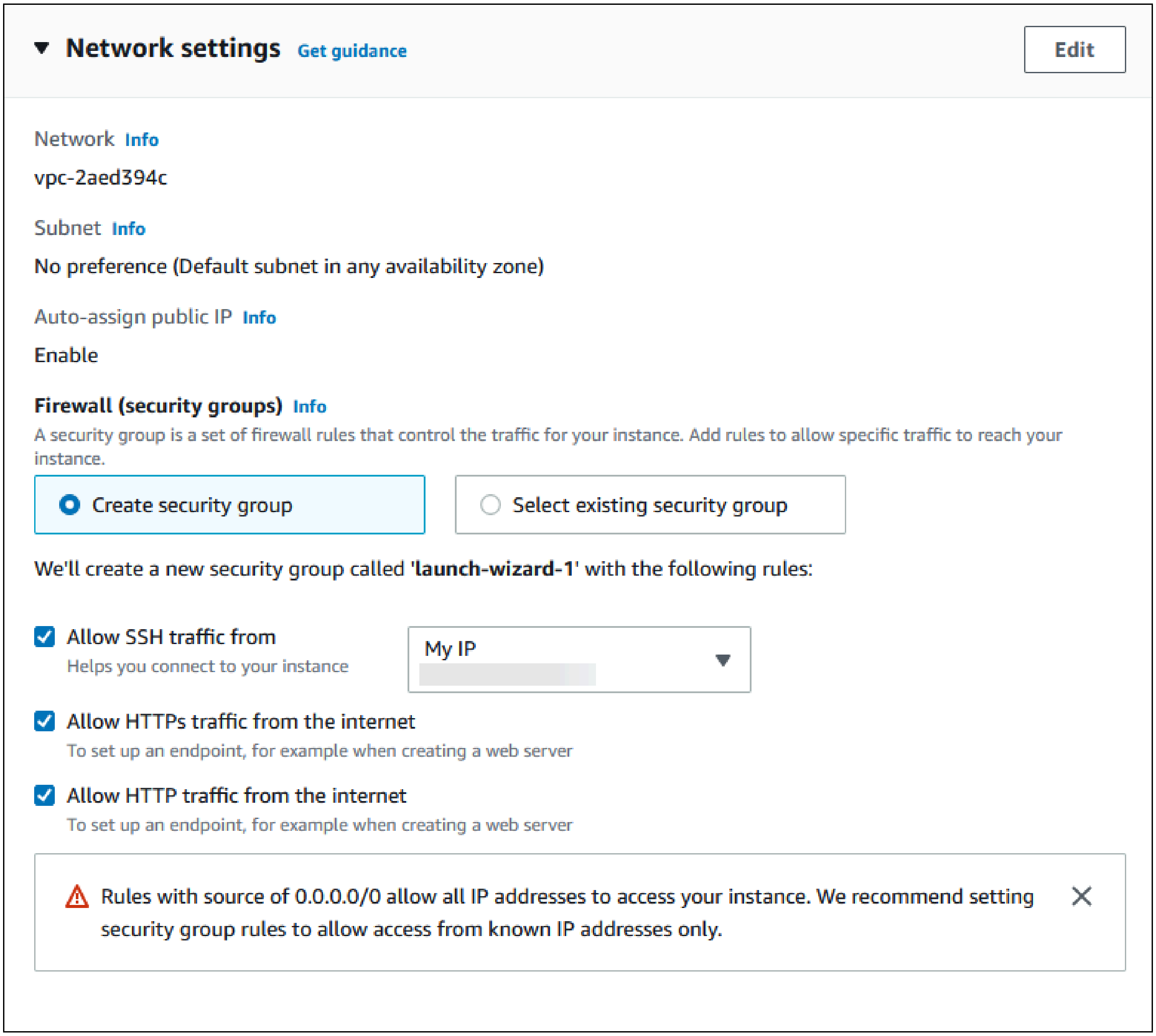
如果显示的 IP 地址对于 SSH 连接是正确的，您可以选择 My IP（我的 IP）。

否则，您可以确定要用来通过 Secure Shell（SSH）连接到 VPC 中的 EC2 实例的 IP 地址。要确定您的公有 IP 地址，请在新的浏览器窗口或标签页中，使用 https://checkip.amazonaws.com 上的服务。IP 地址的一个示例为 `203.0.113.25/32`。

在许多情况下，您可能通过互联网服务提供商（ISP）进行连接，或者在不使用静态 IP 地址的情况下从防火墙之后进行连接。如果是这样，请确保确定客户端计算机使用的 IP 地址范围。

> **Warning**
>
> 如果您使用 `0.0.0.0/0` 进行 SSH 访问，则所有 IP 地址可能能够使用 SSH 访问您的公有实例。在测试环境下短时间内，此方法尚可接受，但它对于生产环境并不安全。在生产环境中，将仅向特定 IP 地址或地址范围授权使用 SSH 访问您的实例。

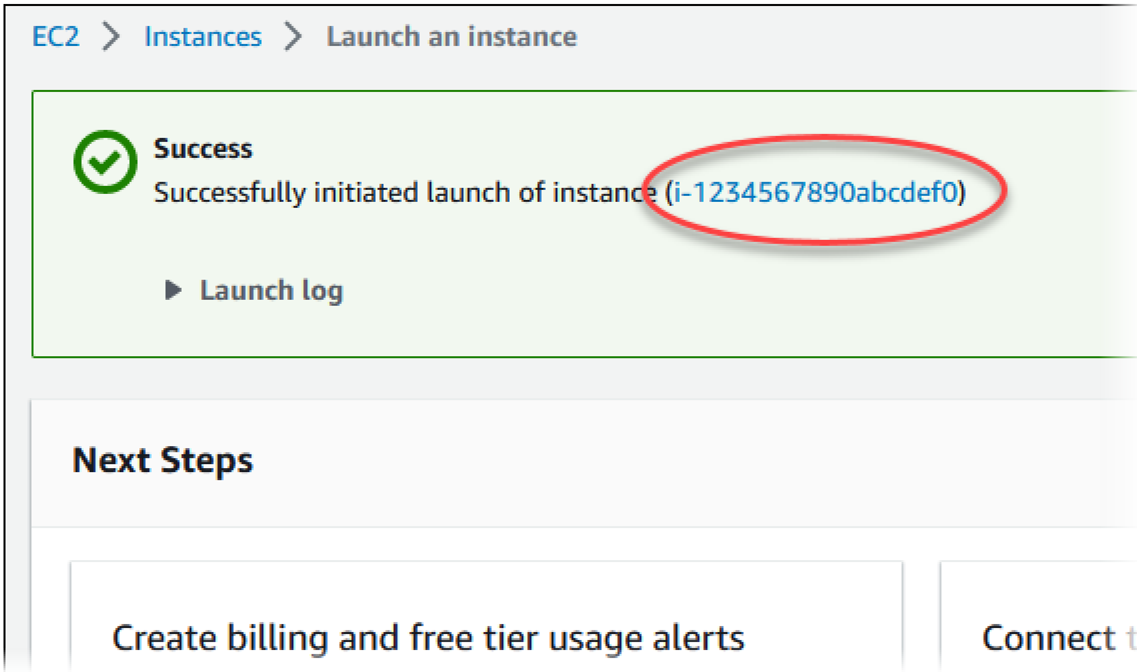
- 开启 Allow HTTPs traffic from the internet（允许来自互联网的 HTTPs 流量）。
- 开启 Allow HTTP traffic from the internet（允许来自互联网的 HTTP 流量）。



f. 对于其余部分保留原定设置值。

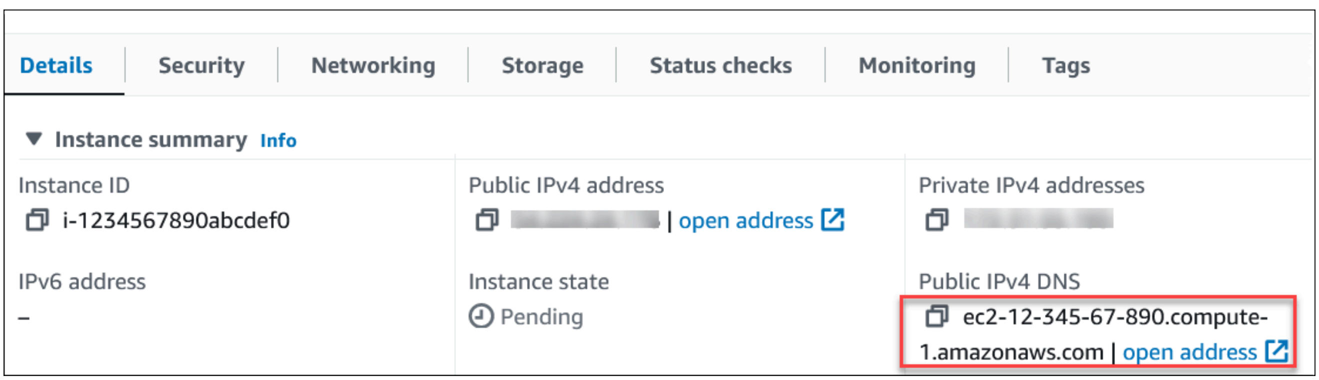
g. 查看 Summary（摘要）面板中您的实例配置的摘要，当您准备好后，选择 Launch instance（启动实例）。

5. 在启动状态页面上，记下新 EC2 实例的标识符，例如：`i-1234567890abcdef0`。

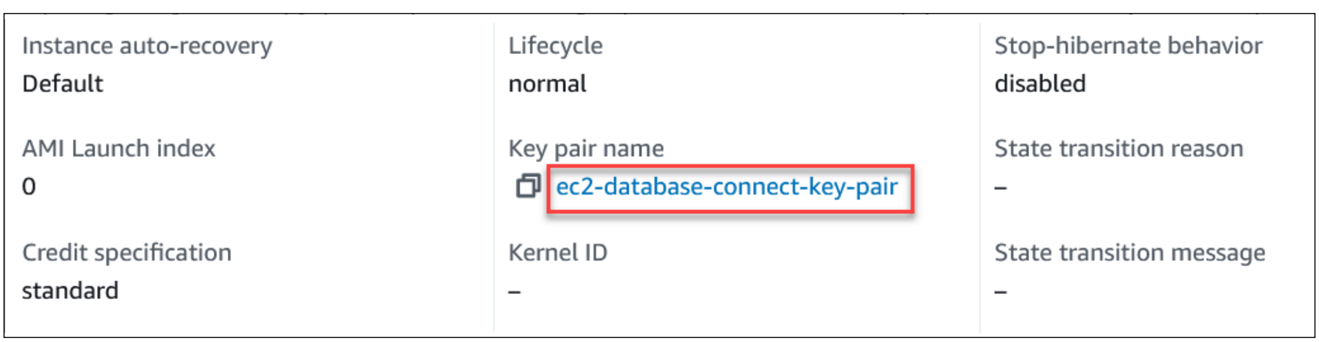


6. 选择 EC2 实例标识符以打开 EC2 实例列表，然后选择您的 EC2 实例。
7. 在详细信息选项卡中，记下使用 SSH 进行连接时所需的以下值：

   a. 在实例摘要中，记下公有 IPv4 DNS 的值。

   b. 在实例详细信息中，记下密钥对名称的值。

8. 一直等到实例的 Instance state（实例状态）变为 Running（正在运行），才能继续。

9. 完成 创建 Amazon RDS 数据库实例。

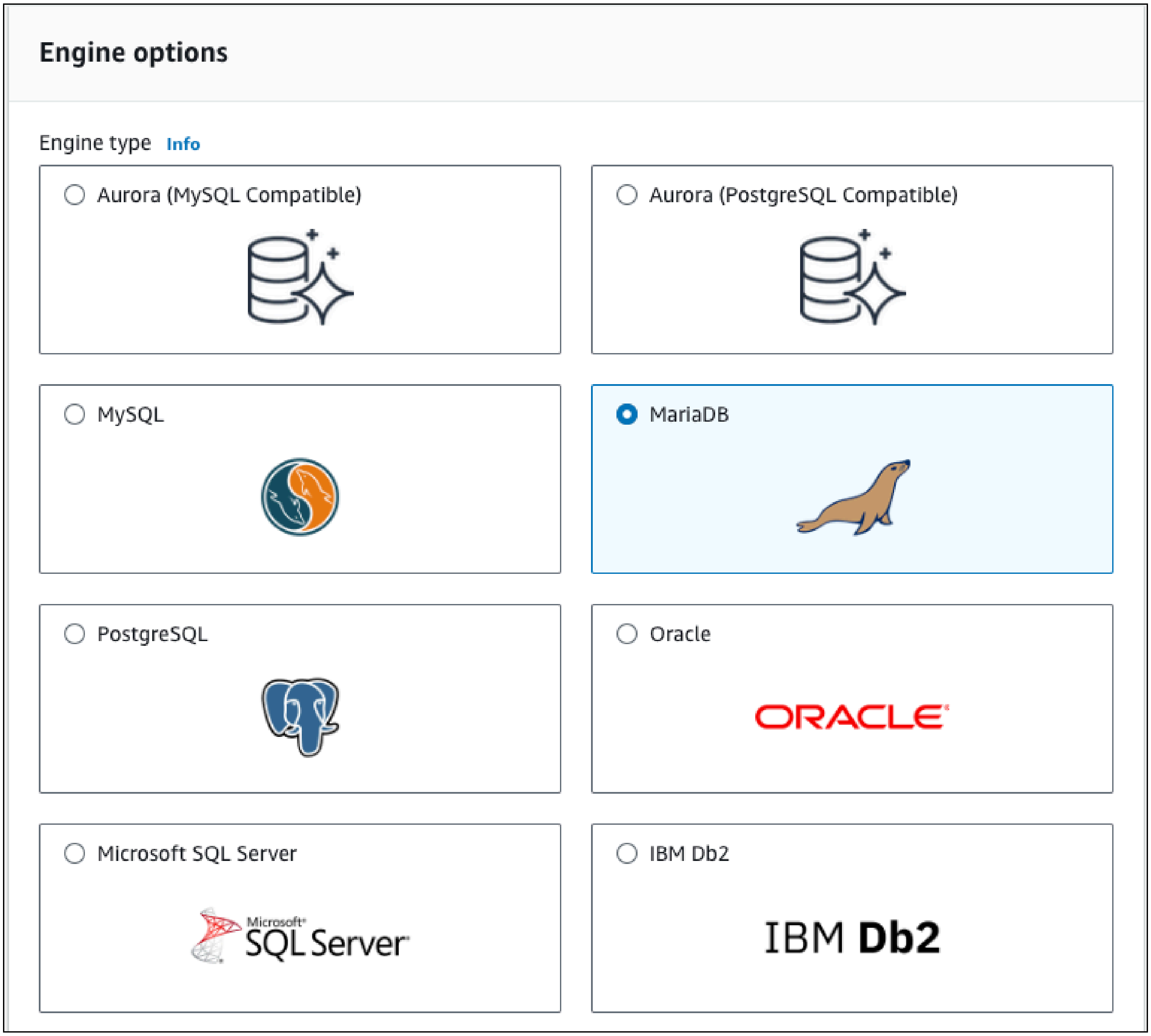
创建 Amazon RDS 数据库实例

创建维护 Web 应用程序所用数据的 RDS for MariaDB、RDS for MySQL 或 RDS for PostgreSQL 数据库实例。

RDS for MariaDB

创建 MariaDB 实例

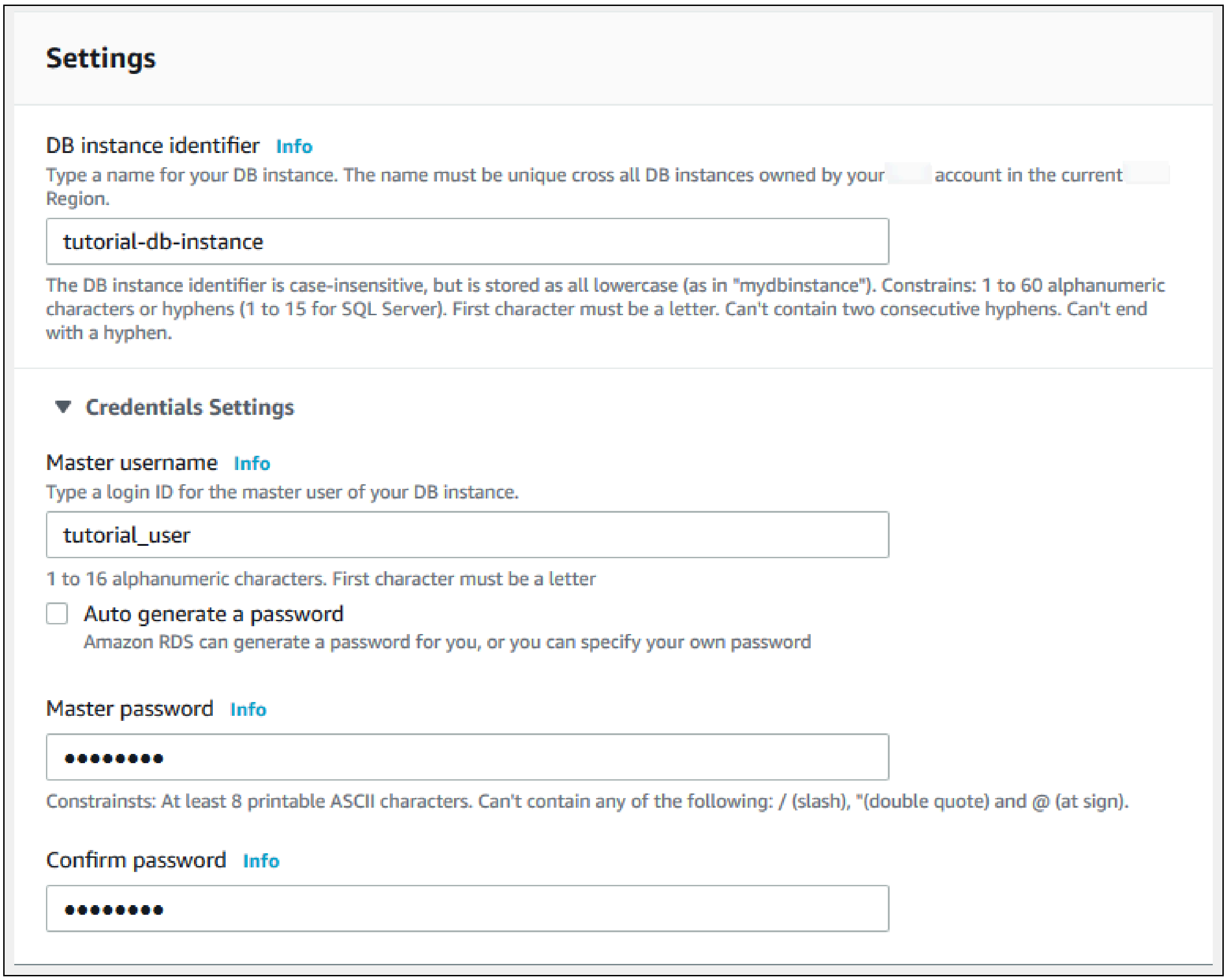
1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在 Amazon Web Services Management Console的右上角，检查 Amazon Web Services 区域。它应与您在其中创建了 EC2 实例的区域相同。
3. 在导航窗格中，选择 Databases（数据库）。
4. 选择创建数据库。
5. 在创建数据库页面上，选择标准创建。
6. 对于引擎选项，选择 MariaDB。



7. 对于模板，选择免费套餐。

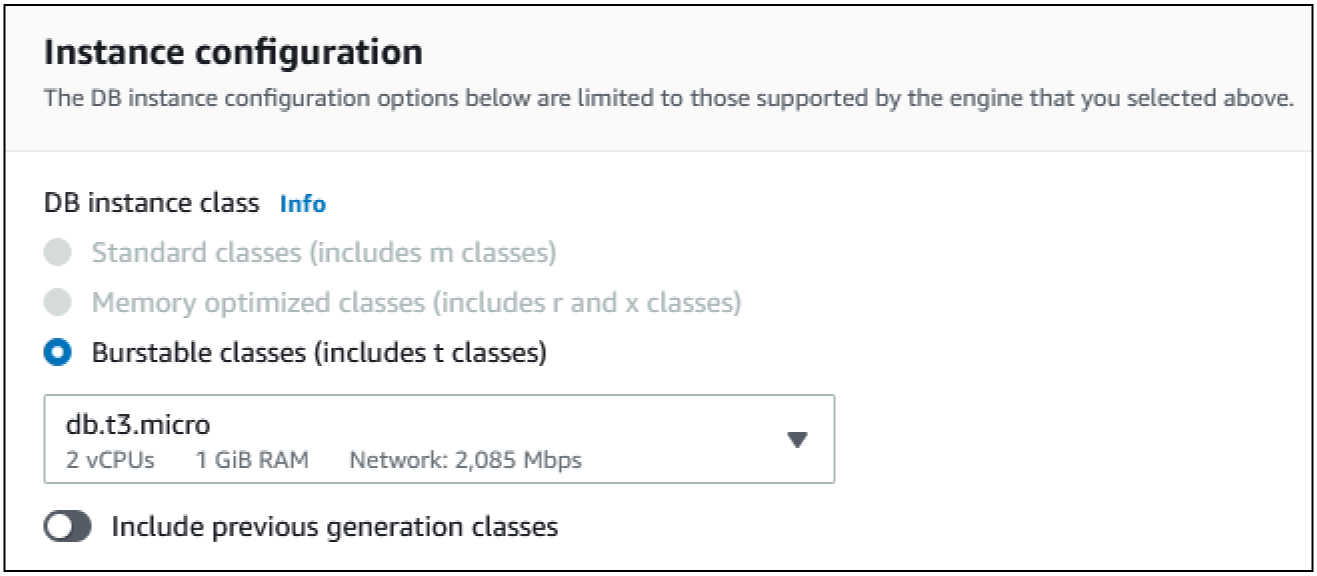
8. 在 Availability and durability（可用性和持久性）部分中，保留默认值。
9. 在设置部分中，选择这些值：

   - DB instance identifier（数据库实例标识符）– 键入 **`tutorial-db-instance`**。
   - Master username（主用户名）– 键入 **`tutorial_user`**。
   - Auto generate a password（自动生成密码）– 将该选项保留为关闭状态。
   - Master password（主密码）– 键入密码。
   - 确认密码 – 重新键入密码。



10. 在 Instance configuration（实例配置）部分中，设置以下值：

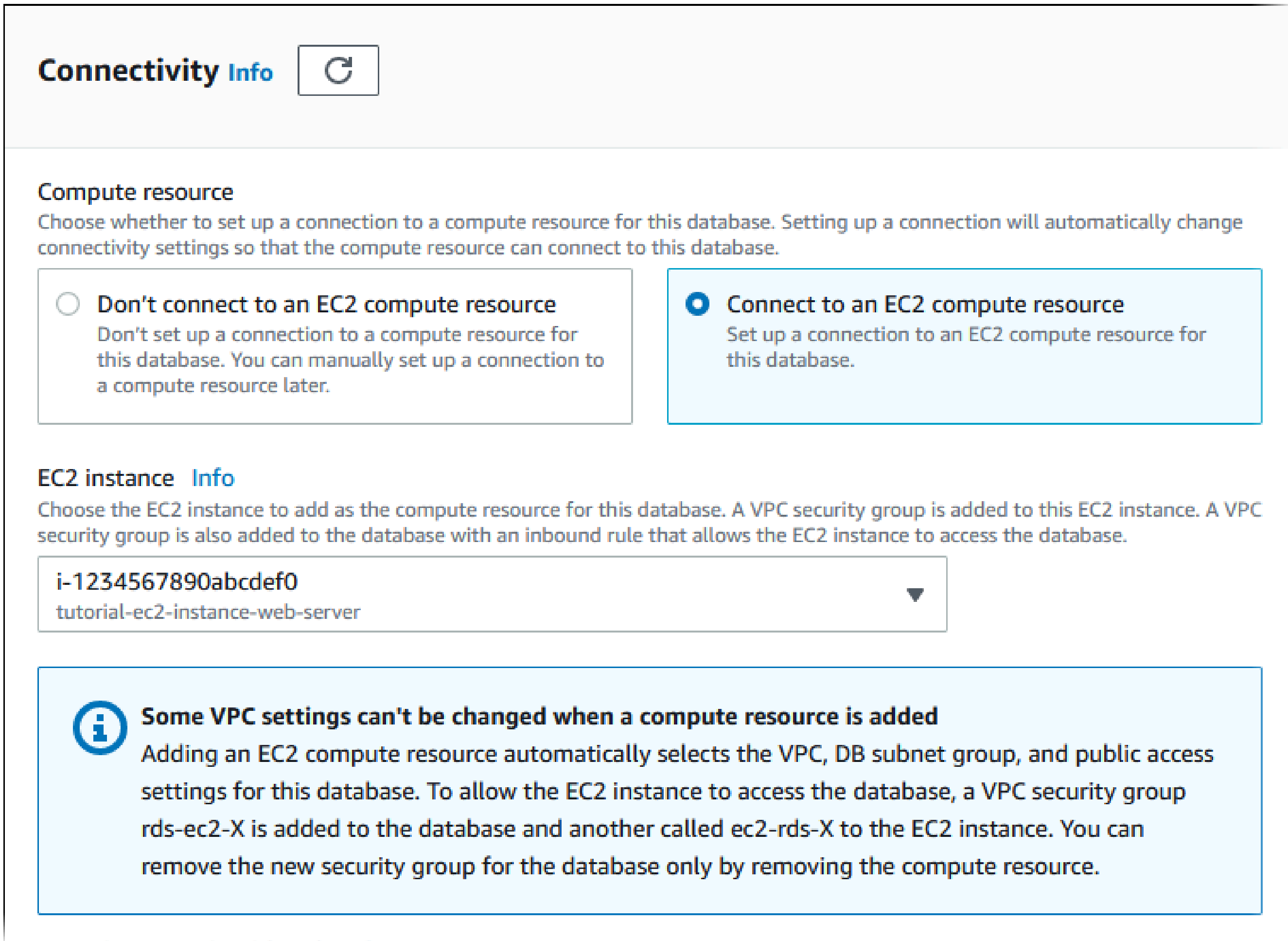
    - 可突增类（包括 t 类）
    - db.t3.micro



11. 在 Storage（存储）部分中，保留默认值。

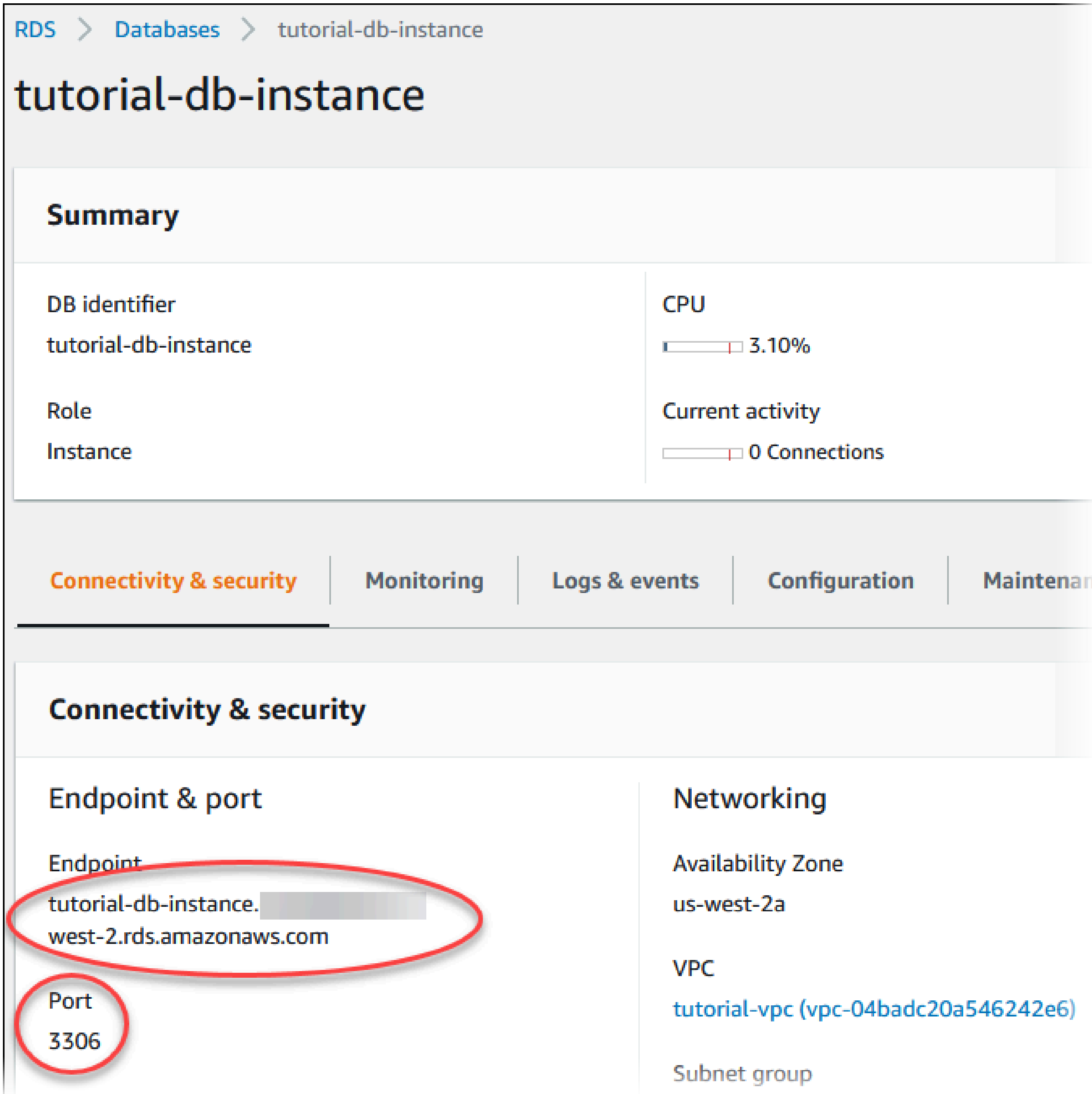
12. 在 Connectivity（连接）部分中，设置以下值并将其他值保留为其默认值：

    - 对于 Compute resource（计算资源），选择 Connect to an EC2 compute resource（连接到 EC2 计算资源）。
    - 对于 EC2 instance（EC2 实例），选择您之前创建的 EC2 实例，例如 tutorial-ec2-instance-web-server。



13. 在 Database authentication（数据库身份验证）部分中，确保选择了 Password authentication（密码身份验证）。
14. 打开附加配置部分，然后为初始数据库名称输入 **`sample`**。保留其他选项的默认设置。
15. 要创建 MariaDB 实例，请选择创建数据库。

    您的新数据库实例显示在数据库列表中，状态为正在创建。
16. 等待新数据库实例的状态显示为可用。然后选择数据库实例名称以显示其详细信息。
17. 在连接性和安全性部分中，查看数据库实例的终端节点和端口。



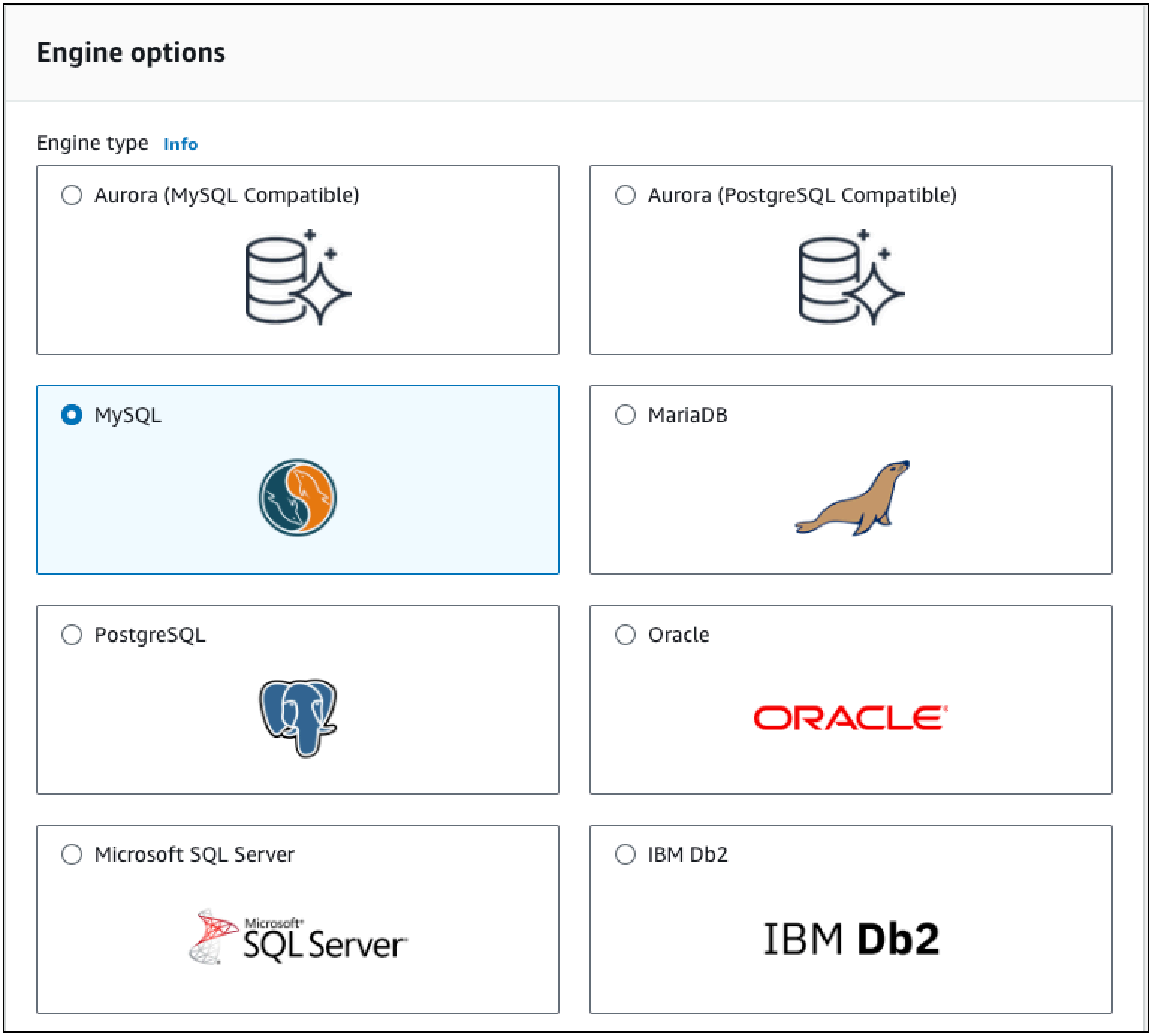
记下数据库实例的终端节点和端口。您使用这些信息将 Web 服务器连接到数据库实例。

18. 完成在 EC2 实例上安装 Web 服务器。

RDS for MySQL

创建 MySQL 数据库实例

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在 Amazon Web Services Management Console的右上角，检查 Amazon Web Services 区域。它应与您在其中创建了 EC2 实例的区域相同。
3. 在导航窗格中，选择 Databases（数据库）。
4. 选择创建数据库。
5. 在创建数据库页面上，选择标准创建。
6. 对于引擎选项，选择 MySQL。

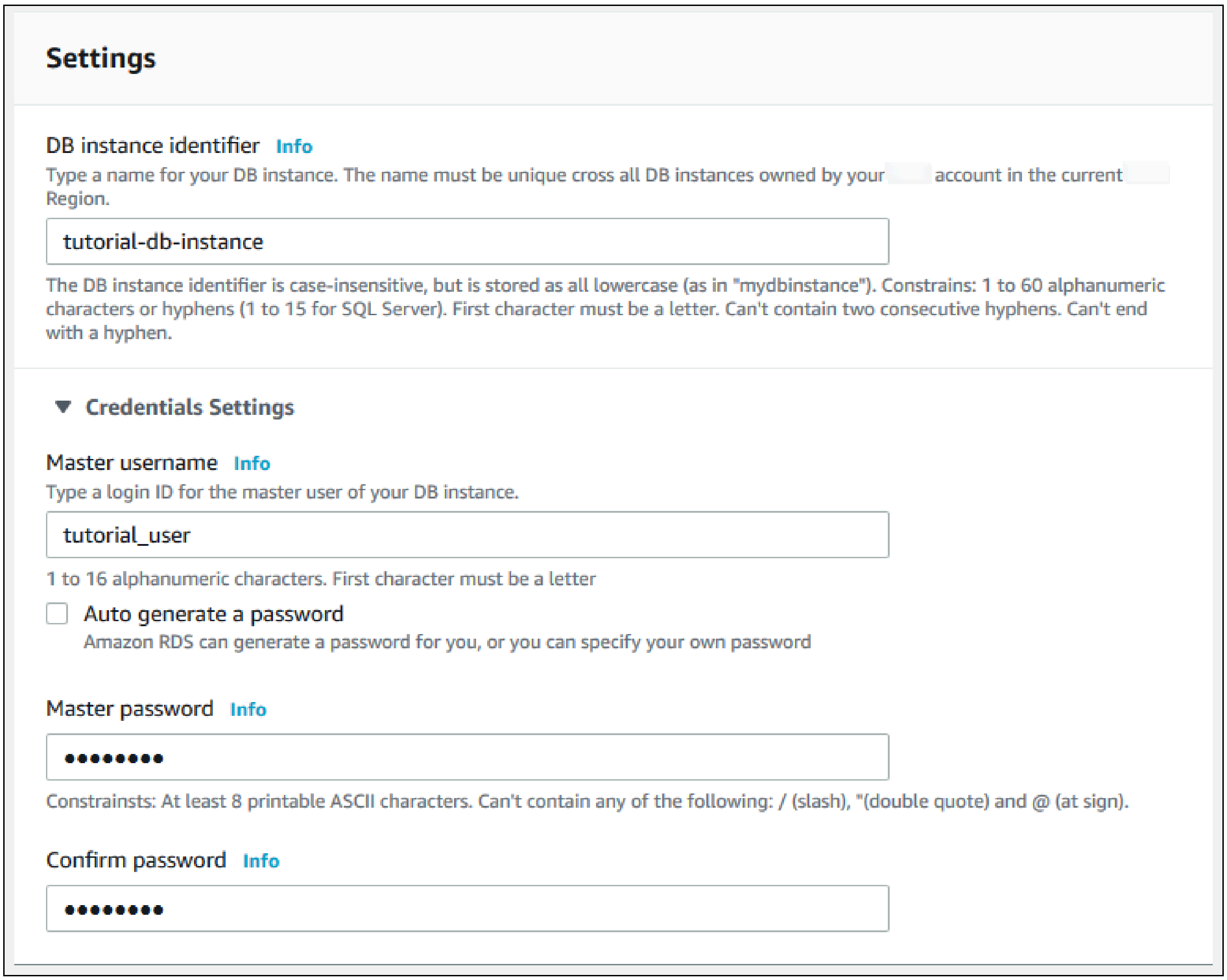


7. 对于模板，选择免费套餐。

8. 在 Availability and durability（可用性和持久性）部分中，保留默认值。

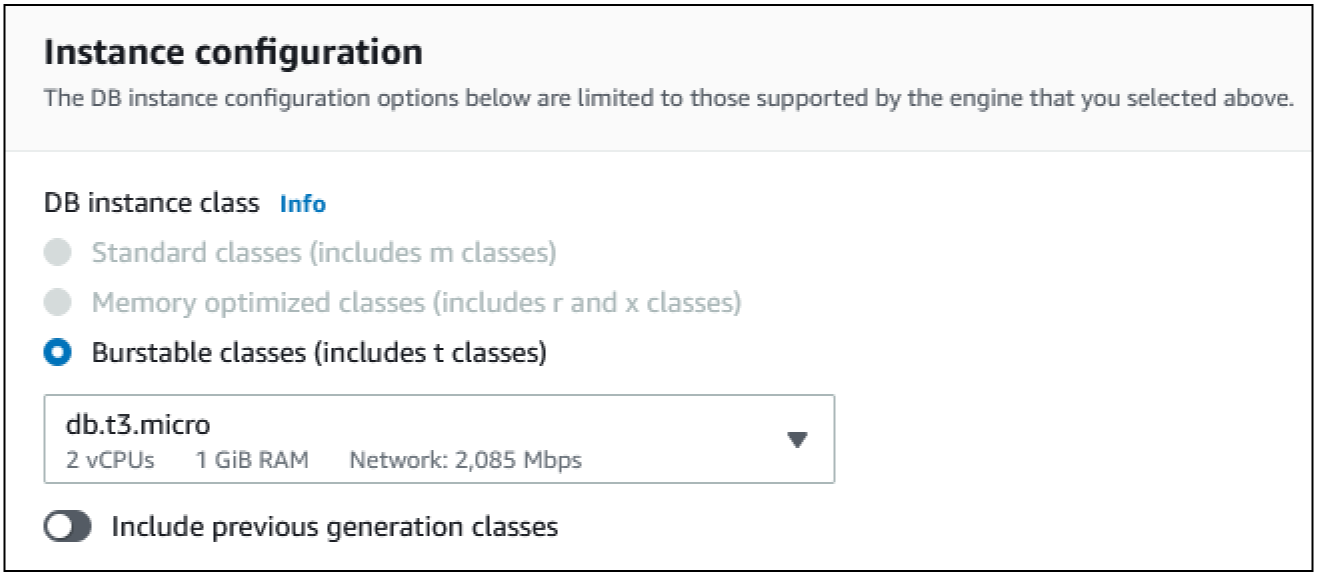
9. 在设置部分中，选择这些值：

   - DB instance identifier（数据库实例标识符）– 键入 **tutorial-db-instance**。
   - Master username（主用户名）– 键入 **tutorial_user**。
   - Auto generate a password（自动生成密码）– 将该选项保留为关闭状态。
   - Master password（主密码）– 键入密码。
   - 确认密码 – 重新键入密码。

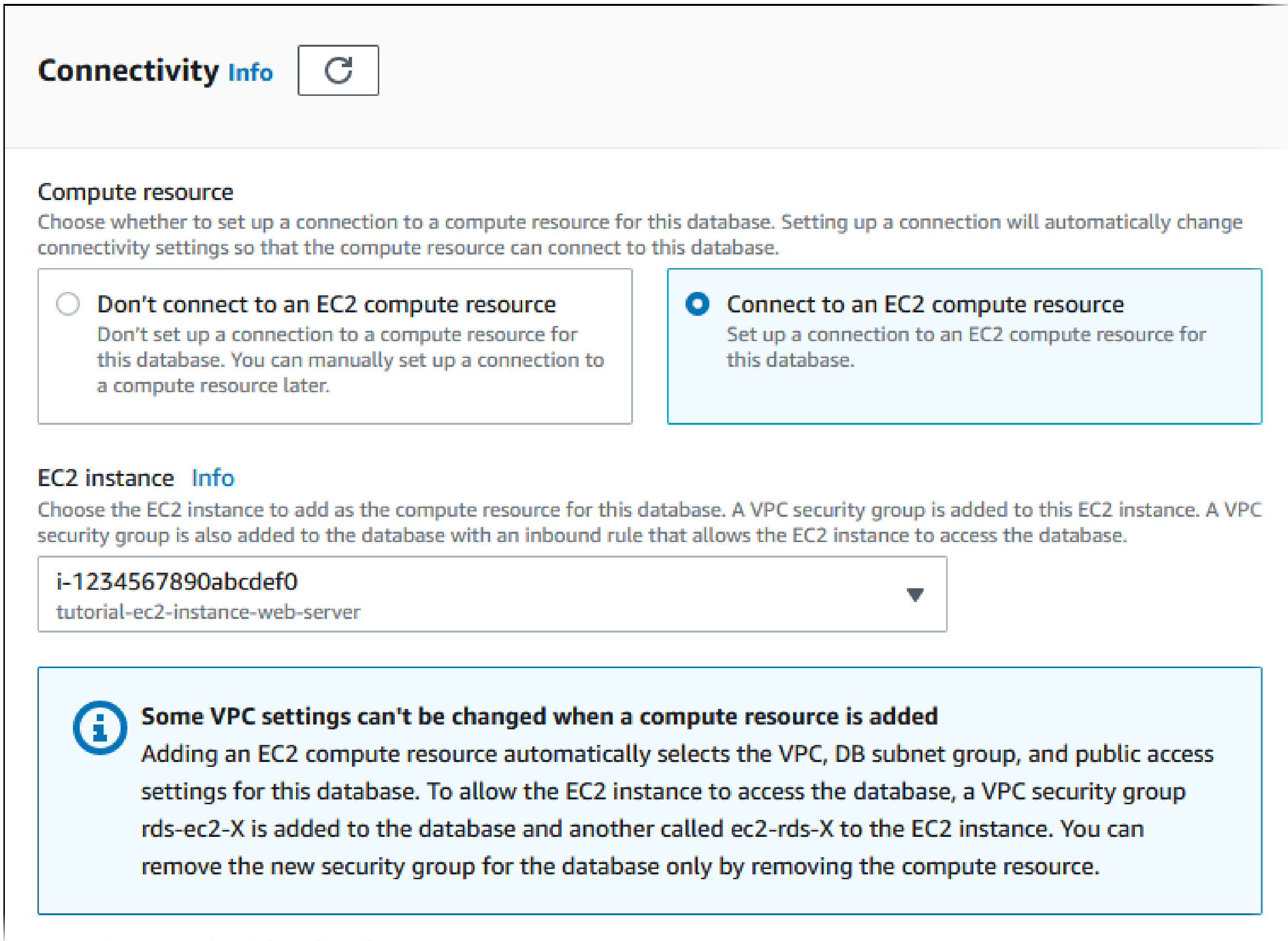


10. 在 Instance configuration（实例配置）部分中，设置以下值：

    - 可突增类（包括 t 类）
    - db.t3.micro

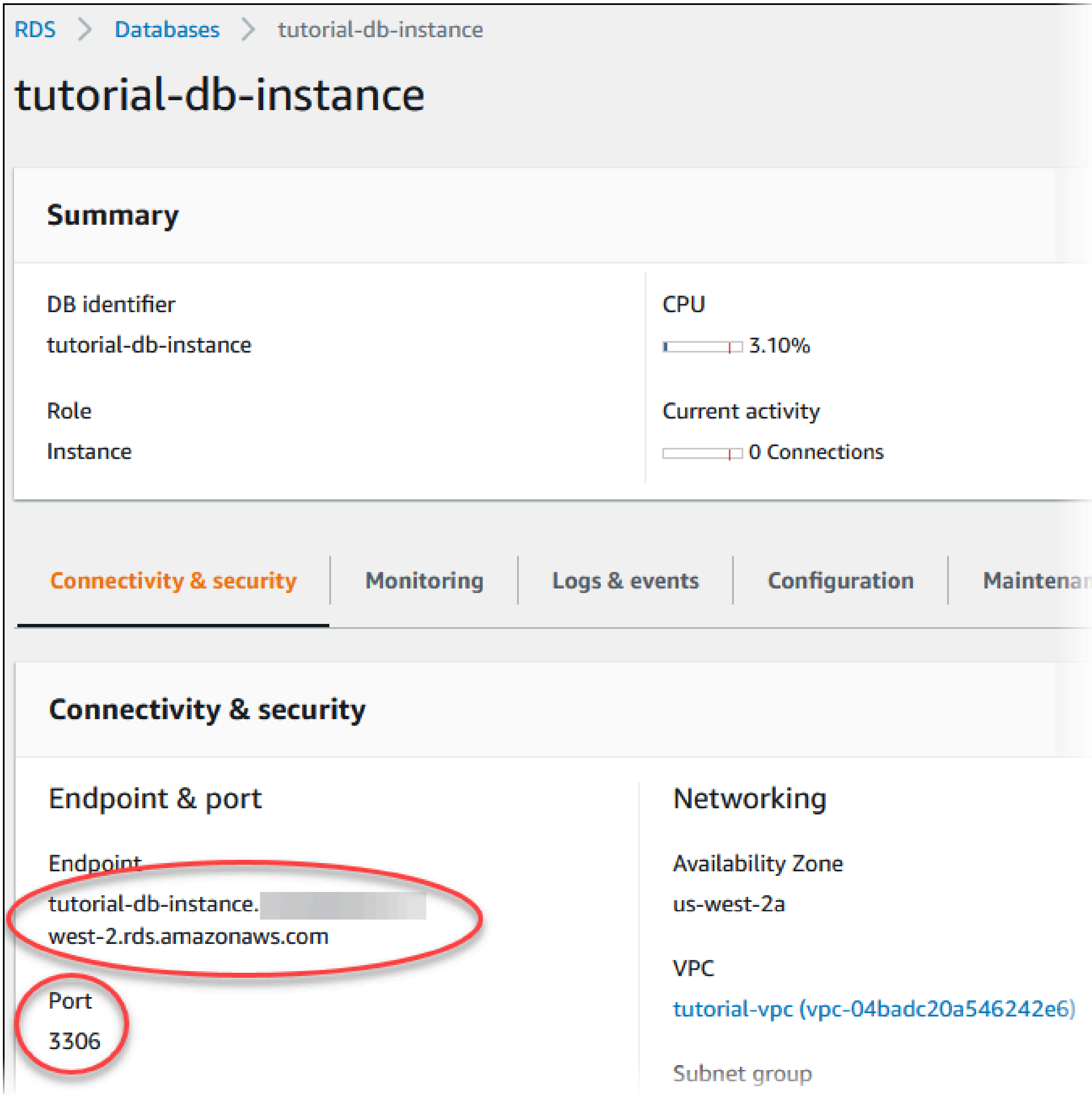


11. 在 Storage（存储）部分中，保留默认值。
12. 在 Connectivity（连接）部分中，设置以下值并将其他值保留为其默认值：
    - 对于 Compute resource（计算资源），选择 Connect to an EC2 compute resource（连接到 EC2 计算资源）。
    - 对于 EC2 instance（EC2 实例），选择您之前创建的 EC2 实例，例如 tutorial-ec2-instance-web-server。



13. 在 Database authentication（数据库身份验证）部分中，确保选择了 Password authentication（密码身份验证）。
14. 打开附加配置部分，然后为初始数据库名称输入 **`sample`**。保留其他选项的默认设置。
15. 要创建 MySQL 数据库实例，请选择 Create database（创建数据库）。

    您的新数据库实例显示在数据库列表中，状态为正在创建。
16. 等待新数据库实例的状态显示为可用。然后选择数据库实例名称以显示其详细信息。
17. 在连接性和安全性部分中，查看数据库实例的终端节点和端口。



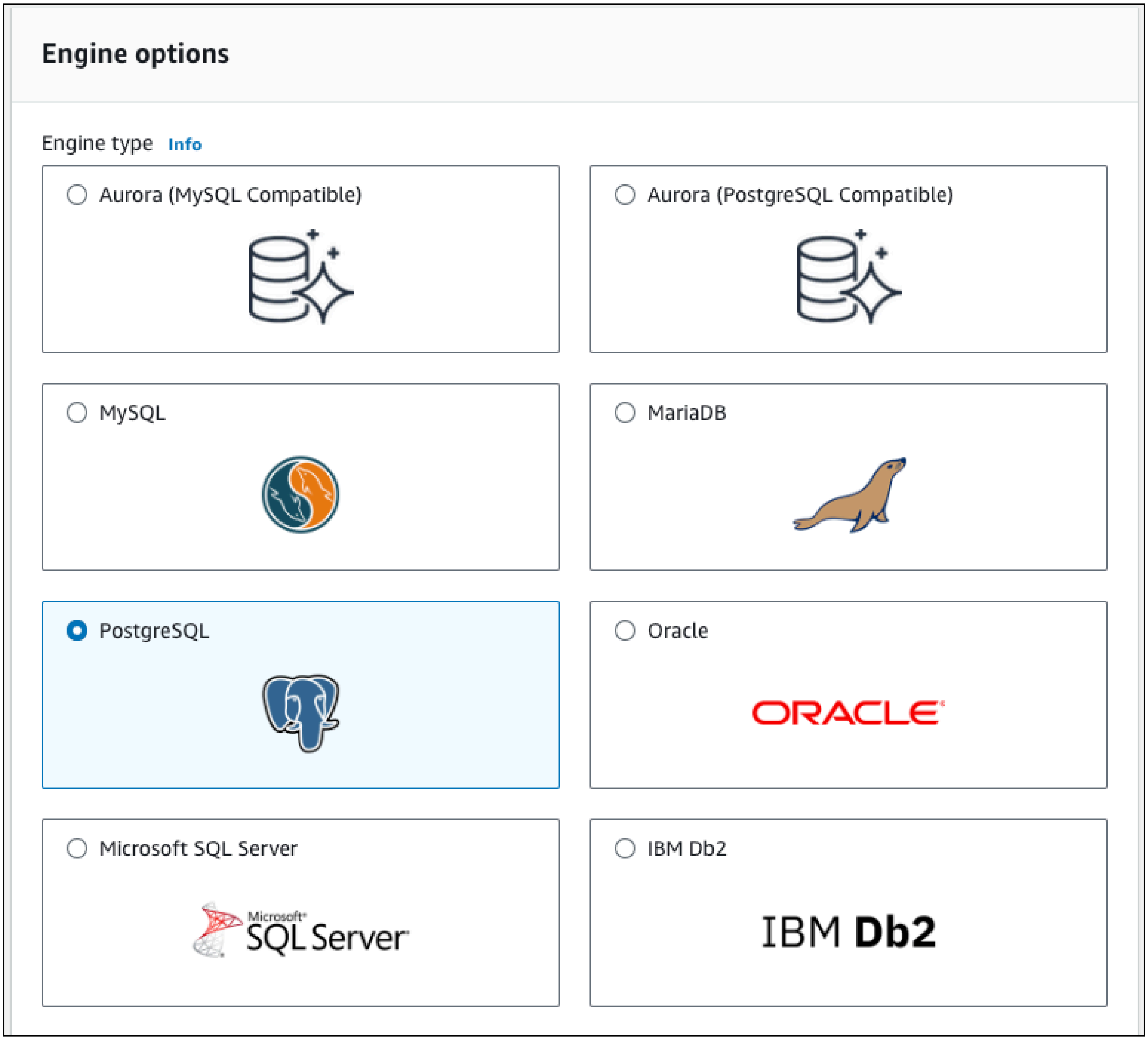
记下数据库实例的终端节点和端口。您使用这些信息将 Web 服务器连接到数据库实例。

18. 完成[在 EC2 实例上安装 Web 服务器](#)。

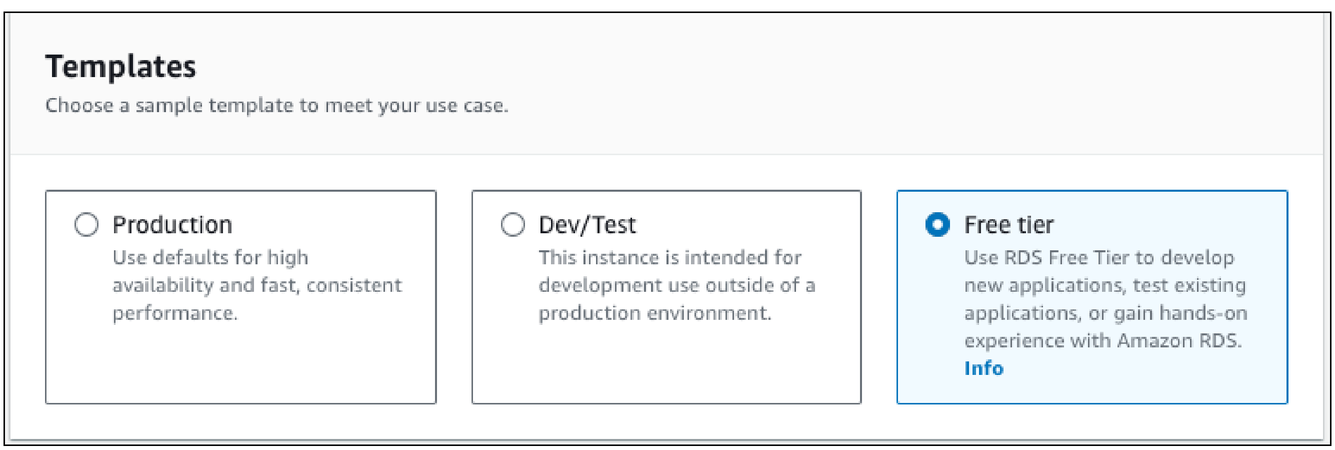
RDS for PostgreSQL

创建 PostgreSQL 数据库实例

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在 Amazon Web Services Management Console的右上角，检查 Amazon Web Services 区域。它应与您在其中创建了 EC2 实例的区域相同。
3. 在导航窗格中，选择 Databases（数据库）。
4. 选择创建数据库。
5. 在创建数据库页面上，选择标准创建。
6. 对于引擎选项，选择 PostgreSQL。

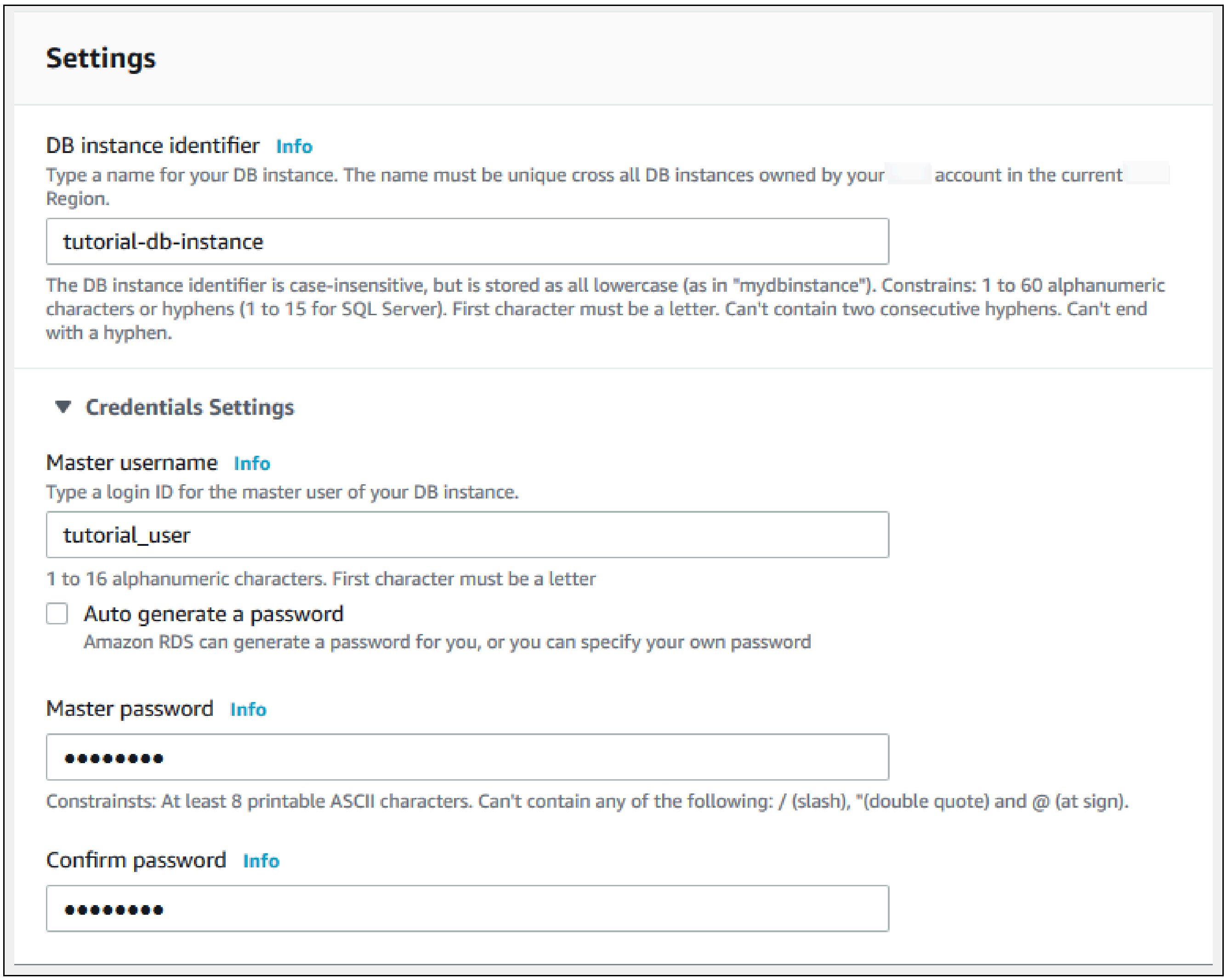


7. 对于模板，选择免费套餐。



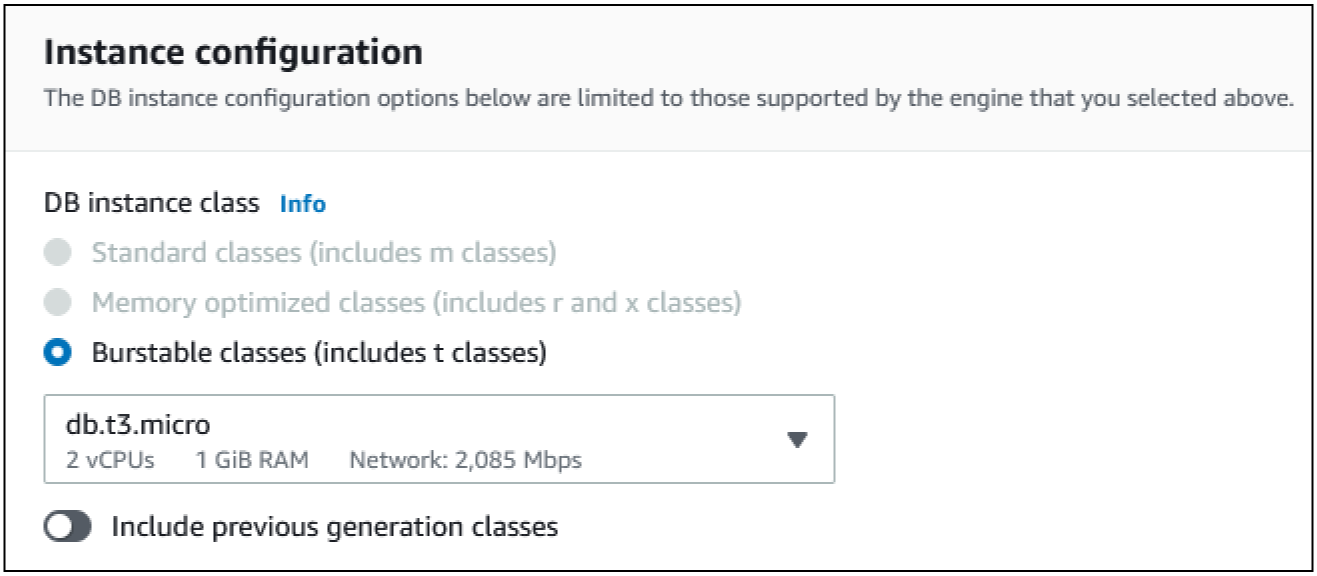
8. 在 Availability and durability（可用性和持久性）部分中，保留默认值。
9. 在设置部分中，选择这些值：

   - DB instance identifier（数据库实例标识符）– 键入 **`tutorial-db-instance`**。
   - Master username（主用户名）– 键入 **`tutorial_user`**。
   - Auto generate a password（自动生成密码）– 将该选项保留为关闭状态。
   - Master password（主密码）– 键入密码。
   - 确认密码 – 重新键入密码。

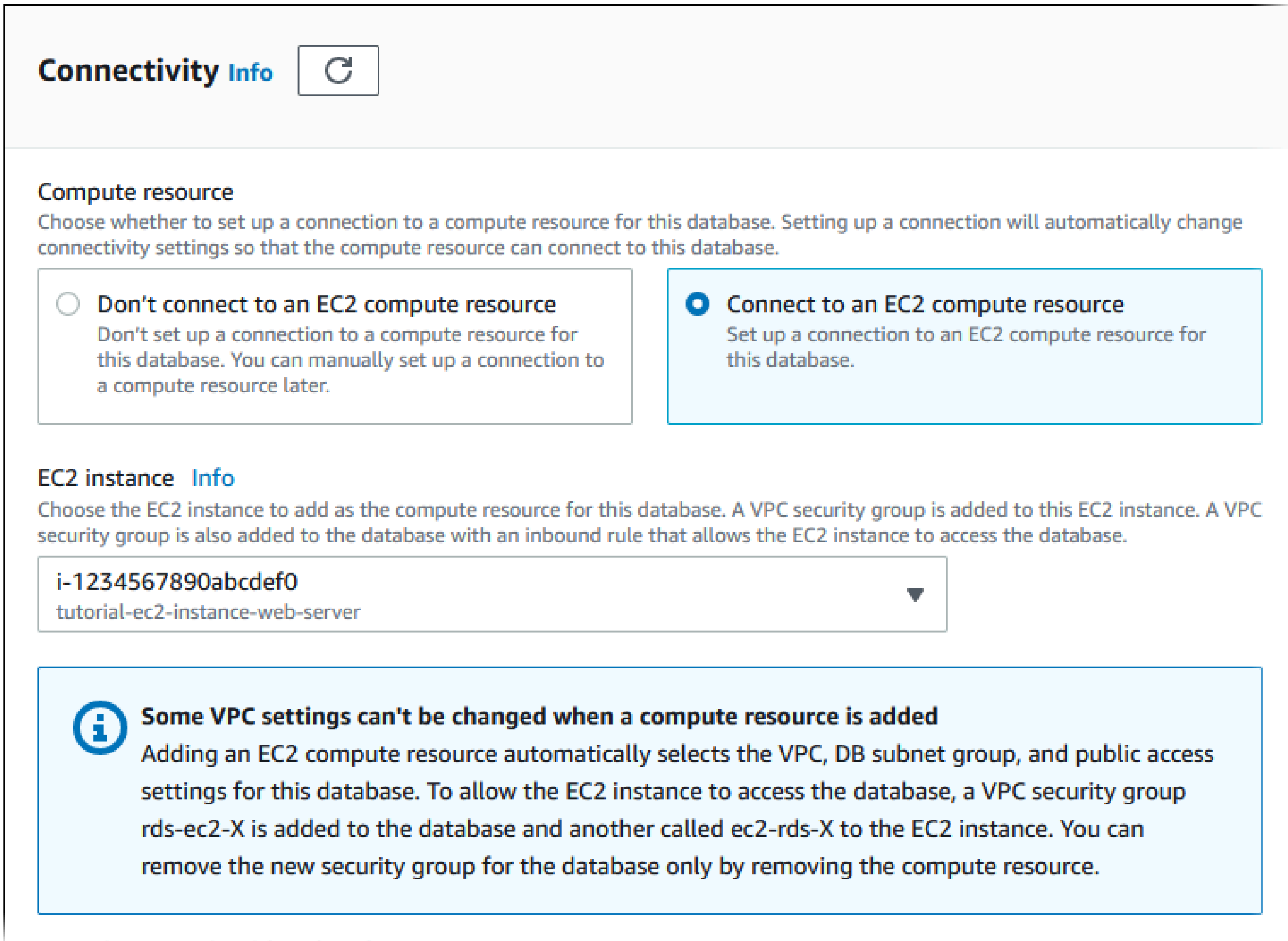


10. 在 Instance configuration（实例配置）部分中，设置以下值：

   - 可突增类（包括 t 类）
   - db.t3.micro

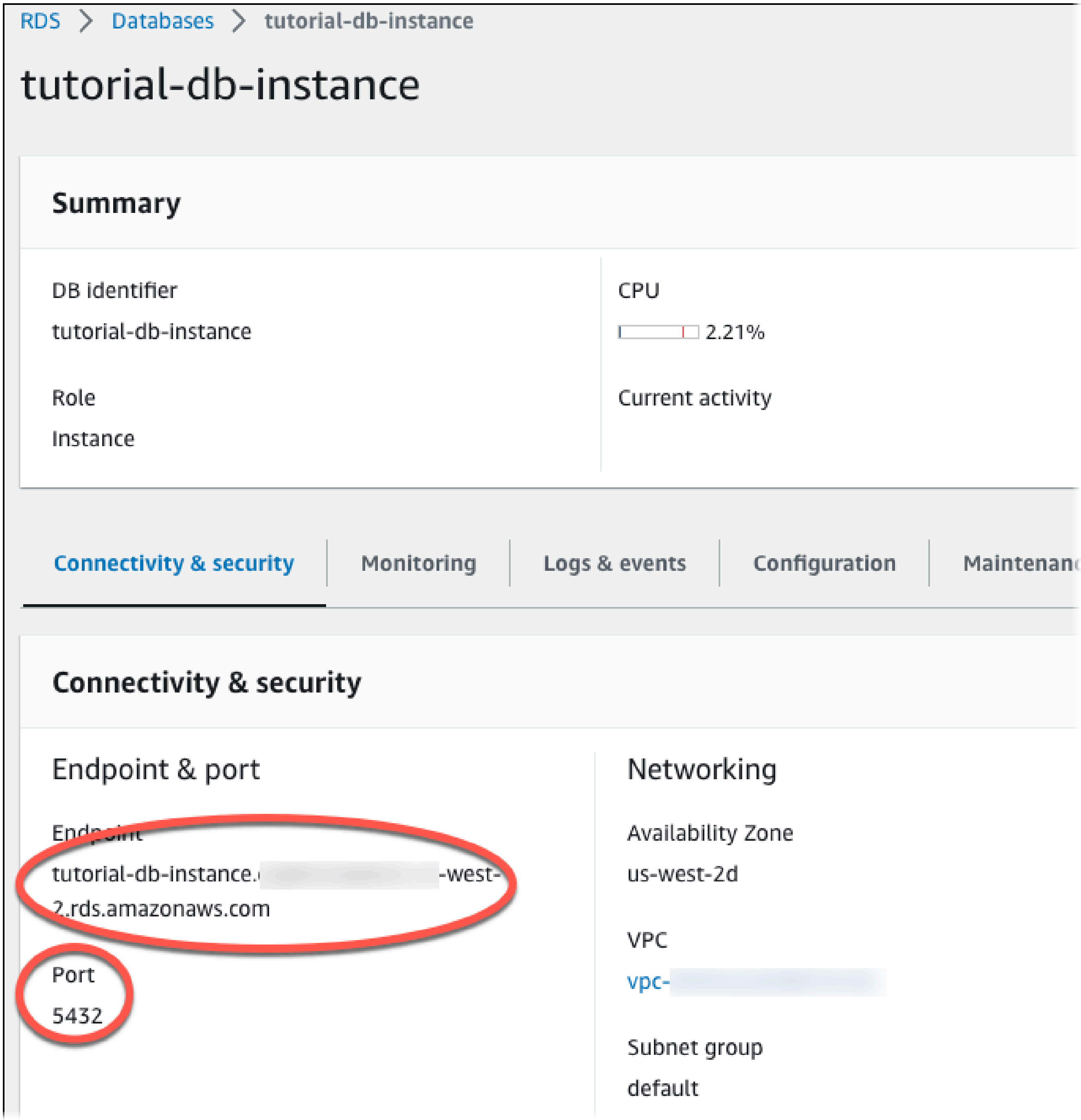


11. 在 Storage（存储）部分中，保留默认值。
12. 在 Connectivity（连接）部分中，设置以下值并将其他值保留为其默认值：
    - 对于 Compute resource（计算资源），选择 Connect to an EC2 compute resource（连接到 EC2 计算资源）。
    - 对于 EC2 instance（EC2 实例），选择您之前创建的 EC2 实例，例如 tutorial-ec2-instance-web-server。



13. 在 Database authentication（数据库身份验证）部分中，确保选择了 Password authentication（密码身份验证）。
14. 打开附加配置部分，然后为初始数据库名称输入 **sample**。保留其他选项的默认设置。
15. 要创建 PostgreSQL 数据库实例，请选择创建数据库。

    您的新数据库实例显示在数据库列表中，状态为正在创建。
16. 等待新数据库实例的状态显示为可用。然后选择数据库实例名称以显示其详细信息。
17. 在连接性和安全性部分中，查看数据库实例的终端节点和端口。



记下数据库实例的终端节点和端口。您使用这些信息将 Web 服务器连接到数据库实例。

18. 完成在 EC2 实例上安装 Web 服务器。

在 EC2 实例上安装 Web 服务器

在您在启动 EC2 实例以连接数据库实例中创建的 EC2 实例上安装 Web 服务器。Web 服务器连接到您在创建 Amazon RDS 数据库实例中创建的 Amazon RDS 数据库实例。

使用 PHP 和 MariaDB 安装 Apache Web 服务器

连接到 EC2 实例并安装 Web 服务器。

连接到 EC2 实例并安装带有 PHP 的 Apache Web 服务器

1. 按照《Amazon EC2 用户指南》中的连接到 Linux 实例中的步骤操作，连接到您之前创建的 EC2 实例。

   我们建议您使用 SSH 连接到 EC2 实例。如果 SSH 客户端实用程序安装在 Windows、Linux 或 Mac 上，则可以使用以下命令格式连接到该实例：

   ```
   ssh -i location_of_pem_file ec2-user@ec2-instance-public-dns-name
   ```

   例如，假设在 Linux 上 `ec2-database-connect-key-pair.pem` 存储在 `/dir1` 中，而 EC2 实例的公有 IPv4 DNS 为 `ec2-12-345-678-90.compute-1.amazonaws.com`。SSH 命令将如下所示：

   ```
   ssh -i /dir1/ec2-database-connect-key-pair.pem ec2-
   user@ec2-12-345-678-90.compute-1.amazonaws.com
   ```

2. 通过更新 EC2 实例上的软件，获取最新的错误修复和安全更新。要执行此操作，请使用以下命令。

   > **Note**
   >
   > `-y` 选项安装更新时不提示确认。要在安装前检查更新，请忽略该选项。

   ```
   sudo dnf update -y
   ```

3. 更新完成后，使用以下命令安装 Apache Web 服务器、PHP 和 MariaDB 或 PostgreSQL 软件。此命令同时安装多个软件包和相关依赖项。

MariaDB & MySQL

```
sudo dnf install -y httpd php php-mysqli mariadb105
```

PostgreSQL

```
sudo dnf install -y httpd php php-pgsql postgresql15
```

如果您收到错误，则您的实例可能不是使用 Amazon Linux 2023 AMI 启动的。您可能使用的是 Amazon Linux 2 AMI。您可以使用以下命令查看 Amazon Linux 的版本。

```
cat /etc/system-release
```

有关更多信息，请参阅更新实例软件。

4. 使用下面所示的命令启动 Web 服务器。

```
sudo systemctl start httpd
```

您可以测试 Web 服务器是否已正确安装和启动。为此，请在 Web 浏览器的地址栏中输入 EC2 实例的公有域名系统 (DNS) 名称，例如：`http://ec2-42-8-168-21.us-west-1.compute.amazonaws.com`。如果 Web 服务器正在运行，您将看到 Apache 测试页面。

如果您没有看到 Apache 测试页面，请检查您在教程：创建 VPC 以用于数据库实例（仅限 IPv4）中创建的 VPC 安全组的入站规则。确保入站规则包括一条允许 HTTP（端口 80）访问 IP 地址以连接到 Web 服务器的规则。

> **Note**
>
> Apache 测试页面仅在文档根目录 `/var/www/html` 中无内容时才显示。将内容添加到文档根目录后，您的内容将显示在 EC2 实例的公有 DNS 地址处。在此之前，它出现在 Apache 测试页面上。

5. 使用 `systemctl` 命令配置 Web 服务器以使其在每次系统启动时启动。

```
sudo systemctl enable httpd
```

要允许 ec2-user 在 Apache Web 服务器的默认根目录中管理文件，请修改 /var/www 目录的所有权和权限。有多种方式可以完成此任务。在本教程中，可将 ec2-user 添加到 apache 组，将 apache 目录的所有权授予 /var/www 组，并为该组指定写入权限。

设置 Apache Web 服务器的文件权限

1. 将 ec2-user 用户添加到 apache 组。

```
sudo usermod -a -G apache ec2-user
```

2. 注销以刷新您的权限并包含新的 apache 组。

```
exit
```

3. 再重新登录并使用 apache 命令验证 groups 组是否存在。

```
groups
```

   输出看上去类似于以下内容：

```
ec2-user adm wheel apache systemd-journal
```

4. 将 /var/www 目录的组所有权及其内容更改到 apache 组。

```
sudo chown -R ec2-user:apache /var/www
```

5. 更改 /var/www 及其子目录的目录权限，以添加组写入权限并设置未来创建的子目录上的组 ID。

```
sudo chmod 2775 /var/www
find /var/www -type d -exec sudo chmod 2775 {} \;
```

6. 递归地更改 /var/www 目录及其子目录中的文件的权限，以添加组写入权限。

```
find /var/www -type f -exec sudo chmod 0664 {} \;
```

现在，ec2-user（和 apache 组的任何将来成员）可以添加、删除和编辑 Apache 文档根目录中的文件。这使您可以添加内容，例如静态网站或 PHP 应用程序。

> **Note**
>
> 运行 HTTP 协议的 Web 服务器不为其发送或接收的数据提供传输安全。当您使用 Web 浏览器连接 HTTP 服务器时，窃取者可在沿网络路径的任何位置看到许多信息。这些信息包括您访问的 URL、您接收的网页内容以及任何 HTML 表单的内容（包括密码）。
>
> 保护您的 Web 服务器的最佳实践是安装对于 HTTPS（HTTP Secure）的支持。此协议利用 SSL/TLS 加密保护您的数据。有关更多信息，请参阅在Amazon EC2用户指南中的[教程：使用 Amazon Linux AMI 配置 SSL/TLS]。

将您的 Apache Web 服务器连接到数据库实例

接着，将内容添加到连接到 Amazon RDS 数据库实例的 Apache Web 服务器。

将内容添加到连接到数据库实例的 Apache Web 服务器

1. 在仍连接到 EC2 实例时，将目录更改到 `/var/www` 并创建名为 `inc` 的新子目录。

   ```
   cd /var/www
   mkdir inc
   cd inc
   ```

2. 在名为 `inc` 的 `dbinfo.inc` 目录中新建文件，然后通过调用 nano（或您选择的编辑器）编辑文件。

   ```
   >dbinfo.inc
   nano dbinfo.inc
   ```

3. 将以下内容添加到 `dbinfo.inc` 文件。在这里，*`db_instance_endpoint`* 是数据库集群的不带端口的数据库集群写入器端点。

   > **Note**
   >
   > 我们建议将用户名和密码信息放在不属于 Web 服务器的文档根目录的文件夹中。这样做会减少您的安全信息被泄露的可能性。
   >
   > 确保在应用程序中将 `master password` 更改为合适的密码。

   ```
   <?php
   ```

```
define('DB_SERVER', 'db_instance_endpoint');
define('DB_USERNAME', 'tutorial_user');
define('DB_PASSWORD', 'master password');
define('DB_DATABASE', 'sample');
?>
```

4. 保存并关闭 dbinfo.inc文件。如果您使用的是 nano，请使用 Ctrl+S 和 Ctrl+X 保存并关闭文件。
5. 将目录更改为 /var/www/html。

```
cd /var/www/html
```

6. 在名为 html 的 SamplePage.php 目录中新建文件，然后通过调用 nano（或您选择的编辑器）编辑文件。

```
>SamplePage.php
nano SamplePage.php
```

7. 将以下内容添加到 SamplePage.php 文件：

MariaDB & MySQL

```
<?php include "../inc/dbinfo.inc"; ?>
<html>
<body>
<h1>Sample page</h1>
<?php

  /* Connect to MySQL and select the database. */
  $connection = mysqli_connect(DB_SERVER, DB_USERNAME, DB_PASSWORD);

  if (mysqli_connect_errno()) echo "Failed to connect to MySQL: " .
 mysqli_connect_error();

  $database = mysqli_select_db($connection, DB_DATABASE);

  /* Ensure that the EMPLOYEES table exists. */
  VerifyEmployeesTable($connection, DB_DATABASE);

  /* If input fields are populated, add a row to the EMPLOYEES table. */
  $employee_name = htmlentities($_POST['NAME']);
```

```
  $employee_address = htmlentities($_POST['ADDRESS']);

  if (strlen($employee_name) || strlen($employee_address)) {
    AddEmployee($connection, $employee_name, $employee_address);
  }
?>

<!-- Input form -->
<form action="<?PHP echo $_SERVER['SCRIPT_NAME'] ?>" method="POST">
  <table border="0">
    <tr>
      <td>NAME</td>
      <td>ADDRESS</td>
    </tr>
    <tr>
      <td>
        <input type="text" name="NAME" maxlength="45" size="30" />
      </td>
      <td>
        <input type="text" name="ADDRESS" maxlength="90" size="60" />
      </td>
      <td>
        <input type="submit" value="Add Data" />
      </td>
    </tr>
  </table>
</form>

<!-- Display table data. -->
<table border="1" cellpadding="2" cellspacing="2">
  <tr>
    <td>ID</td>
    <td>NAME</td>
    <td>ADDRESS</td>
  </tr>

<?php

$result = mysqli_query($connection, "SELECT * FROM EMPLOYEES");

while($query_data = mysqli_fetch_row($result)) {
  echo "<tr>";
  echo "<td>",$query_data[0], "</td>",
       "<td>",$query_data[1], "</td>",
```

```
        "<td>",$query_data[2], "</td>";
  echo "</tr>";
}
?>

</table>

<!-- Clean up. -->
<?php

  mysqli_free_result($result);
  mysqli_close($connection);

?>

</body>
</html>



<?php

/* Add an employee to the table. */
function AddEmployee($connection, $name, $address) {
   $n = mysqli_real_escape_string($connection, $name);
   $a = mysqli_real_escape_string($connection, $address);

   $query = "INSERT INTO EMPLOYEES (NAME, ADDRESS) VALUES ('$n', '$a');";

   if(!mysqli_query($connection, $query)) echo("<p>Error adding employee data.</
p>");
}

/* Check whether the table exists and, if not, create it. */
function VerifyEmployeesTable($connection, $dbName) {
  if(!TableExists("EMPLOYEES", $connection, $dbName))
  {
     $query = "CREATE TABLE EMPLOYEES (
         ID int(11) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
         NAME VARCHAR(45),
         ADDRESS VARCHAR(90)
       )";

     if(!mysqli_query($connection, $query)) echo("<p>Error creating table.</
p>");
```

```
  }
}

/* Check for the existence of a table. */
function TableExists($tableName, $connection, $dbName) {
  $t = mysqli_real_escape_string($connection, $tableName);
  $d = mysqli_real_escape_string($connection, $dbName);

  $checktable = mysqli_query($connection,
      "SELECT TABLE_NAME FROM information_schema.TABLES WHERE TABLE_NAME = '$t'
 AND TABLE_SCHEMA = '$d'");

  if(mysqli_num_rows($checktable) > 0) return true;

  return false;
}
?>
```

PostgreSQL

```
<?php include "../inc/dbinfo.inc"; ?>

<html>
<body>
<h1>Sample page</h1>
<?php

/* Connect to PostgreSQL and select the database. */
$constring = "host=" . DB_SERVER . " dbname=" . DB_DATABASE . " user=" .
 DB_USERNAME . " password=" . DB_PASSWORD ;
$connection = pg_connect($constring);

if (!$connection){
 echo "Failed to connect to PostgreSQL";
 exit;
}

/* Ensure that the EMPLOYEES table exists. */
VerifyEmployeesTable($connection, DB_DATABASE);

/* If input fields are populated, add a row to the EMPLOYEES table. */
$employee_name = htmlentities($_POST['NAME']);
```

```
$employee_address = htmlentities($_POST['ADDRESS']);

if (strlen($employee_name) || strlen($employee_address)) {
  AddEmployee($connection, $employee_name, $employee_address);
}

?>

<!-- Input form -->
<form action="<?PHP echo $_SERVER['SCRIPT_NAME'] ?>" method="POST">
  <table border="0">
    <tr>
      <td>NAME</td>
      <td>ADDRESS</td>
    </tr>
    <tr>
      <td>
    <input type="text" name="NAME" maxlength="45" size="30" />
      </td>
      <td>
    <input type="text" name="ADDRESS" maxlength="90" size="60" />
      </td>
      <td>
    <input type="submit" value="Add Data" />
      </td>
    </tr>
  </table>
</form>
<!-- Display table data. -->
<table border="1" cellpadding="2" cellspacing="2">
  <tr>
    <td>ID</td>
    <td>NAME</td>
    <td>ADDRESS</td>
  </tr>

<?php

$result = pg_query($connection, "SELECT * FROM EMPLOYEES");

while($query_data = pg_fetch_row($result)) {
  echo "<tr>";
  echo "<td>",$query_data[0], "</td>",
       "<td>",$query_data[1], "</td>",
```

```
        "<td>",$query_data[2], "</td>";
  echo "</tr>";
}
?>
</table>

<!-- Clean up. -->
<?php

  pg_free_result($result);
  pg_close($connection);
?>
</body>
</html>


<?php

/* Add an employee to the table. */
function AddEmployee($connection, $name, $address) {
   $n = pg_escape_string($name);
   $a = pg_escape_string($address);
   echo "Forming Query";
   $query = "INSERT INTO EMPLOYEES (NAME, ADDRESS) VALUES ('$n', '$a');";

   if(!pg_query($connection, $query)) echo("<p>Error adding employee data.</
p>");
}

/* Check whether the table exists and, if not, create it. */
function VerifyEmployeesTable($connection, $dbName) {
  if(!TableExists("EMPLOYEES", $connection, $dbName))
  {
     $query = "CREATE TABLE EMPLOYEES (
         ID serial PRIMARY KEY,
         NAME VARCHAR(45),
         ADDRESS VARCHAR(90)
       )";

     if(!pg_query($connection, $query)) echo("<p>Error creating table.</p>");
  }
}
/* Check for the existence of a table. */
function TableExists($tableName, $connection, $dbName) {
```

```
  $t = strtolower(pg_escape_string($tableName)); //table name is case sensitive
  $d = pg_escape_string($dbName); //schema is 'public' instead of 'sample' db
 name so not using that

  $query = "SELECT TABLE_NAME FROM information_schema.TABLES WHERE TABLE_NAME =
 '$t';";
  $checktable = pg_query($connection, $query);

  if (pg_num_rows($checktable) >0) return true;
  return false;

}
?>
```

8. 保存并关闭 `SamplePage.php` 文件。
9. 打开 Web 浏览器并浏览到 `http://EC2 instance endpoint/SamplePage.php`（例如：`http://ec2-12-345-67-890.us-west-2.compute.amazonaws.com/SamplePage.php`）来验证 Web 服务器是否已成功连接到数据库实例。

您可以使用 `SamplePage.php` 将数据添加到数据库实例。您添加的数据之后将显示在该页面上。要验证数据是否已插入到表中，请在 Amazon EC2 实例上安装 MySQL 客户端。然后，连接到数据库实例并查询表。

有关安装 MySQL 客户端并连接到数据库实例的信息，请参阅 连接到 MySQL 数据库实例。

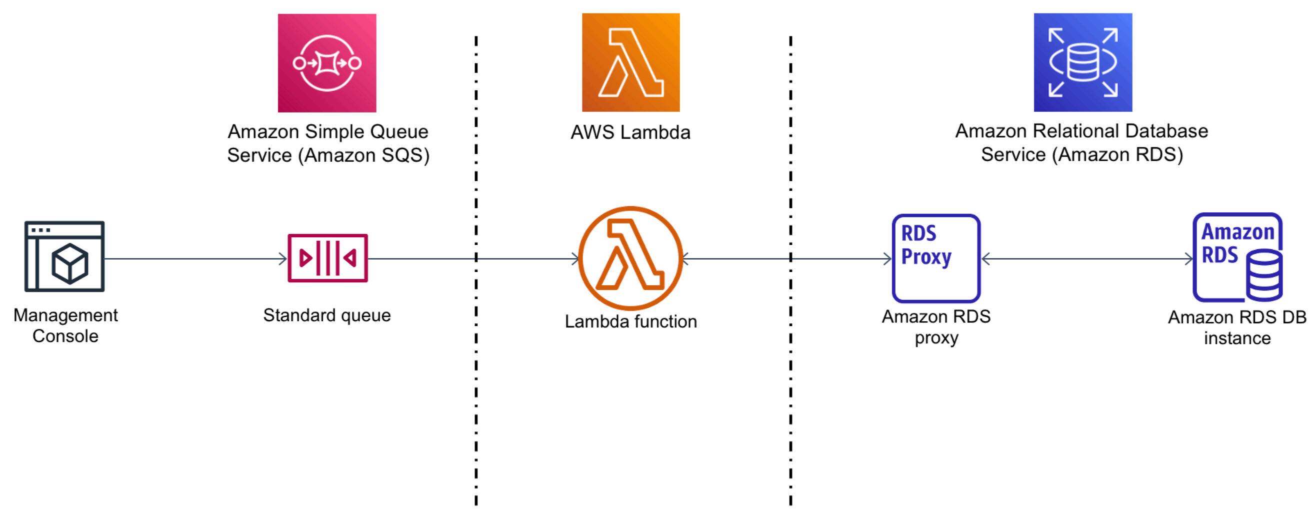
要确保您的数据库实例尽可能安全，请验证 VPC 外部的源是否无法连接到您的数据库实例。

在您完成 Web 服务器和数据库测试后，应删除您的数据库实例 和 Amazon EC2 实例。

- 要删除数据库实例，请按照 删除数据库实例 中的说明操作。您无需创建最终快照。
- 要终止 Amazon EC2 实例，请按照 Amazon EC2 用户指南中的终止实例中的说明操作。

教程：使用 Lambda 函数访问 Amazon RDS 数据库

在本教程中，您使用 Lambda 函数通过 RDS 代理将数据写入 Amazon Relational Database Service（Amazon RDS）数据库。您的 Lambda 函数在 Amazon Simple Queue Service（Amazon SQS）队列中读取记录，每当添加消息时，都将新的项目写入您数据库的表中。在此示例中，您使用 Amazon Web Services Management Console 手动向队列添加消息。下图显示了您用于完成教程的 Amazon 资源。



借助 Amazon RDS，您可以使用 Microsoft SQL Server、MariaDB、MySQL、Oracle Database 和 PostgreSQL 等常见数据库产品在云中运行托管式关系数据库。通过使用 Lambda 访问您的数据库，您可以读取和写入数据以响应事件，例如在您的网站上注册的新客户。您的函数、数据库实例和代理可以自动扩展以满足高需求时段。

要完成本教程，请执行以下任务：

1. 在您的 Amazon Web Services 账户的原定设置 VPC 中启动 RDS for MySQL 数据库实例和代理。
2. 创建并测试 Lambda 函数，该函数在您的数据库中创建新表并将数据写入其中。
3. 创建 Amazon SQS 队列并将其配置为在添加新消息时调用您的 Lambda 函数。
4. 使用 Amazon Web Services Management Console向队列添加消息并使用 CloudWatch Logs 监控结果，来测试完整设置。

通过完成这些步骤，您将学习：

- 如何使用 Amazon RDS 创建数据库实例和代理，并将 Lambda 函数连接到代理。
- 如何使用 Lambda 对 Amazon RDS 数据库执行创建和读取操作。
- 如何使用 Amazon SQS 调用 Lambda 函数。

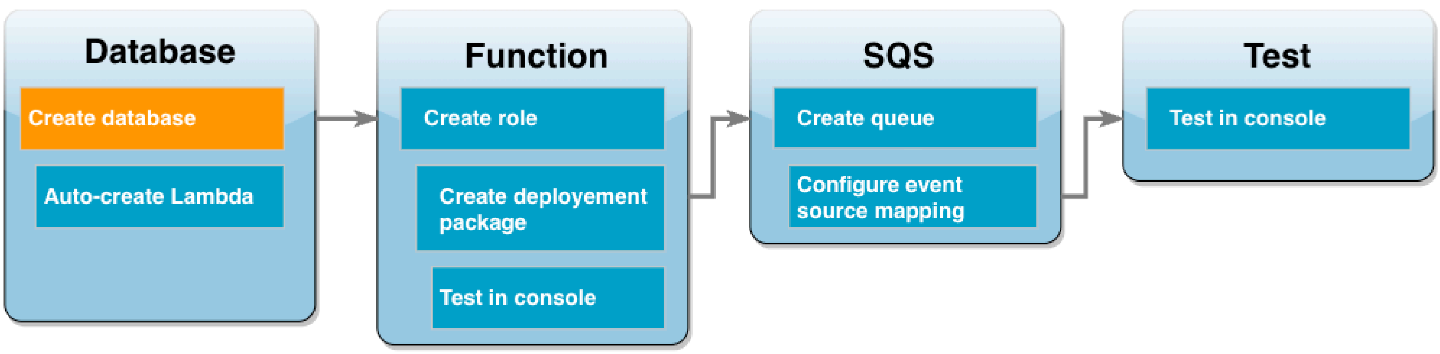
您可以使用 Amazon Web Services Management Console 或 Amazon Command Line Interface (Amazon CLI) 完成此教程。

先决条件

在开始之前，请完成以下各节中的步骤：

- 注册 Amazon Web Services 账户
- 保护 IAM 用户

创建 Amazon RDS 数据库实例



Amazon RDS 数据库实例是在 Amazon Web Services 云 中运行的独立数据库环境。实例可以包含一个或多个由用户创建的数据库。除非您另行指定，否则 Amazon RDS 会在您的 Amazon Web Services 账户包含的原定设置 VPC 中创建新的数据库实例。有关 Amazon VPC 的更多信息，请参阅 Amazon Virtual Private Cloud 用户指南。

在本教程中，您在您 Amazon Web Services 账户 的默认 VPC 中创建一个新实例，并在该实例中创建一个名为 `ExampleDB` 的数据库。您可以使用 Amazon Web Services Management Console或 Amazon CLI 创建数据库实例和数据库。

创建数据库实例

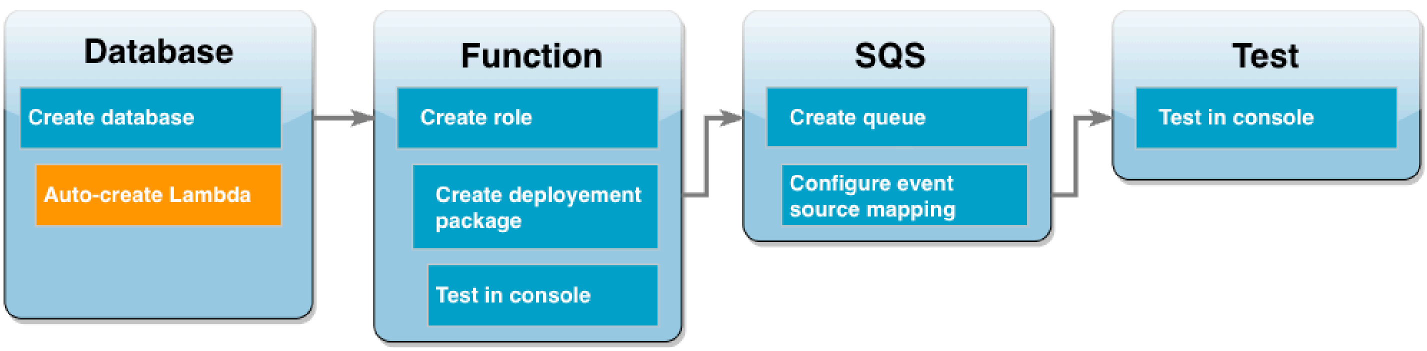
1. 打开 Amazon RDS 控制台并选择创建数据库。

2. 保持标准创建选项处于选中状态，然后在引擎选项中选择 MySQL。
3. 在模板部分中，选择免费套餐。
4. 在设置中，为 DB 实例标识符输入 **MySQLForLambda**。
5. 请通过执行以下操作设置用户名和密码：

   a. 在凭证设置中，将主用户名设置为 admin。
   b. 对于主密码，输入并确认密码以访问您的数据库。
6. 通过执行以下操作指定数据库名称：

   - 将所有剩余的原定设置选项保持选中状态，然后向下滚动到其他配置部分。
   - 展开此部分并输入 **ExampleDB** 作为初始数据库名称。
7. 保持所有其余默认选项处于选中状态，然后选择创建队列。

创建 Lambda 函数和代理



您可以使用 RDS 控制台在与数据库相同的 VPC 中创建 Lambda 函数和代理。

> Note
> 只有在数据库完成创建并处于可用状态时，才能创建这些关联的资源。

创建关联函数和代理

1. 从数据库页面，检查您的数据库是否处于可用状态。如果是，请继续执行下一步。否则，请等到您的数据库变为可用。
2. 选择您的数据库，并从操作中选择设置 Lambda 连接。
3. 在设置 Lambda 连接页面上，选择创建新函数。

将新的 Lambda 函数名称设置为 **LambdaFunctionWithRDS**。

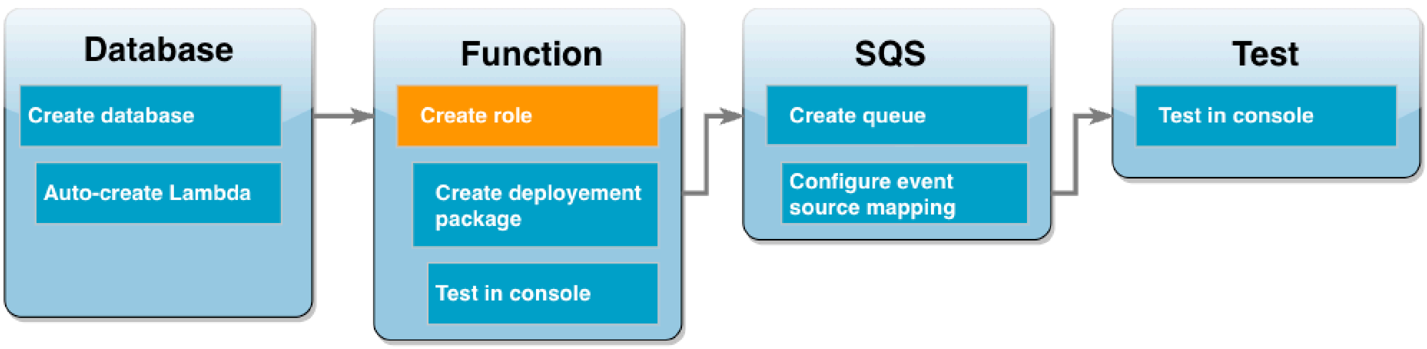
4. 在 RDS 代理部分，选择使用 RDS 代理进行连接选项。进一步选择创建新代理。

   - 对于数据库凭证，选择数据库用户名和密码。
   - 对于用户名，指定 admin。
   - 对于密码，输入您为数据库实例创建的密码。

5. 选择设置以完成代理和 Lambda 函数的创建。

向导完成设置，并提供指向 Lambda 控制台的链接以查看您的新函数。在切换到 Lambda 控制台之前，请记下代理端点。

创建函数执行角色



在创建 Lambda 函数之前，您需要创建一个执行角色来为您的函数提供必要的权限。在本教程中，Lambda 需要权限来管理与包含您数据库实例的 VPC 的网络连接，以及轮询来自 Amazon SQS 队列的消息。

为了向您的 Lambda 函数提供其所需的权限，本教程使用 IAM 托管策略。这些策略可授予许多常见使用案例的权限，可在您的 Amazon Web Services 账户 中使用。有关使用托管策略的更多信息，请参阅策略最佳实践。

创建 Lambda 执行角色

1. 打开 IAM 控制台的角色页面，然后选择创建角色。
2. 对于受信任的实体类型，选择 Amazon 服务，对于使用案例，选择 Lambda。
3. 选择下一步。
4. 通过执行以下操作添加 IAM 托管策略：

   a. 使用策略搜索框，搜索 **AWSLambdaSQSQueueExecutionRole**。

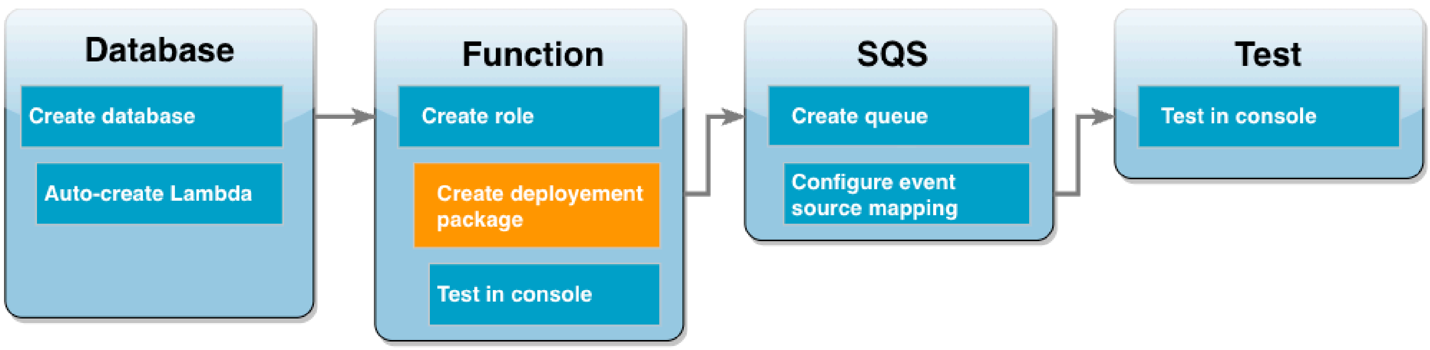
   b. 在结果列表中，选中角色旁的复选框，然后选择清除筛选条件。
   c. 使用策略搜索框，搜索 **AWSLambdaVPCAccessExecutionRole**。
   d. 在结果列表中，选中角色旁的复选框，然后选择下一步。
5. 对于角色名称，输入 **lambda-vpc-sqs-role**，然后选择创建角色。

在本教程的后面部分，您需要提供刚刚创建的执行角色的 Amazon 资源名称（ARN）。

查找执行角色 ARN

1. 打开 IAM 控制台的[角色]页面，然后选择您的角色（`lambda-vpc-sqs-role`）。
2. 复制摘要部分中显示的 ARN。

创建 Lambda 部署包



以下示例 Python 代码使用 [PyMySQL] 包打开与您的数据库的连接。首次调用函数时，它还会创建一个名为 `Customer` 的新表。该表使用以下架构，其中 `CustID` 是主键：

```
Customer(CustID, Name)
```

该函数还使用 PyMySQL 向该表添加记录。该函数使用您将添加到 Amazon SQS 队列的消息中指定的客户 ID 和名称来添加记录。

该代码在处理程序函数之外创建了与数据库的连接。在初始化代码中创建连接，使后续调用能够重用该连接并提高性能。在生产应用程序中，您还可以使用[预置并发]来初始化所请求数量的数据库连接。调用函数后，这些连接即可用。

```
import sys
import logging
import pymysql
```

```
import json
import os

# rds settings
user_name = os.environ['USER_NAME']
password = os.environ['PASSWORD']
rds_proxy_host = os.environ['RDS_PROXY_HOST']
db_name = os.environ['DB_NAME']

logger = logging.getLogger()
logger.setLevel(logging.INFO)

# create the database connection outside of the handler to allow connections to be
# re-used by subsequent function invocations.
try:
        conn = pymysql.connect(host=rds_proxy_host, user=user_name, passwd=password,
 db=db_name, connect_timeout=5)
except pymysql.MySQLError as e:
    logger.error("ERROR: Unexpected error: Could not connect to MySQL instance.")
    logger.error(e)
    sys.exit(1)

logger.info("SUCCESS: Connection to RDS for MySQL instance succeeded")

def lambda_handler(event, context):
    """
    This function creates a new RDS database table and writes records to it
    """
    message = event['Records'][0]['body']
    data = json.loads(message)
    CustID = data['CustID']
    Name = data['Name']

    item_count = 0
    sql_string = f"insert into Customer (CustID, Name) values(%s, %s)"

    with conn.cursor() as cur:
        cur.execute("create table if not exists Customer ( CustID  int NOT NULL, Name
 varchar(255) NOT NULL, PRIMARY KEY (CustID))")
        cur.execute(sql_string, (CustID, Name))
        conn.commit()
        cur.execute("select * from Customer")
        logger.info("The following items have been added to the database:")
        for row in cur:
```

```
            item_count += 1
            logger.info(row)
    conn.commit()

    return "Added %d items to RDS for MySQL table" %(item_count)
```

Note

在此示例中，您的数据库访问凭证将存储为环境变量。在生产应用程序中，建议您将 Amazon Secrets Manager 用作更安全的选项。请注意，如果 Lambda 函数位于 VPC 中，您需要创建 VPC 端点才能连接到 Secrets Manager。要了解更多信息，请参阅如何在虚拟私有云中连接到 Secrets Manager 服务。

要在函数代码中包含 PyMySQL 依赖关系，请创建一个 .zip 部署包。以下命令适用于 Linux、macOS 或 Unix：

创建 .zip 部署包

1. 将代码示例保存为名为 `lambda_function.py` 的文件。
2. 在创建 `lambda_function.py` 文件的同一目录中，创建一个名为 `package` 的新目录并安装 PyMySQL 库。

```
mkdir package
pip install --target package pymysql
```

3. 创建一个包含您的应用程序代码和 PyMySQL 库的 zip 文件。在 Linux 或 MacOS 中，运行以下 CLI 命令。在 Windows 中，使用您首选的压缩工具来创建 `lambda_function.zip` 文件。您的 `lambda_function.py` 源代码文件和包含依赖项的文件夹必须安装在.zip 文件的根目录下。

```
cd package
zip -r ../lambda_function.zip .
cd ..
zip lambda_function.zip lambda_function.py
```

您也可以使用 Python 虚拟环境创建部署包。参阅使用 .zip 文件归档部署 Python Lambda 函数。

更新 Lambda 函数

现在，您可以使用刚创建的 .zip 程序包，通过 Lambda 控制台更新 Lambda 函数。要支持函数访问数据库，您还需要使用访问凭证配置环境变量。

更新 Lambda 函数

1. 打开 Lambda 控制台的函数页面，然后选择您的函数 `LambdaFunctionWithRDS`。
2. 在运行时系统设置选项卡中，选择编辑，以将函数的运行时系统更改为 Python 3.10。
3. 将处理程序更改为 `lambda_function.lambda_handler`。
4. 在代码选项卡中，选择上传自，然后选择 .zip 文件。
5. 选择您在前一阶段创建的 `lambda_function.zip` 文件，然后选择保存。

现在，使用您之前创建的执行角色来配置该函数。这会授予该函数访问您的数据库实例和轮询 Amazon SQS 队列所需的权限。

配置函数的执行角色

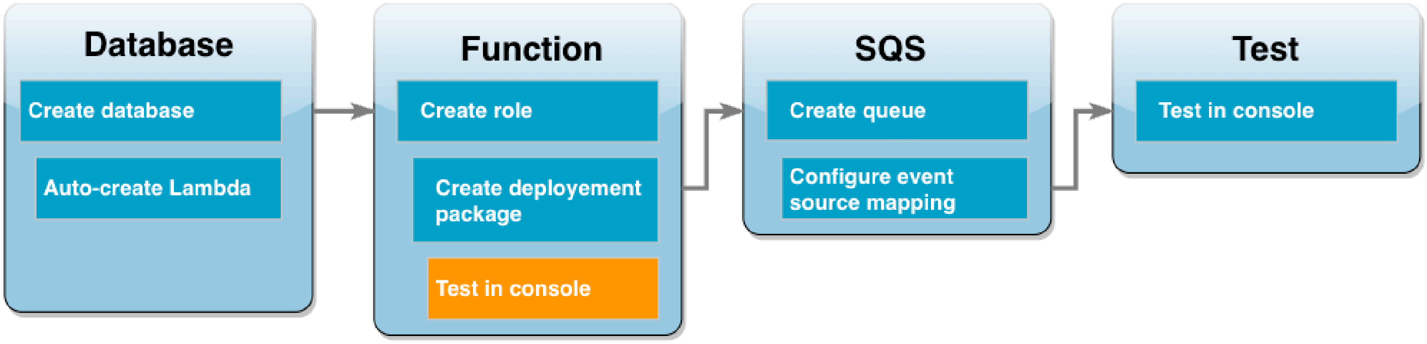
1. 在 Lambda 控制台的函数页面中，选择配置选项卡，然后选择权限。
2. 在执行角色中，选择编辑。
3. 在现有角色中，选择您的执行角色（`lambda-vpc-sqs-role`）。
4. 选择保存。

配置函数的环境变量

1. 在 Lambda 控制台的函数页面中，选择配置选项卡，然后选择环境变量。
2. 选择编辑。
3. 要添加数据库访问凭证，请执行以下操作：

   a. 选择添加环境变量，然后为键输入 **`USER_NAME`**，并为值输入 **`admin`**。
   b. 选择添加环境变量，然后为键输入 **`DB_NAME`**，并为值输入 **`ExampleDB`**。
   c. 选择添加环境变量，然后为键输入 **`PASSWORD`**，并为值输入您在创建数据库时选择的密码。
   d. 选择添加环境变量，然后为键输入 **`RDS_PROXY_HOST`**，并为值输入您之前记下的 RDS 代理端点。
   e. 选择保存。

在控制台中测试 Lambda 函数。



您现在可以使用 Lambda 控制台测试您的函数。您可以创建一个测试事件，该事件模仿您在教程的最后阶段使用 Amazon SQS 调用函数时将收到的数据。您的测试事件包含一个 JSON 对象，该对象指定要添加到您的函数创建的 `Customer` 表中的客户 ID 和客户名称。

测试 Lambda 函数

1. 打开 Lambda 控制台的函数页面，然后选择一个函数。
2. 选择测试部分。
3. 选择创建新事件，然后输入 **`myTestEvent`** 作为事件名称。
4. 将以下代码复制到事件 JSON 中，然后选择保存。

```
{
  "Records": [
    {
      "messageId": "059f36b4-87a3-44ab-83d2-661975830a7d",
      "receiptHandle": "AQEBwJnKyrHigUMZj6rYigCgxlaS3SLy0a...",
      "body": "{\n     \"CustID\": 1021,\n     \"Name\": \"Martha Rivera\"\n}",
      "attributes": {
        "ApproximateReceiveCount": "1",
        "SentTimestamp": "1545082649183",
        "SenderId": "AIDAIENQZJOLO23YVJ4VO",
        "ApproximateFirstReceiveTimestamp": "1545082649185"
      },
      "messageAttributes": {},
      "md5OfBody": "e4e68fb7bd0e697a0ae8f1bb342846b3",
      "eventSource": "aws:sqs",
      "eventSourceARN": "arn:aws:sqs:us-west-2:123456789012:my-queue",
      "awsRegion": "us-west-2"
    }
  ]
```

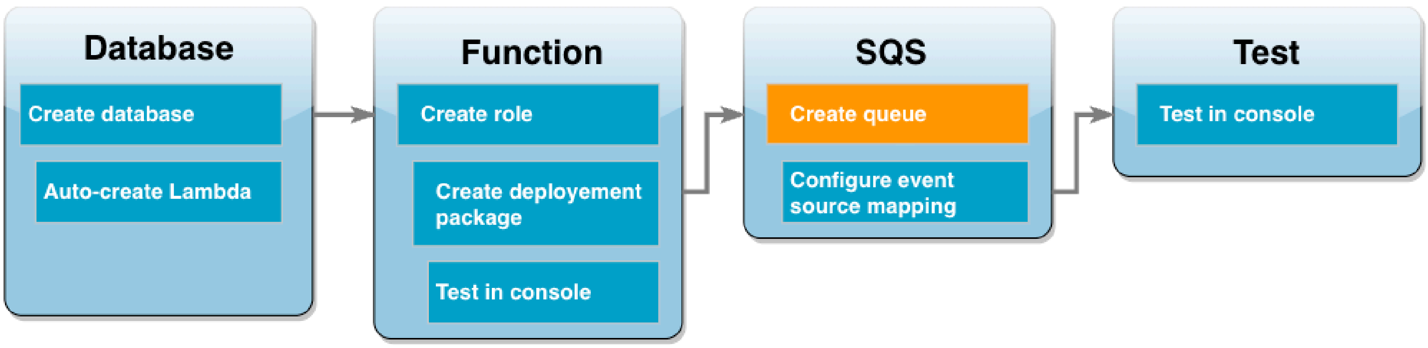
```
}
```

5. 选择测试。

在执行结果选项卡中，您应该看到与函数日志中显示的以下内容类似的结果：

```
[INFO] 2023-02-14T19:31:35.149Z bdd06682-00c7-4d6f-9abb-89f4bbb4a27f The following
 items have been added to the database:
[INFO] 2023-02-14T19:31:35.149Z bdd06682-00c7-4d6f-9abb-89f4bbb4a27f (1021, 'Martha
 Rivera')
```

创建 Amazon SQS 队列

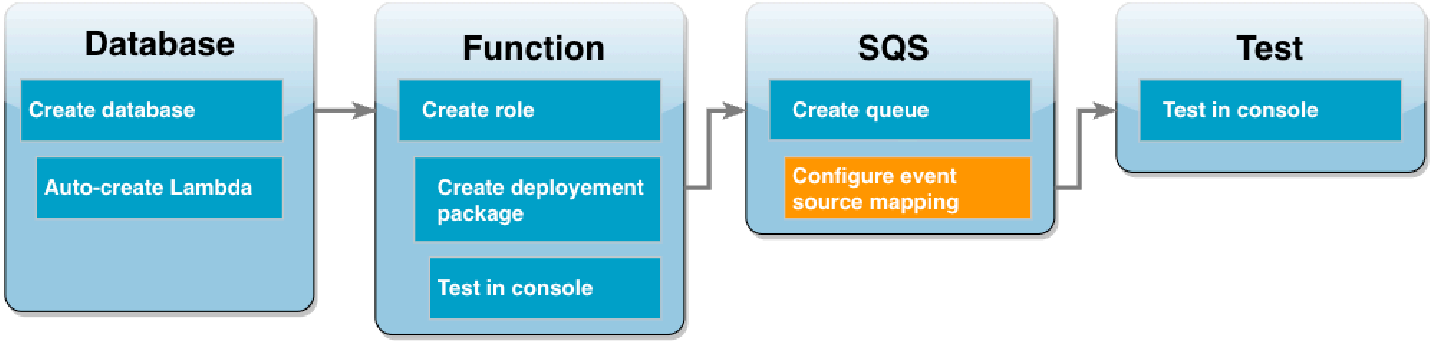


您已成功测试了您的 Lambda 函数和 Amazon RDS 数据库实例的集成。现在，您创建了 Amazon SQS 队列，用于在本教程的最后阶段调用 Lambda 函数。

创建 Amazon SQS 队列（控制台）。

1. 打开 Amazon SQS 控制台的队列页面，然后选择创建队列。
2. 将类型保留为标准，然后为您的队列名称输入 **LambdaRDSQueue**。
3. 保持所有默认选项处于选中状态，然后选择创建队列。

创建一个事件源映射以调用您的 Lambda 函数



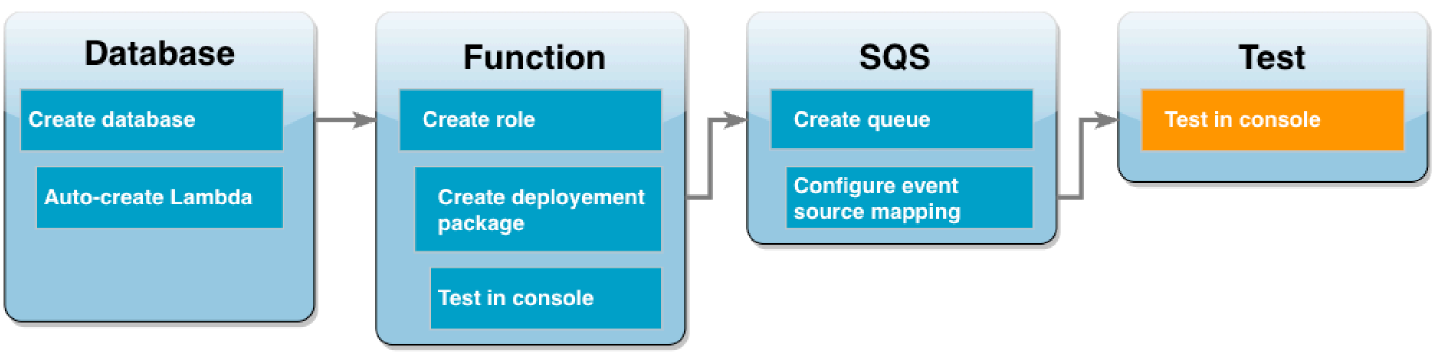
事件源映射是 Lambda 中的一种资源，它从流或队列中读取项目并调用 Lambda 函数。配置事件源映射时，可以指定批处理大小，以便可以将来自您的流或队列的记录批处理成单个负载。在此示例中，您将批处理大小设置为 1，这样每次向队列发送消息时都会调用 Lambda 函数。您可以使用 Amazon CLI 或 Lambda 控制台配置事件源映射。

创建事件源映射（控制台）

1. 打开 Lambda 控制台的函数页面，然后选择一个函数 (`LambdaFunctionWithRDS`)。
2. 在函数概述部分中，选择添加触发器。
3. 对于源，选择 Amazon SQS，然后选择队列的名称（`LambdaRDSQueue`）。
4. 对于批处理大小，输入 **1**。
5. 将所有其他选项设置为默认值，然后选择添加。

现在，您可以通过向 Amazon SQS 队列添加一条消息来测试您的完整设置。

测试和监控您的设置



要测试您的完整设置，请使用控制台向 Amazon SQS 队列添加消息。然后，您可以使用 CloudWatch Logs 确认您的 Lambda 函数正在按预期将记录写入数据库。

测试和监控您的设置

1. 打开 Amazon SQS 控制台的队列页面，然后选择创建队列 (LambdaRDSQueue)。
2. 选择发送和接收消息，然后将以下 JSON 粘贴到发送消息部分的消息正文中。

```
{
    "CustID": 1054,
    "Name": "Richard Roe"
}
```

3. 选择 Send message（发送消息）。

   将您的消息发送到队列将导致 Lambda 通过您的事件源映射调用您的函数。要确认 Lambda 已按预期调用您的函数，请使用 CloudWatch Logs 验证该函数是否已将客户名称和 ID 写入您的数据库表。
4. 打开 CloudWatch 控制台的日志组页面，为您的函数选择日志组 (/aws/lambda/LambdaFunctionWithRDS)。
5. 在日志流部分中，选择最新的日志流。

   您的表应包含两条客户记录，每次调用您的函数都有一条记录。在日志流中，您应看到类似以下内容的消息：

```
[INFO] 2023-02-14T19:06:43.873Z 45368126-3eee-47f7-88ca-3086ae6d3a77 The following
 items have been added to the database:
[INFO] 2023-02-14T19:06:43.873Z 45368126-3eee-47f7-88ca-3086ae6d3a77 (1021, 'Martha
 Rivera')
[INFO] 2023-02-14T19:06:43.873Z 45368126-3eee-47f7-88ca-3086ae6d3a77 (1054,
 'Richard Roe')
```

清除资源

除非您想要保留为本教程创建的资源，否则可立即将其删除。通过删除您不再使用的 Amazon 资源，可防止您的 Amazon 账户产生不必要的费用。

删除 Lambda 函数

1. 打开 Lamba 控制台的 Functions（函数）页面。
2. 选择您创建的函数。

3. 依次选择 Actions（操作）和 Delete（删除）。
4. 选择 Delete（删除）。

删除执行角色

1. 打开 IAM 控制台的角色页面。
2. 选择您创建的执行角色。
3. 选择 Delete role（删除角色）。
4. 选择 Yes, delete（是，删除）。

删除 MySQL 数据库实例

1. 打开 Amazon RDS 控制台的 Databases（数据库）页面。
2. 选择您创建的数据库。
3. 依次选择 Actions（操作）和 Delete（删除）。
4. 清除 Create final snapshot（创建最终快照）复选框。
5. 在文本框中输入 **`delete me`**。
6. 选择 Delete（删除）。

删除 Amazon SQS 队列

1. 登录到 Amazon Web Services Management Console 并打开 Amazon SQS 控制台，网址：https://console.aws.amazon.com/vpc/。
2. 选择创建的队列。
3. 选择 Delete（删除）。
4. 在文本框中输入 **`delete`**。
5. 选择 Delete（删除）。

Amazon RDS 教程和示例代码

Amazon 文档包括多个教程，可指导您完成常见的 Amazon RDS 使用案例。这些教程中的许多教程都向您展示了如何将 Amazon RDS 与其他Amazon服务一起使用。此外，您可以在 GitHub 中访问示例代码。

> **Note**
>
> 您可以在Amazon数据库博客中找到更多教程。有关培训的信息，请参阅 Amazon Training and Certification。

主题

本指南中的教程

本指南中的以下教程为您演示如何使用 Amazon RDS 执行常见任务。

- 教程：创建 VPC 以用于数据库实例（仅限 IPv4）

  了解如何在虚拟私有云（VPC）中基于 Amazon VPC 服务包括数据库实例。在这种情况下，VPC 与在同一 VPC 中的 Amazon EC2 实例上运行的 Web 服务器共享数据。
- 教程：创建 VPC 以用于数据库实例（双堆栈模式）

  了解如何在虚拟私有云（VPC）中基于 Amazon VPC 服务包括数据库实例。在这种情况下，VPC 与同一 VPC 中的 Amazon EC2 实例共享数据。在本教程中，您将为此场景创建 VPC，该 VPC 与在双堆栈模式下运行的数据库一起使用。
- 教程：创建 Web 服务器和 Amazon RDS 数据库实例

了解如何使用 PHP 安装 Apache Web 服务器并创建 MySQL 数据库。Web 服务器在使用 Amazon Linux 的 Amazon EC2 实例上运行，而 MySQL 数据库是 MySQL 数据库实例。Amazon EC2 实例和数据库实例都在 Amazon VPC 中运行。

- 教程：从数据库快照还原 Amazon RDS 数据库实例

  了解如何从数据库快照还原数据库实例。

- 教程：使用 Lambda 函数访问 Amazon RDS 数据库

  了解如何从 RDS 控制台创建一个 Lambda 函数，以通过代理访问数据库，创建一个表，添加几个记录，然后检索表中的记录。您还将学习如何调用 Lambda 函数并验证查询结果。

- 教程：使用标签指定要停止的数据库实例

  了解如何使用标签指定要停止的数据库实例。

- 教程：使用 Amazon EventBridge 记录数据库实例的状态更改

  了解如何使用 Amazon EventBridge 和 Amazon Lambda 记录数据库实例阶段更改。

- 教程：为 Amazon RDS 的多可用区数据库集群副本滞后创建 Amazon CloudWatch 警报

  了解如何创建 CloudWatch 告警，使其在多可用区数据库集群的副本滞后超过阈值时发送 Amazon SNS 消息。告警会在您指定的时间范围内监控 `ReplicaLag` 指标。操作是向 Amazon SNS 主题或 Amazon EC2 Auto Scaling 策略发送的通知。

其他 Amazon 指南中的教程

其他Amazon指南中的以下教程为您演示了如何使用 Amazon RDS 执行常见任务：

- Amazon Secrets Manager 用户指南中的教程：为 Amazon 数据库轮换密钥

  了解如何为 Amazon 数据库创建密钥并将密钥配置为按计划轮换。您手动触发一次轮换，然后确认密钥的新版本能否继续提供访问权限。

- Amazon Elastic Beanstalk 开发人员指南中的教程和示例

  了解如何部署将 Amazon RDS 数据库与Amazon Elastic Beanstalk结合使用的应用程序。

- Amazon Machine Learning Developer Guide 中的使用来自 Amazon RDS 数据库的数据创 Amazon ML 数据源

了解如何从存储在 MySQL 数据库实例中的数据创建 Amazon Machine Learning (Amazon ML) 数据源对象。

- Amazon QuickSight 用户指南 中的在 VPC 中手动启用 Amazon RDS 实例访问权限

  了解如何启用 QuickSight 对 VPC 中的 Amazon RDS 数据库实例的访问权限。

Amazon RDS PostgreSQL 的 Amazon 研讨会和实验室内容门户

以下一组研讨会和其他实践内容集可帮助您了解 Amazon RDS PostgreSQL 的特征和功能：

- 创建数据库实例

  了解如何创建数据库实例。

- 使用 RDS 工具监控性能

  了解如何使用 Amazon 和 SQL 工具（Cloudwatch、增强监控、慢速查询日志、Performance Insights、PostgreSQL 目录视图）来了解性能问题，并确定提高数据库性能的方法。

Amazon RDS MySQL 的 Amazon 研讨会和实验室内容门户

以下一组研讨会和其他实践内容集可帮助您了解 Amazon RDS MySQL 的特征和功能：

- 创建数据库实例

  了解如何创建数据库实例。

- 使用性能详情

  了解如何使用性能详情来监控和调整数据库实例。

GitHub 中的教程和示例代码

GitHub 中的以下教程和示例代码为您演示如何使用 Amazon RDS 执行常见任务：

- 创建 Amazon Relational Database Service 项目追踪器

  了解如何创建用于跟踪和报告工作项目的应用程序。此应用程序使用 Amazon RDS、Amazon Simple Email Service、Elastic Beanstalk 和 SDK for Java 2.x。

将此服务与 Amazon SDK 结合使用

Amazon 软件开发工具包（SDK）适用于许多常用编程语言。每个软件开发工具包都提供 API、代码示例和文档，使开发人员能够更轻松地以其首选语言构建应用程序。

SDK 文档
Amazon CLI
适用于 Java 的 Amazon SDK
适用于 JavaScript 的 Amazon SDK
适用于 .NET 的 Amazon SDK
适用于 PHP 的 Amazon SDK
Amazon Tools for PowerShell
适用于 Python (Boto3) 的 Amazon SDK
适用于 Ruby 的 Amazon SDK
适用于 SAP ABAP 的 Amazon SDK

有关特定于此服务的示例，请参阅适用于使用 Amazon SDK 的 Amazon RDS 的代码示例。

Amazon RDS 的最佳实践

了解使用 Amazon RDS 的最佳实践。随着新的最佳实践的确定，我们将持续更新此部分。

主题

> Note
>
> 有关 Amazon RDS 的常见建议，请参阅 来自 Amazon RDS 的建议。

Amazon RDS 基本操作指导方针

以下是使用 Amazon RDS 时每个人都应遵循的基本操作指导方针。请注意，Amazon RDS 服务等级协议要求您遵循以下指导方针：

- 使用指标监控您的内存、CPU、副本滞后和存储使用情况。可将 Amazon CloudWatch 设置为在使用模式发生变化或当部署接近容量限制时向您发送通知。这样，您就可以保持系统的性能和可用性。

- 当接近存储容量限制时，可以纵向扩展数据库实例。存储和内存中应含有一些缓冲区，以适应应用程序的意外增大需求。
- 启用自动备份并设置备份时段，以在每天写入 IOPS 较低的时段进行。这时备份对数据库使用的影响最小。
- 如果您的数据库工作负载需要的 I/O 超过您的配置，那么出现故障转移或数据库故障后，恢复的速度将会变缓。要增加数据库实例的 I/O 容量，请执行以下任一或所有操作：
  - 迁移到其他具有高 I/O 容量的数据库实例类。
  - 从磁性存储转换为通用存储或预置 IOPS 存储，具体取决于您需要增加的量。有关可用存储类型的信息，请参阅 Amazon RDS 存储类型。

    如果转换为配置的 IOPS 存储，请确保还使用已针对配置的 IOPS 进行优化的数据库实例类。有关配置的 IOPS 的信息，请参阅 预置 IOPS SSD 存储。
  - 如果您已在使用配置的 IOPS 存储，请额外配置吞吐量容量。
- 如果您的客户端应用程序正在缓存数据库实例的域名服务 (DNS) 数据，请将生存时间 (TTL) 值设置为小于 30 秒。数据库实例的底层 IP 地址在故障转移后可能会发生变化。因此，长时间缓存 DNS 数据可能会导致连接故障。您的应用程序可能会尝试连接到不再使用的 IP 地址。
- 测试数据库实例的故障转移，以了解对于您的特定使用案例而言，该过程需要多长时间。此外，测试故障转移，以确保访问您数据库实例的应用程序在故障转移发生之后，可以自动连接到新数据库实例。

数据库实例 RAM 建议

Amazon RDS 性能最佳实践是分配足够的 RAM，以便您的工作集几乎完全驻留在内存中。工作集是经常在实例上使用的数据和索引。使用数据库实例的次数越多，工作集的增长量就越大。

若要确定您的工作集是否几乎完全位于内存中，请在数据库实例加载期间检查 ReadIOPS 指标 (使用 Amazon CloudWatch)。ReadIOPS 的值应是一个较小且稳定的值。在某些情况下，将数据库实例类纵向扩展为具有更多 RAM 的类会导致 ReadIOPS 大幅降低。在这些情况下，您的工作集几乎不完全在内存中。继续向上扩展直至 ReadIOPS 不再在扩展操作后大幅降低，否则 ReadIOPS 将降低至非常小的数量。有关监控数据库实例的指标的信息，请参阅在 Amazon RDS 控制台中查看指标。

使数据库引擎版本保持最新

定期升级数据库引擎版本，以保持安全性、性能和合规性。Amazon RDS 会发布新的次要版本和主要版本，其中包括安全补丁、性能增强和新功能。运行过时的数据库引擎可能会使您的工作负载暴露于已知的漏洞、兼容性问题，以及来自 Amazon 和数据库供应商的支持减少。

为了最大限度地减少中断，在计划升级时，请考虑以下几点：

- 在暂存环境中进行测试：在升级生产数据库之前，对照您的工作负载验证新版本。
- 使用 Amazon RDS 托管式升级：启用自动次要版本升级，便于修补。
- 安排主要版本升级：查看发布说明，测试应用程序兼容性，并规划受控的升级时段。

定期升级有助于确保您的数据库保持安全、优化并符合 Amazon 最佳实践。

Amazon 数据库驱动程序

推荐使用 Amazon 驱动程序套件来建立应用程序连接。借助这些驱动程序可显著缩短切换和故障转移时间，并支持使用 Amazon Secrets Manager、Amazon Identity and Access Management（IAM）和联合身份进行身份验证。Amazon 驱动程序依靠监控数据库实例状态和了解实例拓扑，来确定新的写入器。这种方法将切换和故障转移时间缩短到几秒钟，而开源驱动程序的切换和故障转移时间则为几十秒。

随着新服务功能的推出，使用 Amazon 驱动程序套件可为这些服务功能提供内置支持。

有关更多信息，请参阅 [使用 Amazon 驱动程序连接到数据库实例]。

使用增强监控来确定操作系统问题

启用增强监测后，Amazon RDS 可提供运行数据库实例的操作系统 (OS) 的实时指标。您可以使用控制台查看数据库实例的指标。您还可以在您选择的监控系统中通过 Amazon CloudWatch Logs 使用增强型监控 JSON 输出。有关增强监控的更多信息，请参阅 [使用增强监控来监控操作系统指标]。

使用指标确定性能问题

要确定资源不足和其他常见瓶颈导致的性能问题，您可以监控可用于 Amazon RDS 数据库实例的指标。

查看性能指标

您应定期监控性能指标以查看各种时间范围内的平均值、最大值和最小值。通过这样做，您可以确定性能下降的时间。您还可以针对特定指标阈值设置 Amazon CloudWatch 警报，以便在达到这些阈值时向您发出警报。

要排除性能问题，了解系统的基准性能十分重要。当您设置数据库实例并在典型工作负载下运行该实例时，请捕获所有性能指标的平均值、最大值和最小值。按多个不同的间隔（例如，一小时、24 小时、

一周、两周）执行此操作。这使您能够了解正常运行状况。这有助于将操作的峰值时间与非峰值时间进行比较。您随后可以利用这些信息确定性能何时降到标准水平以下。

如果使用多可用区数据库集群，则监控写入器数据库实例上的最新事务与读取器数据库实例上最新应用的事务之间的时间差异。这种差异称为副本滞后。有关更多信息，请参阅 副本滞后和多可用区数据库集群。

您可以在 Performance Insights 控制面板中查看组合的 Performance Insights 和 CloudWatch 指标，并监控您的数据库实例。为使用此监控视图，必须为您的数据库实例开启 Performance Insights。有关监控视图的信息，请参阅使用性能详情控制面板查看组合指标。

您可以创建特定时间段的性能分析报告，并查看已确定的见解和解决问题的建议。有关更多信息，请参阅在性能详情中创建性能分析报告。

查看性能指标

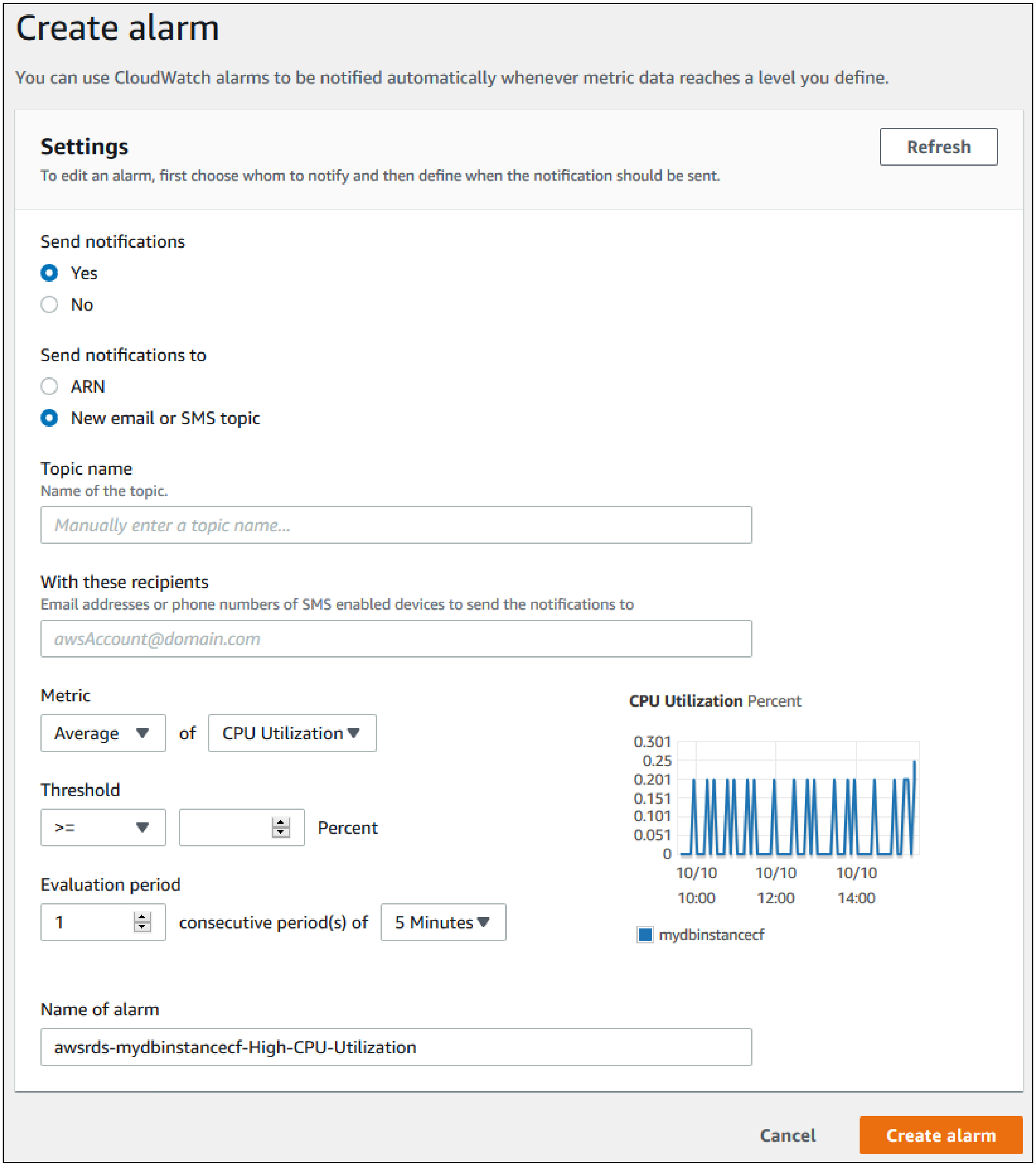
1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)，然后选择数据库实例。
3. 选择监控。

   控制面板提供性能指标。指标原定设置为显示最近三小时的信息。
4. 使用页面右上角的编号按钮翻阅其他指标，或调整设置以查看更多指标。
5. 可以选择性能指标来调整时间范围，以便查看除了当日之外的数据。可以更改 Statistic、Time Range 和 Period 值以调整显示的信息。例如，您可能希望查看某个指标在最近两周内每天的峰值。如果是这样，请将 Statistic（统计数据）设置为 Maximum（最大值），将 Time Range（时间范围）设置为 Last 2 Weeks（最近 2 周），将 Period（时段）设置为 Day（天）。

还可以使用 CLI 或 API 查看性能指标。有关更多信息，请参阅“在 Amazon RDS 控制台中查看指标”。

设置 CloudWatch 警报

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)，然后选择数据库实例。
3. 选择日志和事件。
4. 在 CloudWatch 警报部分中，选择创建警报。



5. 对于发送通知，选择是；对于发送通知到，选择新电子邮件或 SMS 主题。

6. 对于主题名称，输入通知的名称；对于收件人如下，输入以逗号分隔的电子邮件地址和电话号码列表。

7. 对于指标，选择要设置的警报统计信息和指标。
8. 对于阈值，指定指标必须大于、小于还是等于阈值，然后指定阈值。
9. 对于 Evaluation period（评估期），选择告警的评估期。对于 consecutive period(s) of（个连续周期，时间为），选择必须达到阈值才能触发告警的时段。
10. 对于 Name of alarm (警报名称)，输入警报的名称。
11. 选择 Create Alarm。

警报将显示在 CloudWatch 警报部分中。

评估性能指标

数据库实例有一些不同类别的指标，确定可接受的值的方式取决于具体的指标。

CPU

- CPU 使用率 – 使用的计算机处理容量的百分比。

内存

- 可释放内存 – 数据库实例上可用的 RAM 量（以字节为单位）。对于 CPU、内存和存储指标，“监控”选项卡中的红线指标标记为 75%。如果实例内存消耗频繁越过红线，则这指示您应检查您的工作负载或升级您的实例。
- 交换区使用情况 – 数据库实例使用的交换空间量（以字节为单位）。

磁盘空间

- 可用存储空间 – 数据库实例当前未使用的磁盘空间量（以 MB 为单位）。

输入/输出操作

- 读取 IOPS，写入 IOPS – 每秒进行的磁盘读取或写入操作平均数。
- 读取延迟，写入延迟 – 读取或写入操作的平均时间（以毫秒为单位）。
- 读取吞吐量，写入吞吐量 – 每秒对磁盘读取或写入的平均兆字节数。
- 队列深度 – 等待对磁盘写入或读取的 I/O 操作数。

网络流量

- 网络接收吞吐量，网络传输吞吐量 – 每秒出入数据库实例的网络流量速率（以每秒字节数为单位）。

数据库连接

- 数据库连接 – 连接到数据库实例的客户端会话数。

有关可用的性能指标的各个更详细的说明，请参阅使用 Amazon CloudWatch 监控 Amazon RDS 指标。

一般而言，性能指标的可接受值取决于您的基准性能以及应用程序执行的操作。应调查相对于基准性能的一致或趋势性变化。有关特定类型指标的建议如下：

- High CPU or RAM consumption（高 CPU 或 RAM 消耗）– CPU 或 RAM 消耗值高可能是正常情况。例如，如果它们符合您的应用程序目标（如吞吐量或并发度）并且符合预期，则此类消耗值可能就是高。
- 磁盘空间消耗 – 如果使用的空间始终不低于总磁盘空间的 85%，则应调查磁盘空间消耗。应查看是否可以从实例中删除数据或是将数据存档到其他系统以释放空间。
- 网络流量 – 对于网络流量，应与系统管理员进行讨论，以了解域网络和 Internet 连接的预期吞吐量。如果吞吐量始终低于预期，则应调查网络流量。
- 数据库连接 – 如果发现用户连接数较高，同时实例性能下降并且响应时间延长，请考虑约束数据库连接。数据库实例的最佳用户连接数因您的实例类所执行操作的复杂性而异。要确定数据库连接的数量，请将数据库实例与参数组关联。在此组中，将 User Connections（用户连接）参数设置为 0 以外的值（无限制）。您可以使用现有参数组或新建一个。有关更多信息，请参阅“Amazon RDS 的参数组”。
- IOPS 指标 – IOPS 指标的预期值取决于磁盘规格和服务器配置，因此，请使用您的基准来了解典型状况。调查值是否始终与您的基准不同。要获得最佳 IOPS 性能，请确保典型工作集适合内存大小，以最大程度减少读取和写入操作。

对于与性能指标有关的问题，改善性能的第一步是优化最常使用和成本最高昂的查询。对它们进行优化，看看这样做能否减轻系统资源承受的压力。有关更多信息，请参阅 优化查询。

如果您的查询经优化后，问题仍然存在，请考虑升级您的 Amazon RDS 数据库实例类。您可以将其升级为具有更多与问题相关的资源（CPU、RAM、磁盘空间、网络带宽，I/O 容量）的实例类。

优化查询

提高数据库实例性能的最佳方式之一是优化最常使用和占用资源最多的查询。在这里，您可以优化它们以降低它们的运行成本。有关改进查询的信息，请使用以下资源：

- MySQL – 请参阅 MySQL 文档中的优化 SELECT 语句。有关其他查询优化资源，请参阅 MySQL 性能调整和优化资源。
- Oracle – 请参阅 Oracle 数据库文档中的数据库 SQL 优化指南。
- SQL Server – 请参阅 Microsoft 文档中的分析查询。您还可以使用与执行、索引和 I/O 相关的数据管理视图 (DMV)（在 Microsoft 文档的系统动态管理视图和功能中进行了介绍）以排查 SQL Server 查询问题。

  查询优化的一个常见途径是创建有效的索引。有关数据库实例的潜在索引改进，请参阅 Microsoft 文档中的 Database Engine Tuning Advisor。有关在 RDS for SQL Server 上使用 Tuning Advisor 的信息，请参阅 使用 Database Engine Tuning Advisor 分析 Amazon RDS for SQL Server 数据库实例上的数据库工作负载。
- PostgreSQL – 请参阅 PostgreSQL 文档中的使用 EXPLAIN 以了解如何分析查询计划。您可以参考此信息修改查询或底层表以提高查询性能。

  有关如何在查询中指定联接以获得最佳性能的信息，请参阅使用显式 JOIN 子句控制计划程序。
- MariaDB – 请参阅 MariaDB 文档中的查询优化。

使用 MySQL 的最佳实践

MySQL 数据库中的表大小和表的数量都会影响性能。

表大小

通常，操作系统对文件大小的约束决定了 MySQL 数据库的有效最大表大小。因此，限制通常不是由内部 MySQL 约束决定的。

在 MySQL 数据库实例中，请避免数据库中的表增长得过大。虽然一般存储限制为 64 TiB，但预置的存储限制将 MySQL 表文件的最大大小限制为 16 TiB。请对大型表进行分区，以将文件大小限制在 16 TiB 以内。此方法还可以提高性能并缩短恢复时间。有关更多信息，请参阅“Amazon RDS 中的 MySQL 文件大小限制”。

大小非常大的表（大于 100 GB）可能会对读取和写入（包括 DML 语句，尤其是 DDL 语句）的性能有负面影响。大型表上的索引可以显著提高选择性能，但也可能降低 DML 语句的性能。DDL 语句（例如

ALTER TABLE）在处理大型表时性能会显著降低，因为在某些情况下，这些操作可能会完全重建表。这些 DDL 语句可能会在操作期间锁定表。

MySQL 进行读取和写入操作所需的内存量取决于操作中涉及的表。最佳做法是至少有足够的 RAM 来保存主动使用的表的索引。要查找数据库中十个最大的表和索引，请使用以下查询：

```
select table_schema, TABLE_NAME, dat, idx from
(SELECT table_schema, TABLE_NAME,
       ( data_length ) / 1024 / 1024 as dat,
       ( index_length ) / 1024 / 1024 as idx
FROM information_schema.TABLES
order by 3 desc ) a
order by 3 desc
limit 10;
```

表的数量

底层文件系统可能对表示表的文件的数量有限制。但是，MySQL 对表的数量没有限制。尽管如此，MySQL InnoDB 存储引擎中的表总数还是可能导致性能下降，无论这些表的大小如何。要限制操作系统的影响，您可以将表拆分到同一 MySQL 数据库实例中的多个数据库。这样做可能会限制目录中文件的数量，但不能解决整体问题。

当由于表的数量过多（超过 1 万个）而出现性能下降时，这种性能下降是由 MySQL 处理存储文件（包括打开和关闭存储文件）造成的。要解决此问题，可以提高 table_open_cache 和 table_definition_cache 参数的大小。但是，提高这些参数的值可能会显著增加 MySQL 使用的内存量，甚至可能会占用所有可用内存。有关更多信息，请参阅 MySQL 文档中的 MySQL 如何打开和关闭表。

此外，表的数量过多会显著影响 MySQL 的启动时间。干净关机和重启以及崩溃恢复都可能受到影响，尤其是在 MySQL 8.0 之前的版本中。

我们建议数据库实例的所有数据库中表的总数少于 10,000。有关 MySQL 数据库中大量表的使用案例，请参阅 MySQL 8.0 中的一百万个表。

存储引擎

Amazon RDS for MySQL 时间点还原和快照还原功能要求使用崩溃恢复存储引擎。仅 InnoDB 存储引擎支持这些功能。尽管 MySQL 支持功能不同的多种存储引擎，但并非所有引擎都为崩溃恢复和数据持久性而进行了优化。例如，MyISAM 存储引擎不支持可靠的崩溃恢复，并且可能使时间点还原或快照还原无法按预期工作。这可能导致在崩溃后重新启动 MySQL 时丢失或损坏数据。

InnoDB 是 Amazon RDS 上的 MySQL 数据库实例的推荐和支持的存储引擎。InnoDB 实例还可迁移到 Aurora，而 MyISAM 实例无法迁移。但是，如果您需要高强度的全文搜索功能，MyISAM 的效果比 InnoDB 更好。如果您仍然选择对 Amazon RDS 使用 MyISAM，遵循 使用不支持的 MySQL 存储引擎进行自动备份 中概述的步骤可能在某些情况下对执行快照恢复功能会有所帮助。

如果要将现有 MyISAM 表转换为 InnoDB 表，可以使用 MySQL 文档的 Converting Tables from MyISAM to InnoDB 中概述的过程。MyISAM 和 InnoDB 各有优缺点，所以请在执行转换前充分评估转换可能对您的应用程序造成的影响。

此外，Amazon RDS for MySQL 当前不支持联合存储引擎。

使用 MariaDB 的最佳实践

MariaDB 数据库中的表大小和表的数量都会影响性能。

表大小

通常，操作系统对文件大小的约束决定了 MariaDB 数据库的有效最大表大小。因此，限制通常不是由内部 MariaDB 约束决定的。

在 MariaDB 数据库实例中，请避免数据库中的表增长得过大。虽然一般存储限制为 64 TiB，但预置的存储限制将 MariaDB 表文件的最大大小限制为 16 TiB。请对大型表进行分区，以将文件大小限制在 16 TiB 以内。此方法还可以提高性能并缩短恢复时间。

大小非常大的表（大于 100 GB）可能会对读取和写入（包括 DML 语句，尤其是 DDL 语句）的性能有负面影响。大型表上的索引可以显著提高选择性能，但也可能降低 DML 语句的性能。DDL 语句（例如 `ALTER TABLE`）在处理大型表时性能会显著降低，因为在某些情况下，这些操作可能会完全重建表。这些 DDL 语句可能会在操作期间锁定表。

MariaDB 进行读取和写入操作所需的内存量取决于操作中涉及的表。最佳做法是至少有足够的 RAM 来保存主动使用的表的索引。要查找数据库中十个最大的表和索引，请使用以下查询：

```
select table_schema, TABLE_NAME, dat, idx from
(SELECT table_schema, TABLE_NAME,
       ( data_length ) / 1024 / 1024 as dat,
       ( index_length ) / 1024 / 1024 as idx
FROM information_schema.TABLES
order by 3 desc ) a
order by 3 desc
```

```
limit 10;
```

表的数量

底层文件系统可能对表示表的文件的数量有限制。但是，MariaDB 对表的数量没有限制。尽管如此，MariaDB InnoDB 存储引擎中的表总数还是可能导致性能下降，无论这些表的大小如何。要限制操作系统的影响，您可以将表拆分到同一 MariaDB 数据库实例中的多个数据库。这样做可能会限制目录中文件的数量，但不能解决整体问题。

当由于表的数量较多（超过 10,000）而出现性能下降时，这种性能下降是由 MariaDB 处理存储文件造成的。这项工作包括 MariaDB 打开和关闭存储文件。要解决此问题，可以提高 `table_open_cache` 和 `table_definition_cache` 参数的大小。但是，提高这些参数的值可能会显著增加 MariaDB 使用的内存量。它甚至可能会占用所有可用内存。有关详细信息，请参阅 MariaDB 文档中的优化 table_open_cache 。

此外，表的数量过多会显著影响 MariaDB 的启动时间。干净关机和重启以及崩溃恢复都可能受到影响。我们建议数据库实例中所有数据库中表的总数少于一万。

存储引擎

Amazon RDS for MariaDB 时间点还原和快照还原功能要求使用崩溃恢复存储引擎。尽管 MariaDB 支持功能不同的多种存储引擎，但并非所有引擎都针对崩溃恢复和数据耐久性而进行了优化。例如，尽管 Aria 是 MyISAM 的崩溃安全替代，但它仍可能使时间点还原或快照还原无法按预期工作。这可能导致在崩溃后重新启动 MariaDB 时丢失或损坏数据。InnoDB 是 Amazon RDS 上的 MariaDB 数据库实例的推荐和支持的存储引擎。如果您仍然选择对 Amazon RDS 使用 Aria，遵循使用不支持的 MariaDB 存储引擎进行自动备份中概述的步骤可能在某些情况下对执行快照恢复功能会有所帮助。

如果要将现有 MyISAM 表转换为 InnoDB 表，可以使用 MariaDB 文档的 Converting Tables from MyISAM to InnoDB 中概述的过程。MyISAM 和 InnoDB 各有优缺点，所以请在执行转换前充分评估转换可能对您的应用程序造成的影响。

有关使用 Amazon RDS for Oracle 的最佳实践信息，请参阅在 Amazon Web Services 中运行 Oracle 数据库的最佳实践。

使用 PostgreSQL 的最佳实践

在以下两种重要情况下，您可改善 RDS for PostgreSQL 的性能：一种情况是在将数据加载到数据库实例中时。另一种情况是使用 PostgreSQL autovacuum 功能时。以下各节涵盖了我们针对以下情况建议的一些实例。

有关如何 Amazon RDS 实施其他常见 PostgreSQL DBA 任务的信息，请参阅 Amazon RDS for PostgreSQL 的数据库管理员常见任务。

将数据加载到 PostgreSQL 数据库实例中

在将数据加载到 Amazon RDS for PostgreSQL 数据库实例中时，修改数据库实例设置和数据库参数组值。将它们设置为允许以最高效的方式将数据导入数据库实例中。

对数据库实例设置进行以下修改：

- 禁用数据库实例备份（将 backup_retention 设置为 0）
- 禁用多可用区

修改数据库参数组以包括以下设置。此外，测试参数设置以找到对数据库实例最高效的设置。

- 增大 `maintenance_work_mem` 参数的值。有关 PostgreSQL 资源消耗参数的更多信息，请参阅 PostgreSQL 文档。
- 增加 `max_wal_size` 和 `checkpoint_timeout` 参数的值可减少对预写日志（WAL）进行写入的次数。
- 禁用 `synchronous_commit` 参数
- 禁用 PostgreSQL autovacuum 参数。
- 确保您要导入的所有表都不是未记录的表。在故障转移期间，存储在未记录的表中的数据可能会丢失。有关更多信息，请参阅 CREATE TABLE UNLOGGED。

将 `pg_dump -Fc` (压缩) 或 `pg_restore -j` (并行) 命令与这些设置结合使用。

加载操作完成后，将数据库实例和数据库参数恢复为正常设置。

使用 PostgreSQL Autovacuum 功能

PostgreSQL 数据库的 autovacuum 功能是我们强烈向您推荐的一项功能，您可使用此功能保持 PostgreSQL 数据库实例正常运行。Autovacuum 自动执行 VACUUM 和 ANALYZE 命令。使用 autovacuum 是 PostgreSQL 所需的而不是 Amazon RDS 强制的，并且使用此功能对于获得良好性能是至关重要的。原定设置情况下，为所有新的 Amazon RDS for PostgreSQL 数据库实例启用此功能，并将适当地设置相关的配置参数。

您的数据库管理员需要知道和了解此维护操作。有关 autovacuum 的 PostgreSQL 文档，请参阅Autovacuum 守护程序。

Autovacuum 不是一个“不占用资源”的操作，但它将会在后台运行并尽可能地在让位于用户操作。启用后，autovacuum 将检查是否有包含大量更新的或删除的元组的表。它还可防止因事务 ID 重叠导致丢失非常旧的数据。有关更多信息，请参阅防止事务 ID 重叠故障。

Autovacuum 不应被视为是一种开销较高的操作 (可减少此类操作以获得更佳性能)。相反，如果未运行 autovacuum，则更新和删除速度较快的表的性能将随时间推移而快速降低。

> **Important**
>
> 不运行 autovacuum 可能导致最终需要中断工作来执行侵入性更强的 vacuum 操作。在某些情况下，RDS for PostgreSQL 数据库实例可能由于过于保守地使用 autovacuum 而变得不可用。在这些情况下，PostgreSQL 数据库会关闭以保护自己。此时，Amazon RDS 必须直接在数据库实例上执行单一用户模式完全 vacuum。这种完全 vacuum 可能导致多个小时的中断。因此，强烈建议您不要关闭在原定设置情况下开启的 autovacuum。

autovacuum 参数确定 autovacuum 运行的时间和难度。`autovacuum_vacuum_threshold` 和 `autovacuum_vacuum_scale_factor` 参数确定 autovacuum 运行的时间。`autovacuum_max_workers`、`autovacuum_nap_time`、`autovacuum_cost_limit` 和 `autovacuum_cost_delay` 参数确定 autovacuum 运行的难度。有关 autovacuum、其运行时间和所需参数的更多信息，请参阅 PostgreSQL 文档中的日常清理。

以下查询显示名为 table1 的表中的“不活动”元祖数：

```
SELECT relname, n_dead_tup, last_vacuum, last_autovacuum FROM
pg_catalog.pg_stat_all_tables
WHERE n_dead_tup > 0 and relname = 'table1';
```

查询结果将与下面类似：

```
relname | n_dead_tup | last_vacuum | last_autovacuum
---------+------------+-------------+-----------------
 tasks   |   81430522 |             |
(1 row)
```

使用 SQL Server 的最佳实践

利用 SQL Server 数据库实例实现多可用区部署的最佳实践包括：

- 使用 Amazon RDS 数据库事件来监控故障转移。例如，当数据库实例发生故障转移时，您会收到文本消息或电子邮件通知。有关 Amazon RDS 事件的更多信息，请参阅 使用 Amazon RDS 事件通知。
- 如果您的应用程序缓存了 DNS 值，则将生存时间 (TTL) 设置为小于 30 秒的值。在发生故障转移的情况下，这样设置 TTL 是一个好的做法。在故障转移的情况下，IP 地址可能发生更改，而且缓存值可能不再可用。
- 建议您不要 启用以下模式，因为它们会关闭多可用区所需的事务日志记录：
  - Simple 恢复模式
  - 离线模式
  - 只读模式
- 进行测试以确定数据库实例执行故障转移所需的时长。故障转移时间会因使用的数据库类型、实例类和存储类型而异。如果发生故障转移，还应测试您的应用程序继续运行的能力。
- 要缩短故障转移时间，请执行以下操作：
  - 确保为您的工作负载分配了足够的配置的 IOPS。I/O 不足会导致延长故障转移时间。数据库恢复需要 I/O。
  - 使用小型事务。数据库恢复依赖于事务，因此，如果您可将大型事务分成多个小型事务，则您的故障转移时间将缩短。
- 在故障转移期间，请考虑到延迟可能会有所提升。作为故障转移过程的一部分，Amazon RDS 自动将您的数据复制到新的备用实例。这种复制意味着将新数据提交到两个不同的数据库实例。因此，在备用数据库实例赶上新的主数据库实例的进度之前，可能会出现一些延迟。
- 在所有可用区内部署您的应用程序。如果某个可用区出现故障，您在其他可用区内的应用程序仍将可用。

在使用 SQL Server 的多可用区部署时，请记住，Amazon RDS 会为您实例上的所有 SQL Server 数据库创建副本。如果不希望特定数据库具有辅助副本，请设置不使用这些数据库的多可用区的单独数据库实例。

使用数据库参数组

我们建议，在将参数组更改应用于生产数据库实例前，您应当在测试数据库实例上试验数据库参数组更改。在数据库参数组内不恰当地设置数据库引擎参数可能会产生意外的不利影响，包括性能降低和系统不稳定。修改数据库引擎参数时应始终保持谨慎，并且在修改数据库参数组前要备份数据库实例。

有关备份数据库实例的信息，请参阅备份、还原和导出数据。

自动创建数据库实例的最佳实践

Amazon RDS最佳实践是使用数据库引擎的首选次要版本创建数据库实例。您可以使用 Amazon CLI、Amazon RDS API 或 Amazon CloudFormation 自动创建数据库实例。使用这些方法时，您只能指定主要版本，Amazon RDS 随后将自动创建采用首选次要版本的实例。例如，如果 PostgreSQL 12.5 是首选次要版本，并且您使用 `create-db-instance` 指定版本 12，则数据库实例的版本将使用版本 12.5。

要确定首选次要版本，您可以使用 `describe-db-engine-versions` 选项运行 `--default-only` 命令，如以下示例所示。

```
aws rds describe-db-engine-versions --default-only --engine postgres

{
    "DBEngineVersions": [
        {
            "Engine": "postgres",
            "EngineVersion": "12.5",
            "DBParameterGroupFamily": "postgres12",
            "DBEngineDescription": "PostgreSQL",
            "DBEngineVersionDescription": "PostgreSQL 12.5-R1",
            ...some output truncated...
        }
    ]
}
```

有关以编程方式创建数据库实例的信息，请参阅以下资源：

- 使用 Amazon CLI – create-db-instance
- 使用 Amazon RDS API – CreateDBInstance
- 使用 Amazon CloudFormation – AWS::RDS::DBInstance

以编程方式访问 Amazon RDS

Amazon RDS 为您提供以下工具，以便以编程方式管理 Amazon RDS 资源。

Amazon Command Line Interface (Amazon CLI)

可以在命令行 Shell 中使用 Amazon CLI 来创建和管理 RDS 资源。对于 Amazon RDS 等 Amazon 服务，Amazon CLI 提供对 API 的直接访问。有关 Amazon RDS 命令的语法和示例，请参阅《Amazon CLI 命令参考》中的 rds。

Amazon CloudFormation

使用此 Amazon 基础设施即代码（IaC）工具，可以创建模板来描述您想要的所有 Amazon RDS 资源，而 Amazon CloudFormation 为您预调配和配置这些资源。有关更多信息，请参阅 the section called “使用 Amazon CloudFormation 创建资源”。

Amazon 软件开发工具包（SDK）

Amazon 为许多流行的技术和编程语言提供 SDK。这些 SDK 使您能够更轻松地从应用程序中使用该语言或技术调用 Amazon 服务。有关这些 SDK 的更多信息，请参阅用于在 Amazon 上开发和管理应用程序的工具。

Amazon RDS API

此 API 是 Amazon RDS 的协议级接口。使用此 API 时，必须正确格式化每个 HTTPS 请求，并向每个请求添加有效的数字签名。有关更多信息，请参阅 Amazon RDS API 参考。

Console-to-Code

使用此工具，可以为在 Amazon RDS 控制台中执行的操作生成代码，并在其它工具（例如 Amazon CloudFormation）中使用该代码。有关更多信息，请参阅the section called “Console-to-Code”。

使用 Console-to-Code 为 Amazon RDS 控制台操作生成代码

控制台提供了一条用来创建资源和测试原型的引导式路径。如果您希望大规模创建相同的资源，则需要使用自动化代码。Console-to-Code 是 Amazon Q 开发者版的一项功能，可以帮助您开始使用自动化代码。Console-to-Code 记录控制台操作，包括默认值和您提供的参数。随后，它利用生成式人工智能，以您首选的语言和格式为您选择的操作提供建议的代码。由于控制台工作流程可确保您指定的参数值同时有效，因此，您使用 Console-to-Code 生成的代码具有兼容的参数值。您可以将此代码用作一个起点，然后对其进行自定义，以使它可用于您的特定使用案例的生产。

例如，使用 Console-to-Code，可以记录创建 RDS 数据库实例的过程，并选择以 Amazon CloudFormation JSON 格式生成代码。然后，您可以复制该代码并对其进行自定义，以便在 Amazon CloudFormation 模板中使用。

Console-to-Code 目前能够以下列语言和格式生成基础设施即代码（IaC）：

- CDK Java
- CDK Python
- CDK TypeScript
- CloudFormation JSON
- CloudFormation YAML

有关如何使用 Console-to-Code 的更多信息和说明，请参阅《Amazon Q Developer User Guide》中的 Automating Amazon services with Amazon Q Developer Console-to-Code。

配置 Amazon RDS 数据库实例

本节介绍了如何设置 Amazon RDS 数据库实例。在创建数据库实例之前，请确定将运行该数据库实例的数据库实例类。此外，选择 Amazon 区域以确定运行该数据库实例的位置。接下来，创建数据库实例。

您可以使用选项组和数据库参数组配置数据库实例。

- 通过选项组，可以指定各种名为选项的功能，它们是专为特定的 Amazon RDS 数据库实例提供的。
- 数据库参数组就像是引擎配置值的容器，这些值可应用于一个或多个数据库实例。

可用的选项和参数取决于数据库引擎和数据库引擎版本。您可以在创建数据库实例时指定选项组和数据库参数组。也可以修改数据库实例来指定它们。

主题

创建 Amazon RDS 数据库实例

Amazon RDS 的基本构建块是您在其中创建数据库的数据库实例。在创建数据库实例时，您可以选择其特定于引擎的特征。您还可以选择运行数据库服务器的 Amazon 实例的存储容量、CPU、内存等。

主题

- 数据库实例先决条件
- 创建数据库实例
- 数据库实例的设置

数据库实例先决条件

> **Important**
>
> 在创建 Amazon RDS 数据库实例之前，请先完成设置 Amazon RDS 环境中的任务。

以下是创建 RDS 数据库实例的先决条件。

主题

- 为数据库实例配置网络
- 其他先决条件

为数据库实例配置网络

您只能在虚拟私有云（VPC）中基于 Amazon VPC 服务创建 Amazon RDS 数据库实例。此外，它必须位于至少有两个可用区的 Amazon Web Services 区域中。为数据库实例选择的数据库子网组必须涵盖至少两个可用区。此配置可确保您在创建数据库实例时可以配置多可用区部署，或者将来可以轻松迁移到多可用区部署。

要在同一 VPC 中的新数据库实例和 Amazon EC2 实例之间设置连接，可以在创建数据库实例时执行此操作。要从同一 VPC 中的 EC2 实例以外的资源连接到数据库实例，请手动配置网络连接。

主题

- 配置与 EC2 实例的自动网络连接
- 手动配置网络

配置与 EC2 实例的自动网络连接

创建 RDS 数据库集群时，可以使用 Amazon Web Services Management Console在 EC2 实例和新数据库实例之间设置连接。当您这样做时，RDS 会自动配置您的 VPC 和网络设置。数据库实例与 EC2 实例在同一 VPC 中创建，以便 EC2 实例可以访问该数据库实例。

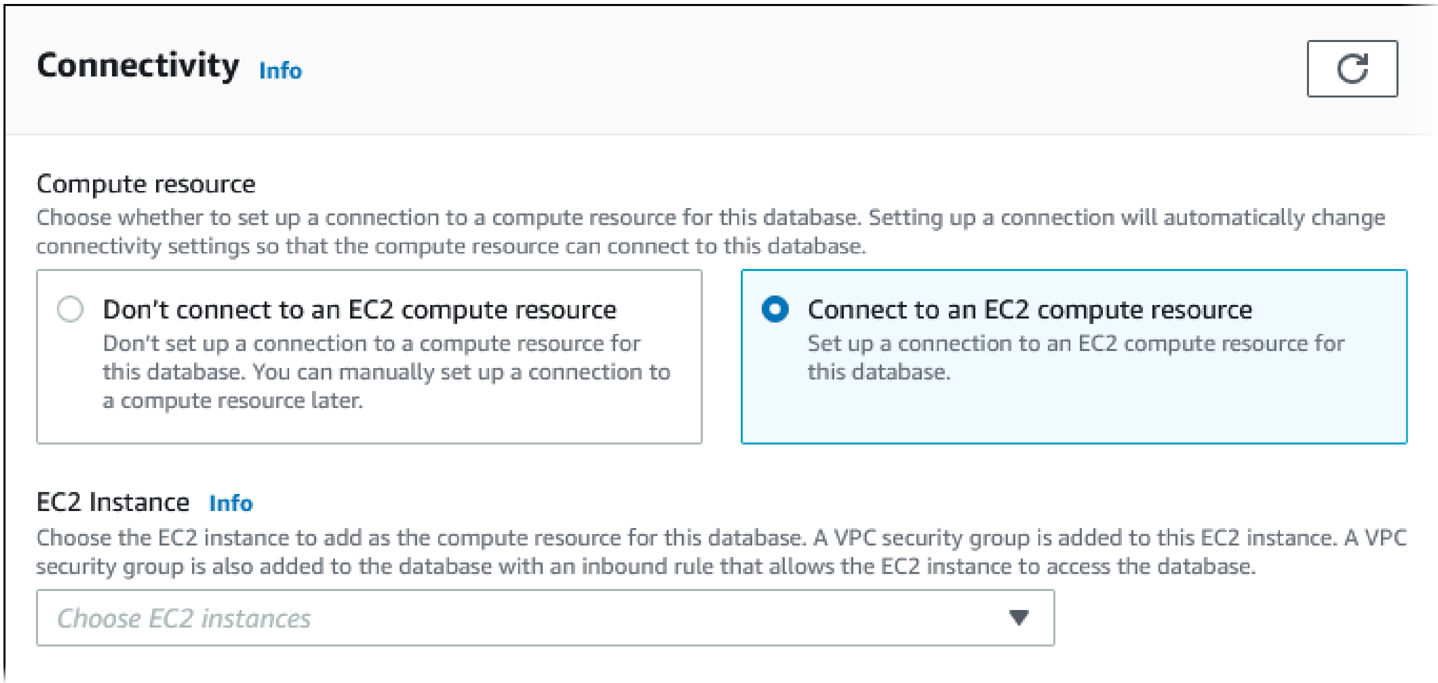
以下是将 EC2 实例与数据库实例连接的要求：

- 在创建数据库实例之前，EC2 实例必须存在于 Amazon Web Services 区域中。

  如果 Amazon Web Services 区域中不存在任何 EC2 实例，控制台将提供创建一个此类实例的链接。
- 创建数据库实例的用户必须具有执行以下操作的权限：
  - `ec2:AssociateRouteTable`
  - `ec2:AuthorizeSecurityGroupEgress`
  - `ec2:AuthorizeSecurityGroupIngress`
  - `ec2:CreateRouteTable`
  - `ec2:CreateSubnet`
  - `ec2:CreateSecurityGroup`
  - `ec2:DescribeInstances`
  - `ec2:DescribeNetworkInterfaces`
  - `ec2:DescribeRouteTables`
  - `ec2:DescribeSecurityGroups`
  - `ec2:DescribeSubnets`
  - `ec2:ModifyNetworkInterfaceAttribute`
  - `ec2:RevokeSecurityGroupEgress`

使用此选项创建私有数据库实例。数据库实例使用仅包含私有子网的数据库子网组，来限制对 VPC 内资源的访问。

要将 EC2 实例连接到数据库实例，请在 Create database（创建数据库）页面上的 Connectivity（连接）部分中，选择 Connect to an EC2 compute resource（连接到 EC2 计算资源）。



当您选择 Connect to an EC2 compute resource（连接到 EC2 计算资源）时，RDS 会自动设置以下选项。除非您通过选择 Don't connect to an EC2 compute resource（不要连接到 EC2 计算资源）来选择不与 EC2 实例建立连接，否则您无法更改这些设置。

控制台选项	自动设置
网络类型	RDS 将网络类型设置为 IPv4。当前，在 EC2 实例和数据库实例之间设置连接时，不支持双堆栈模式。
Virtual Private Cloud (VPC)	RDS 将 VPC 设置为与 EC2 实例关联的 VPC。
DB subnet group（数据库子网组）	RDS 要求在与 EC2 实例相同的可用区中具有带私有子网的数据库子网组。如果存在符合此要求的数据库子网组，则 RDS 将使用现有的数据库子网组。默认情况下，此选项设置为 Automatic setup（自动设置）。 当您选择 Automatic setup（自动设置）但没有满足此要求的数据库子网组时，将执行以下操作。RDS 在三个可用区中使用三个可用的私有子网，其中一个可用区与 EC2 实例相同。如果私有子网在可用区中不可用，则 RDS 会在可用区中创建私有子网。然后，RDS 创建数据库子网组。

控制台选项	自动设置
	当私有子网可用时，RDS 使用与该子网关联的路由表，并将它创建的任何子网添加到该路由表中。当没有可用的私有子网时，RDS 会创建一个没有互联网网关访问权限的路由表，并将它创建的子网添加到该路由表中。 RDS 还允许您使用现有的数据库子网组。如果您想使用您选择的现有数据库子网组，请选择 Choose existing（选择现有）。
公有访问权限	RDS 选择 No（否），以使数据库实例不可供公开访问。 出于安全考虑，最好实践是保持数据库为私有，并确保不能从互联网访问数据库。
VPC security group (firewall) [VPC 安全组（防火墙）]	RDS 创建一个与数据库实例关联的新安全组。安全组命名为 `rds-ec2-`*n*，其中 *n* 是一个数值。该安全组包含一条以 EC2 VPC 安全组（防火墙）作为源的入站规则。这个与数据库实例关联的安全组允许 EC2 实例访问数据库实例。 RDS 还会创建一个与 EC2 实例关联的新安全组。安全组命名为 `ec2-rds-`*n*，其中 *n* 是一个数值。该安全组包含一条以数据库实例的 VPC 安全组作为源的出站规则。该安全组允许 EC2 实例向数据库实例发送流量。 您可以通过选择 Create new（新建）并键入新安全组的名称，添加另一个新安全组。 您可以通过选择 Choose existing（选择现有）并选择要添加的安全组，添加现有安全组。

控制台选项	自动设置
可用区	当您在 Availability & durability（可用性和持久性）（单可用区部署）中选择 Single DB instance（单数据库实例）时，RDS 将选择 EC2 实例的可用区。 当您在 Availability & durability（可用性和耐久性）（多可用区数据库实例部署）中选择 Multi-AZ DB instance（多可用区数据库实例）时，RDS 将为部署中的一个数据库实例选择 EC2 实例的可用区。RDS 为另一个数据库实例随机选择不同的可用区。在 EC2 实例所在的同一个可用区中创建主数据库实例或备用副本。当您选择 Multi-AZ DB instance（多可用区数据库实例）时，如果数据库实例和 EC2 实例位于不同的可用区，则可能会产生跨可用区成本。

有关这些设置的更多信息，请参阅 数据库实例的设置。

如果您在创建数据库实例后更改这些设置，则这些更改可能会影响 EC2 实例与数据库实例之间的连接。

手动配置网络

要从同一 VPC 中的 EC2 实例以外的资源连接到数据库实例，请手动配置网络连接。如果您使用 Amazon Web Services Management Console创建数据库实例，您可以让 Amazon RDS 自动为您创建 VPC。或者，您也可以使用现有 VPC 或为数据库实例创建新的 VPC。无论采用哪种方法，您的 VPC 都要求在至少两个可用区的每个可用区中至少有一个子网，以与 RDS 数据库实例一起使用。

原定设置情况下，Amazon RDS 会在可用区中自动为您创建数据库实例。要选择特定可用区，需要将 Availability & durability（可用性和耐用性）设置更改为 Single DB instance（单数据库实例）。这样做会公开 Availability Zone（可用区）设置，此设置可让您从 VPC 内的可用区中进行选择。但是，如果您选择多可用区部署，则 RDS 会自动选择主数据库实例或写入器数据库实例的可用区，而不显示 Availability Zone（可用区）设置。

在某些情况下，您可能没有原定设置 VPC 或尚未创建 VPC。在这些情况下，您可以在使用控制台创建数据库实例时，让 Amazon RDS 自动为您创建 VPC。否则请执行以下操作：

- 在要部署数据库实例的 Amazon Web Services 区域中，创建在最少两个可用区中均至少有一个子网的 VPC。有关更多信息，请参阅在 VPC 中使用数据库实例和教程：创建 VPC 以用于数据库实例（仅限 IPv4）。
- 指定授权与您的数据库实例的连接的 VPC 安全组。有关更多信息，请参阅通过创建安全组提供对 VPC 中的数据库实例的访问和使用安全组控制访问权限。
- 指定 RDS 数据库子网组，该子网组在 VPC 中定义至少两个可由数据库实例使用的子网。有关更多信息，请参阅 使用数据库子网组。

如果您想连接到与数据库实例不在同一 VPC 中的资源，请参阅在 VPC 中访问数据库实例的场景中的相应方案。

其他先决条件

在创建数据库实例之前，请考虑以下附加先决条件：

- 如果使用 Amazon Identity and Access Management（IAM）凭证连接到 Amazon，您的 Amazon 账户必须具有特定的 IAM policy。这些策略授予执行 Amazon RDS 操作所需的权限。有关更多信息，请参阅 Amazon RDS 的 Identity and Access Management。

  要使用 IAM 访问 RDS 控制台，请使用您的 IAM 用户凭证登录 Amazon Web Services Management Console。然后通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
- 要定制数据库实例的配置参数，请通过所需的参数设置指定数据库参数组。有关创建或修改数据库参数组的信息，请参阅Amazon RDS 的参数组。

> **Important**
>
> 如果您对 Amazon RDS for Db2 使用 BYOL 模型，则在创建数据库实例之前，必须先创建一个包含您的 IBM Site ID 和 IBM Customer ID 的自定义参数组。有关更多信息，请参阅 自带 Db2 许可。

- 确定要为数据库实例指定的 TCP/IP 端口号。有些公司的防火墙阻止连接到 RDS 数据库实例的原定设置端口。如果您公司的防火墙阻止原定设置端口，请为数据库实例选择其他端口。Amazon RDS 数据库引擎的原定设置端口为：

RDS for Db2	RDS for MariaDB	RDS for MySQL	RDS for Oracle	RDS for PostgreSQL	RDS for SQL Server
50000	3306	3306	1521	5432	1433

对于 RDS for SQL Server，预留以下端口，您不能在创建数据库实例时使用它们：1234, 1434, 3260, 3343, 3389, 47001, 和 49152-49156。

创建数据库实例

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 创建 Amazon RDS 数据库实例。

> **Note**
> 对于 RDS for Db2，建议您在创建 RDS for Db2 数据库实例之前设置许可模式所需的项。有关更多信息，请参阅 Amazon RDS for Db2 许可选项。

控制台

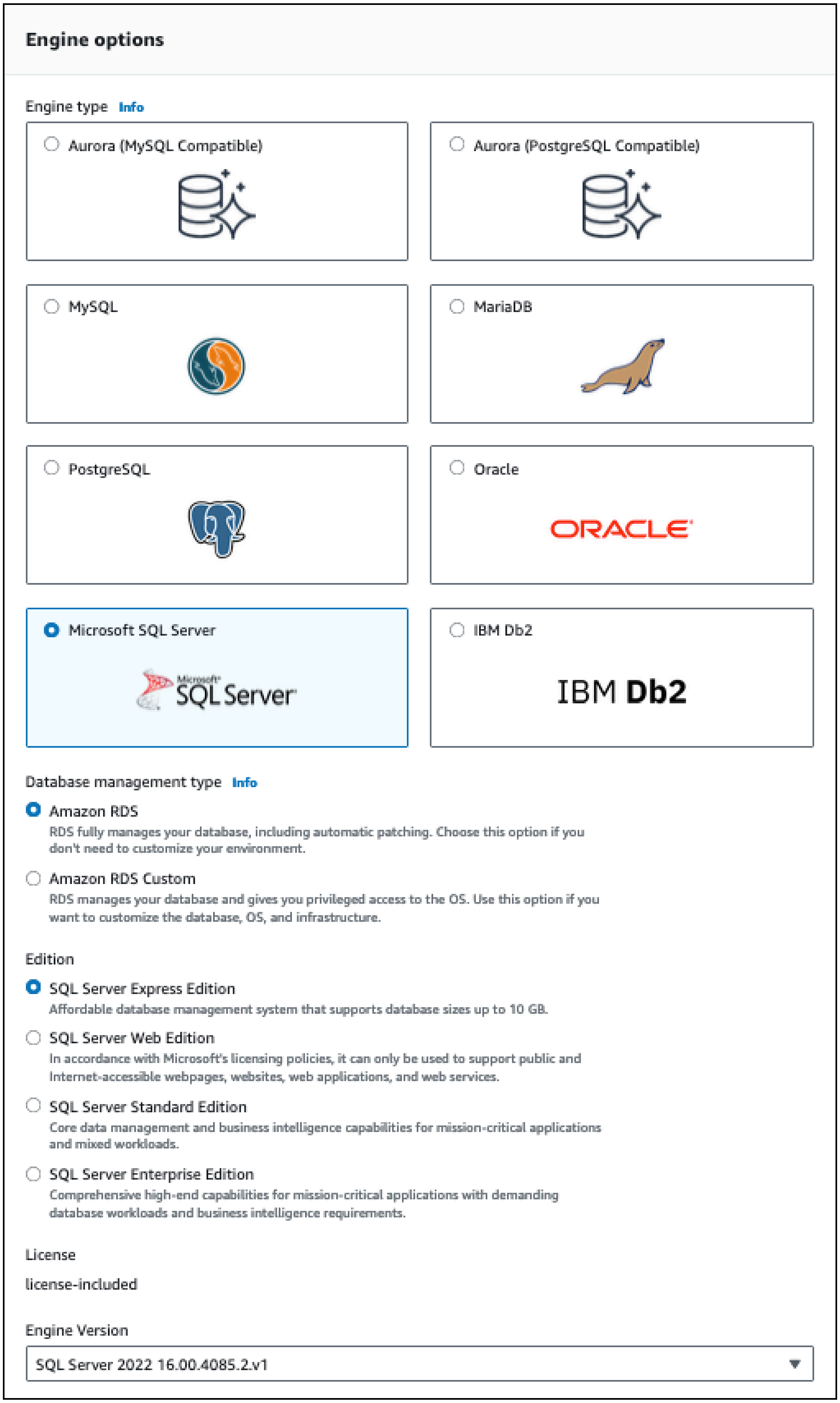
您可以在启用或未启用 Easy create (轻松创建) 的情况下使用 Amazon Web Services Management Console 创建数据库实例。启用 Easy create (轻松创建) 的情况下，您可以仅指定数据库引擎类型、数据库实例大小和数据库实例标识符。Easy create (轻松创建) 为其他配置选项使用默认设置。未启用 Easy create (轻松创建) 的情况下，您在创建数据库时需要指定更多配置选项，包括用于可用性、安全性、备份和维护的选项。

> **Note**
> 在以下过程中，启用了 Standard create (标准创建)，并且未启用 Easy create (轻松创建)。此过程使用 Microsoft SQL Server 作为示例。
> 有关使用 Easy create (轻松创建) 引导您完成为每个引擎创建和连接示例数据库实例的例子，请参阅开始使用 Amazon RDS。

创建数据库实例

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在 Amazon RDS 控制台的右上角，选择要在其中创建数据库实例的Amazon区域。
3. 在导航窗格中，选择 Databases (数据库)。
4. 选择创建数据库，然后选择标准创建。
5. 在引擎选项中，选择 IBM Db2、MariaDB、Microsoft SQL Server、MySQL、Oracle 或 PostgreSQL。

   此处显示了 Microsoft SQL Server。


Engine options
Engine type Info
Aurora (MySQL Compatible)
Aurora (PostgreSQL Compatible)
MySQL
MariaDB
PostgreSQL
Oracle
ORACLE
Microsoft SQL Server
Microsoft SQL Server
IBM Db2
IBM Db2
Database management type Info
Amazon RDS
RDS fully manages your database, including automatic patching. Choose this option if you don't need to customize your environment.
Amazon RDS Custom
RDS manages your database and gives you privileged access to the OS. Use this option if you want to customize the database, OS, and infrastructure.
Edition
SQL Server Express Edition
Affordable database management system that supports database sizes up to 10 GB.
SQL Server Web Edition
In accordance with Microsoft's licensing policies, it can only be used to support public and Internet-accessible webpages, websites, web applications, and web services.
SQL Server Standard Edition
Core data management and business intelligence capabilities for mission-critical applications and mixed workloads.
SQL Server Enterprise Edition
Comprehensive high-end capabilities for mission-critical applications with demanding database workloads and business intelligence requirements.
License
license-included
Engine Version
SQL Server 2022 16.00.4085.2.v1

6. 对于数据库管理类型，如果您使用的是 Oracle 或 SQL Server，请选择 Amazon RDS 或 Amazon RDS Custom。

   此处显示了 Amazon RDS。有关 RDS Custom 的更多信息，请参阅 Amazon RDS Custom。
7. 对于版本，如果您使用的是 Db2、Oracle 或 SQL Server，请选择要使用的数据库引擎版本。

   MySQL 针对该版本只有一个选项，而 MariaDB 和 PostgreSQL 没有选项。
8. 对于 Version (版本)，选择引擎版本。
9. 在 Templates (模板) 中，选择与您的使用案例匹配的模板。如果您选择生产，则将在后面的步骤中预选以下内容：

   - 多可用区故障转移选项
   - Provisioned IOPS SSD (io1) [预调配 IOPS SSD（io1）] 存储选项
   - 启用删除保护选项

   我们建议将这些功能用于任何生产环境。

   > **Note**
   > 模板选择因版本而异。
10. 在设置部分中，打开凭证设置。然后执行以下操作：

    a. （可选）更改 Master username（主用户名）的值。
    b. 选择以下任一其它凭证管理选项：

       - 在 Amazon Secrets Manager 管理

         在选择加密密钥中，选择 Secrets Manager 创建的 KMS 密钥或您已创建的密钥。

         > **Note**
         > 我们建议将 Amazon Secrets Manager 作为管理凭证的最安全的方法。将收取额外费用。有关更多信息，请参阅 使用 Amazon RDS 和 Amazon Secrets Manager 管理密码。
       - 自行管理

要指定密码，请清除自动生成密码复选框（如果已选中）。在主密码和确认主密码中输入相同的密码。

11. （可选）为该数据库实例设置与计算资源的连接。

    在创建数据库实例期间，您可以配置 Amazon EC2 实例和新数据库实例之间的连接。有关更多信息，请参阅 配置与 EC2 实例的自动网络连接。

12. 在连接部分的 VPC 安全组（防火墙）下，如果您选择新建，则会创建一个 VPC 安全组，其入站规则允许本地计算机的 IP 地址访问该数据库。

13. 对于其余部分，请指定数据库实例设置。有关每项设置的信息，请参阅 数据库实例的设置。

14. 选择创建数据库。

    如果选择使用自动生成的密码，则数据库页面上将显示查看凭证详细信息按钮。

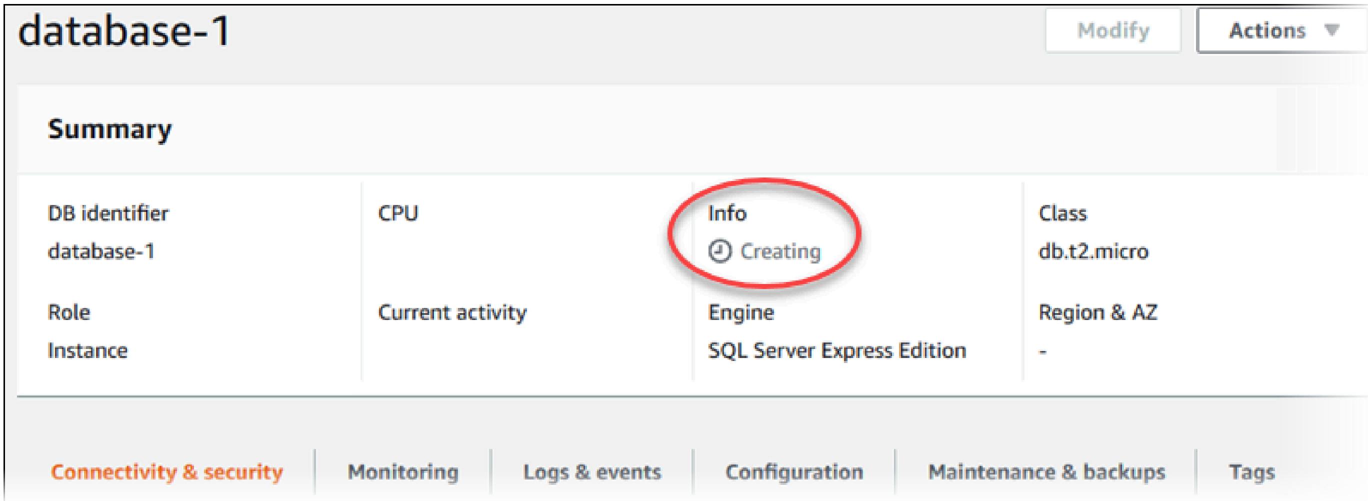
    要查看数据库实例的主用户名和密码，请选择查看凭证详细信息。

    要以主用户身份连接到数据库实例，请使用显示的用户名和密码。

    > **Important**
    >
    > 您无法再次查看主用户密码。如果您不记录它，您可能需要更改它。如果需要在数据库实例可用后更改主用户密码，则可以修改数据库实例以执行此操作。有关修改 数据库实例的更多信息，请参阅 修改 Amazon RDS 数据库实例。

15. 对于 Databases (数据库)，选择新数据库实例的名称。

    在 RDS 控制台上，将显示新数据库实例的详细信息。数据库实例具有 Creating (正在创建) 状态，直到该数据库实例完成创建并可供使用。当状态变为 Available (可用) 时，您可以连接到该数据库实例。根据所分配的数据库实例类和存储的不同，新实例可能需要数分钟时间才能变得可用。



Amazon CLI

> **Note**
>
> 如果您想采用“通过 Amazon Web Services Marketplace 获得 Db2 许可证”模式，则必须先使用 Amazon Web Services Management Console订阅 Amazon Web Services Marketplace 并注册 IBM。有关更多信息，请参阅 订阅 Db2 Marketplace 清单并注册 IBM。

要使用 Amazon CLI 创建数据库实例，请使用以下参数调用 create-db-instance 命令：

- `--db-instance-identifier`
- `--db-instance-class`
- `--vpc-security-group-ids`
- `--db-subnet-group`
- `--engine`
- `--master-username`
- `--master-user-password` 或 `--manage-master-user-password`
- `--allocated-storage`
- `--backup-retention-period`

有关每项设置的信息，请参阅 数据库实例的设置。

此示例使用 Microsoft SQL Server。

Example

对于 Linux、macOS 或 Unix：

```
aws rds create-db-instance \
    --engine sqlserver-se \
    --db-instance-identifier mymsftsqlserver \
    --allocated-storage 250 \
    --db-instance-class db.t3.large \
    --vpc-security-group-ids mysecuritygroup \
    --db-subnet-group mydbsubnetgroup \
    --master-username masterawsuser \
    --manage-master-user-password \
    --backup-retention-period 3
```

对于 Windows：

```
aws rds create-db-instance ^
    --engine sqlserver-se ^
    --db-instance-identifier mydbinstance ^
    --allocated-storage 250 ^
    --db-instance-class db.t3.large ^
    --vpc-security-group-ids mysecuritygroup ^
    --db-subnet-group mydbsubnetgroup ^
    --master-username masterawsuser ^
    --manage-master-user-password ^
    --backup-retention-period 3
```

此命令生成类似于下述信息的输出。

```
DBINSTANCE  mydbinstance  db.t3.large  sqlserver-se  250  sa  creating  3  ****  n
 10.50.2789
SECGROUP  default  active
PARAMGRP  default.sqlserver-se-14  in-sync
```

RDS API

> **Note**
>
> 如果您想采用“通过 Amazon Web Services Marketplace 获得 Db2 许可证”模式，则必须先使用 Amazon Web Services Management Console订阅 Amazon Web Services Marketplace 并注册 IBM。有关更多信息，请参阅 订阅 Db2 Marketplace 清单并注册 IBM。

要使用 Amazon RDS API 创建数据库实例，请调用 CreateDBInstance 操作。

有关每项设置的信息，请参阅 数据库实例的设置。

数据库实例的设置

在下表中可以找到您在创建数据库实例时所选设置的详细信息。该表还显示了支持每项设置的数据库引擎。

您可以使用控制台、create-db-instance CLI 命令或 CreateDBInstance RDS API 操作创建数据库实例。

控制台设置	设置说明	CLI 选项和 RDS API 参数	支持的数据库引擎

分配的存储空间	要为数据库实例分配的存储量（以 GiB 为单位）。有时，为数据库实例分配的存储空间高于数据库大小时可提高 I/O 性能。
	
	有关更多信息，请参阅“Amazon RDS 数据库实例存储”。
		CLI 选项：
		
		`--allocated-storage`
		
		API 参数：
		
		`AllocatedStorage`
			All

架构设置	如果选择使用多租户架构，则 RDS for Oracle 将创建容器数据库（CDB）。如果您不选择此选项，RDS for Oracle 将创建一个非 CDB。非 CDB 使用传统的 Oracle 数据库架构。CDB 可以包含可插拔数据库（PDB），而非 CDB 不能。	CLI 选项： `--engine oracle-ee-cdb`（Oracle 多租户）	Oracle

控制台设置	设置说明	CLI 选项和 RDS API 参数	支持的数据库引擎
	Oracle Database 21c 仅使用 CDB 架构。Oracle Database 19c 可以使用 CDB 或非 CDB 架构。低于 Oracle Database 19c 的版本仅使用非 CDB 架构。 有关更多信息，请参阅 RDS for Oracle CDB 概述。	`--engine oracle-se2-cdb` （Oracle 多租户） `--engine oracle-ee` （传统） `--engine oracle-se2` （传统） API 参数： `Engine`	

控制台设置	设置说明	CLI 选项和 RDS API 参数	支持的数据库引擎
架构配置	只有在架构设置中选择 Oracle 多租户架构时，这些设置才有效。请选择以下任一其它设置： • 使用多租户配置，您的 RDS for Oracle CDB 实例可以包含 1-30 个租户数据库，具体取决于数据库版本和任何所需的选项许可证。在 Oracle 数据库的上下文中，租户数据库是 PDB。不支持应用程序 PDB 和代理 PDB。 您的数据库实例是使用 1 个初始租户数据库创建的。为租户数据库名称、租户数据库主用户名、租户数据库主密码和租户数据库字符集选择值。 多租户配置是永久性的。因此，您无法将多租户配置转换回单租户配置。多租户配置支持的最低版本更新（RU）是 19.0.0.0.ru-2022-01.rur-2022.r1。 **Note** Amazon RDS 配置之所以称为“多-租户”而不是“多租户”，是因为它是 RDS 平台的一项功能，而不仅仅是 Oracle 数据库引擎的功能。同样，RDS 术语“租户”是指 RDS 配置中的任何租户，而不仅仅是 Oracle PDB。在 RDS 文档中，未加连字号的术语“Oracle 多租户”仅指 Oracle 数据库 CDB 架构，该架	CLI 选项： `--multi-tenant`（多租户配置） `--no-multi-tenant`（单租户配置） API 参数： `MultiTenant`	Oracle

控制台设置	设置说明	CLI 选项和 RDS API 参数	支持的数据库引擎
	构与本地部署和 RDS 部署均兼容。 • 使用单租户配置时，您的 RDS for Oracle CDB 包含 1 个 PDB。这是创建 CDB 时的默认配置。您无法删除初始 PDB 或添加更多 PDB。您以后可以将单租户配置的 CDB 转换为多租户配置，但是无法将多租户配置转换回单租户配置。 无论您选择哪种配置，您的 CDB 都包含一个初始 PDB。在多租户配置中，您可以以后使用 RDS API 创建更多 PDB。 有关更多信息，请参阅 RDS for Oracle CDB 概述。		

自动次要版本升级	选择启用自动次要版本升级，以使数据库实例能够在首选次要数据库引擎版本升级可用时自动接收这些升级。这是默认行为。Amazon RDS 在维护时段内执行自动次要版本升级。如果您未选择启用自动次要版本升级，则当新的次要版本可用时，您的数据库实例不会自动升级。
	
	有关更多信息，请参阅 自动升级次要引擎版本。
		CLI 选项：
		
		`--auto-minor-version-upgrade`
		
		`--no-auto-minor-version-upgrade`
		
		API 参数：
		
		`AutoMinorVersionUpgrade`
			全部

控制台设置	设置说明	CLI 选项和 RDS API 参数	支持的数据库引擎

可用区	数据库实例所在的可用区。使用默认值无首选项，除非您要指定一个可用区。
	
	有关更多信息，请参阅“区域、可用区和 Local Zones”。
		CLI 选项：
		
		`--availability-zone`
		
		API 参数：
		
		`AvailabilityZone`
			全部

Amazon KMS key	仅当加密设置为启用加密时可用。选择 Amazon KMS key 以用于加密此数据库实例。有关更多信息，请参阅 加密 Amazon RDS 资源。	CLI 选项：
		
		`--kms-key-id`
		
		API 参数：
		
		`KmsKeyId`
			全部

Amazon License Manager 配置	为 Amazon License Manager 许可证配置输入名称。名称必须不超过 100 个字符，且仅包含 A-Z、A-Z 和 0-9。 有关更多信息，请参阅 the section called “集成 Amazon License Manager”。	CLI 选项： 有关更多信息，请参阅 Amazon License Manager CLI。 API 参数： 有关更多信息，请参阅 Amazon License Manager API。	Db2
备份复制	选择 Enable replication in another Amazon Region（在另一个 Amazon 区域中启用复制），以在其他区域中创建备份用于灾难恢复。 然后为其他备份选择目标区域。	创建数据库实例时不可用。有关使用 Amazon CLI 或 RDS API 启用跨区域备份的信息，请参阅 启用 Amazon RDS 的跨区域自动备份。	Oracle PostgreSQL SQL Server

控制台设置	设置说明	CLI 选项和 RDS API 参数	支持的数据库引擎

备份保留期	您希望保留数据库实例自动备份的天数。对于任何重要的数据库实例，请将该值设置为 **1** 或更大。
	
	有关更多信息，请参阅“备份简介”。
		CLI 选项：
		
		`--backup-retention-period`
		
		API 参数：
		
		`BackupRetentionPeriod`
			全部

Backup target（备份目标）	选择 Amazon Web Services 云 在父级 Amazon 区域中存储自动备份和手动快照。选择 Outposts (on-premises)（Outposts（本地部署））以便将它们存储在您的本地 Outpost 上。
	
	此选项设置仅适用于 Outposts 上的 RDS。有关更多信息，请参阅 创建 Amazon RDS on Amazon Outposts 数据库实例。
		CLI 选项：
		
		`--backup-target`
		
		API 参数：
		
		`BackupTarget`
			MySQL、PostgreSQL、SQL Server

备份时段	Amazon RDS 自动备份数据库实例的时间段。除非您知道需要何时进行数据库备份，否则，请使用无首选项默认值。
	
	有关更多信息，请参阅“备份简介”。
		CLI 选项：
		
		`--preferred-backup-window`
		
		API 参数：
		
		`PreferredBackupWindow`
			全部

控制台设置	设置说明	CLI 选项和 RDS API 参数	支持的数据库引擎

证书颁发机构	数据库实例使用的服务器证书的证书颁发机构（CA）。
	
	有关更多信息，请参阅 使用 SSL/TLS 加密与数据库实例或集群的连接。
		CLI 选项：
		
		`--ca-certificate-identifier`
		
		RDS API 参数：
		
		`CACertificateIdentifier`
			全部

字符集	数据库实例的字符集。数据库字符集的默认值 AL32UTF8 适用于 Unicode 5.0 UTF-8 Universal 字符集。创建数据库实例后，无法更改数据库字符集。
	
	在单租户配置中，非默认数据库字符集仅影响 PDB，不影响 CDB。有关更多信息，请参阅"CDB 架构的单租户配置"。
	
	数据库字符集不同于国家/地区字符集，后者称为 NCHAR 字符集。与数据库字符集不同，NCHAR 字符集指定 NCHAR 数据类型（NCHAR、NVARCHAR2 和 NCLOB）列的编码，而不影响数据库元数据。
	
	有关更多信息，请参阅"RDS for Oracle 字符集"。
		CLI 选项：
		
		`--character-set-name`
		
		API 参数：
		
		`CharacterSetName`
			Oracle

排序规则	数据库实例的服务器级别排序规则。
	
	有关更多信息，请参阅"Microsoft SQL Server 的服务器级别排序规则"。
		CLI 选项：
		
		`--character-set-name`
		
		API 参数：
		
		`CharacterSetName`
			SQL Server

控制台设置	设置说明	CLI 选项和 RDS API 参数	支持的数据库引擎

将标签复制到快照	在创建快照时，该选项将任何数据库实例标签复制到数据库快照中。
	
	有关更多信息，请参阅“为 Amazon RDS 资源添加标签”。
		CLI 选项：
		
		`--copy-tags-to-snapshot`
		
		`--no-copy-tags-to-snapshot`
		
		RDS API 参数：
		
		`CopyTagsToSnapshot`
			All

控制台设置	设置说明	CLI 选项和 RDS API 参数	支持的数据库引擎

数据库身份验证	您想要使用的数据库身份验证选项。
	
	选择 Password authentication 以仅使用数据库密码验证数据库用户的身份。
	
	选择 Password and IAM DB authentication（密码和 IAM 数据库身份验证），以通过用户和角色的数据库密码和用户凭证验证数据库用户的身份。有关更多信息，请参阅 适用于 MariaDB、MySQL 和 PostgreSQL 的 IAM 数据库身份验证。此选项仅支持 MySQL 和 PostgreSQL。
	
	选择 Password and Kerberos authentication (密码和 Kerberos 身份验证) 可使用数据库密码和 Kerberos 身份验证，通过使用 Amazon Managed Microsoft AD 创建的 Amazon Directory Service 来验证数据库用户的身份。接下来，选择目录或选择 Create a new Directory。
	
	有关更多信息，请参阅以下章节之一：
	
	• 将 Kerberos 身份验证用于 Amazon RDS for Db2
	• 对 Amazon RDS for MySQL 进行 Kerberos 身份验证
	• 为 Amazon RDS for Oracle 配置 Kerberos 身份验证
	• 在 Amazon RDS for PostgreSQL 中使用 Kerberos 身份验证
		IAM：
		
		CLI 选项：
		
		`--enable-iam-database-authentication`
		
		`--no-enable-iam-database-authentication`
		
		RDS API 参数：
		
		`EnableIAMDatabaseAuthentication`
		
		Kerberos：
		
		CLI 选项：
		
		`--domain`
		
		`--domain-iam-role-name`
		
		RDS API 参数：
		
		`Domain`
		
		`DomainIAMRoleName`
			因身份验证类型而异

控制台设置	设置说明	CLI 选项和 RDS API 参数	支持的数据库引擎
Database management type（数据库管理类型）	如果您不需要自定义您的环境，请选择 Amazon RDS。 如果您想要自定义数据库、操作系统及基础设施，请选择 Amazon RDS Custom。有关更多信息，请参阅 [Amazon RDS Custom](#)。	对于 CLI 和 API，您可以指定数据库引擎类型。	Oracle SQL Server

数据库端口	要访问数据库实例的端口。显示的是默认端口。
	
	Note
	有些公司的防火墙不允许连接到默认的 MariaDB、MySQL 和 PostgreSQL 端口。如果您的公司防火墙阻止默认端口，请为数据库实例输入另一个端口。
		CLI 选项：
		
		`--port`
		
		RDS API 参数：
		
		`Port`
			全部

数据库引擎版本	要使用的数据库引擎的版本。	CLI 选项：
		
		`--engine-version`
		
		RDS API 参数：
		
		`EngineVersion`
			All

控制台设置	设置说明	CLI 选项和 RDS API 参数	支持的数据库引擎

数据库实例类	数据库实例的配置。例如，db.t3.small 数据库实例类具有 2 GiB 内存、2 个 vCPU、1 个虚拟内核、一个可变 ECU 和中等输入/输出容量。
	
	如有可能，请选择足够大的数据库实例类以便典型查询工作集能够保留在内存中。如果在内存中保留工作集，系统可以避免写入到磁盘，从而提高性能。有关更多信息，请参阅 数据库实例类。
	
	在 RDS for Oracle 中，您可以选择 Include additional memory configurations (包括其他内存配置)。这些配置针对较高的内存与 vCPU 比率进行了优化。例如，db.r5.6xlarge.tpc2.mem4x 是一个 db.r5.8x 数据库实例，每个内核有 2 个线程（tpc2），内存是标准 db.r5.6xlarge 数据库实例的 4 倍。有关更多信息，请参阅 RDS for Oracle 数据库实例类。
		CLI 选项：
		
		`--db-instance-class`
		
		RDS API 参数：
		
		`DBInstanceClass`
			All

数据库实例标识符	数据库实例的名称。请使用与命名本地服务器相同的方式命名数据库实例。数据库实例标识符最多可以包含 63 个字母数字字符，且必须在所选 Amazon 区域中对于您的账户是唯一的。	CLI 选项：
		
		`--db-instance-identifier`
		
		RDS API 参数：
		
		`DBInstanceIdentifier`
			All

控制台设置	设置说明	CLI 选项和 RDS API 参数	支持的数据库引擎

数据库参数组	数据库实例的参数组。您可以选择原定设置参数组，也可以创建自定义参数组。
	
	如果您对于 RDS for Db2 使用 BYOL 模型，则在创建数据库实例之前，必须先创建一个包含您的 IBM Site ID 和 IBM Customer ID 的自定义参数组。有关更多信息，请参阅 自带 Db2 许可。
	
	有关更多信息，请参阅 Amazon RDS 的参数组。
		CLI 选项：
		
		`--db-parameter-group-name`
		
		RDS API 参数：
		
		`DBParameterGroupName`
			全部

DB subnet group（数据库子网组）	要用于数据库集群的数据库子网组。选择 Choose existing（选择现有）以使用现有的数据库子网组。然后，从 Existing DB subnet groups（现有数据库子网组）下拉列表中选择所需的子网组。
	
	选择 Automatic setup（自动设置），让 RDS 选择兼容的数据库子网组。如果不存在任何子网组，RDS 会为您的集群创建一个新的子网组。
	
	有关更多信息，请参阅 使用数据库子网组。
		CLI 选项：
		
		`--db-subnet-group-name`
		
		RDS API 参数：
		
		`DBSubnetGroupName`
			全部

专用日志卷	使用专用日志卷（DLV）将数据库事务日志存储在与包含数据库表的卷不同的存储卷上。
	
	有关更多信息，请参阅 使用专用日志卷（DLV）。
		CLI 选项：
		
		`--dedicated-log-volume`
		
		RDS API 参数：
		
		`DedicatedLogVolume`
			All

控制台设置	设置说明	CLI 选项和 RDS API 参数	支持的数据库引擎

删除保护	启用删除保护以禁止删除数据库实例。如果使用 Amazon Web Services Management Console创建生产数据库实例，将默认启用删除保护。
	
	有关更多信息，请参阅“删除数据库实例”。
		CLI 选项：
		
		`--deletion-protection`
		
		`--no-deletion-protection`
		
		RDS API 参数：
		
		`DeletionProtection`
			All

加密	启用加密可对该数据库实例启用静态加密。
	
	有关更多信息，请参阅“加密 Amazon RDS 资源”。
		CLI 选项：
		
		`--storage-encrypted`
		
		`--no-storage-encrypted`
		
		RDS API 参数：
		
		`StorageEncrypted`
			All

控制台设置	设置说明	CLI 选项和 RDS API 参数	支持的数据库引擎

增强监控	启用增强监控以允许为运行数据库实例的操作系统实时收集指标。
	
	有关更多信息，请参阅“使用增强监控来监控操作系统指标”。
		CLI 选项：
		
		`--monitoring-interval`
		
		`--monitoring-role-arn`
		
		RDS API 参数：
		
		`MonitoringInterval`
		
		`MonitoringRoleArn`
			全部

引擎类型	选择用于此数据库集群的数据库实例。	CLI 选项：
		
		`--engine`
		
		RDS API 参数：
		
		`Engine`
			全部

控制台设置	设置说明	CLI 选项和 RDS API 参数	支持的数据库引擎

Initial database name (初始数据库名称)	数据库实例的数据库名称。如果未提供名称，则 Amazon RDS 不会在数据库实例上创建数据库（Oracle 和 PostgreSQL 除外）。该名称不能是数据库引擎保留的单词，并且具有其他限制，具体取决于数据库引擎。
	
	Db2：
	
	• 必须包含 1–8 个字母数字字符。
	• 它必须以 a-z、A-Z、@、$ 或 # 开头，后跟 a-z、A-Z、0-9、_、@、# 或 $。
	• 它不能包含空格。
	• 有关更多信息，请参阅 额外注意事项。
	
	MariaDB 和 MySQL：
	
	• 必须包含 1–64 个字母数字字符。
	
	Oracle：
	
	• 必须包含 1–8 个字母数字字符。
	• 不能是 `NULL`。默认值为 `ORCL`。
	• 必须以字母开头。
	
	PostgreSQL：
	
	•
		CLI 选项：
		
		`--db-name`
		
		RDS API 参数：
		
		`DBName`
			除了 SQL Server 以外的全部

控制台设置	设置说明	CLI 选项和 RDS API 参数	支持的数据库引擎
	必须包含 1–63 个字母数字字符。 • 必须以字母或下划线开头。后续字符可以是字母、下划线或数字 (0-9)。 • 初始数据库名称为 `postgres`。		

许可证	许可证模型的有效值：
	• 自带许可或市场许可（适用于 Db2）。
	• 通用公共许可证（适用于 MariaDB）。
	• 附带许可证（适用于 Microsoft SQL Server）。
	• 通用公共许可证（适用于 MySQL）。
	• 附带许可证或自带许可（适用于 Oracle）。
	• postgresql 许可证（适用于 PostgreSQL）。
		CLI 选项：
		`--license-model`
		RDS API 参数：
		`LicenseModel`
			全部

日志导出	要发布到 Amazon CloudWatch Logs 的数据库日志文件的类型。
	有关更多信息，请参阅“将数据库日志发布到 Amazon CloudWatch Logs”。
		CLI 选项：
		`--enable-cloudwatch-logs-exports`
		RDS API 参数：
		`EnableCloudwatchLogsExports`
			All

控制台设置	设置说明	CLI 选项和 RDS API 参数	支持的数据库引擎

维护窗口	30 分钟时段，在这段时间内，将会应用对数据库实例的待处理修改。如果该时段无关紧要，请选择无首选项。
	
	有关更多信息，请参阅“Amazon RDS 维护时段”。
		CLI 选项：
		
		`--preferred-maintenance-window`
		
		RDS API 参数：
		
		`PreferredMaintenanceWindow`
			全部

在 Amazon Secrets Manager 中管理主凭证	选择在 Amazon Secrets Manager 中管理主凭证，以在 Secrets Manager 的密钥中管理主用户密码。
	
	（可选）选择用于保护密钥的 KMS 密钥。请从您的账户的 KMS 密钥中进行选择，或输入来自其他账户的密钥。
	
	有关更多信息，请参阅 使用 Amazon RDS 和 Amazon Secrets Manager 管理密码。
		CLI 选项：
		
		`--manage-master-user-password | --no-manage-master-user-password`
		
		`--master-user-secret-kms-key-id`
		
		RDS API 参数：
		
		`ManageMasterUserPassword`
		
		`MasterUserSecretKmsKeyId`
			All

控制台设置	设置说明	CLI 选项和 RDS API 参数	支持的数据库引擎

主密码	您的主用户账户密码。密码具有以下数量的可打印 ASCII 字符（不包括 /、"、空格和 @），具体取决于数据库引擎：
	
	• Db2：8–255
	• Oracle：8–30
	• MariaDB 和 MySQL：8–41
	• SQL Server 和 PostgreSQL：8–128
		CLI 选项：
		
		`--master-user-password`
		
		RDS API 参数：
		
		`MasterUserPassword`
			All

主用户名	用作登录数据库实例的主用户名，拥有所有数据库权限。注意以下命名限制：
	
	• 名称可以包含 1–16 个字母数字字符和下划线。
	• 第一个字符必须是字母。
	• 名称不能是数据库引擎的保留字。
	
	您无法在创建数据库实例之后更改主用户名。
	
	对于 Db2，我们建议您使用与自行管理的 Db2 实例名称相同的主用户名。
	
	有关授予主用户权限的更多信息，请参阅主用户账户权限。
		CLI 选项：
		
		`--master-username`
		
		RDS API 参数：
		
		`MasterUsername`
			All

控制台设置	设置说明	CLI 选项和 RDS API 参数	支持的数据库引擎

Microsoft SQL Server Windows 身份验证	Enable Microsoft SQL Server Windows authentication (启用 Microsoft SQL Server Windows 身份验证)，然后 Browse Directory (浏览目录) 以选择要允许授权域用户使用 Windows 身份验证向此 SQL Server 实例进行身份验证的目录。	CLI 选项：
		`--domain`
		`--domain-iam-role-name`
		RDS API 参数：
		`Domain`
		`DomainIAMRoleName`
			SQL Server

多可用区部署	创建备用实例，以在另一个可用区中创建数据库实例的被动辅助副本，从而提供故障转移支持。建议将多可用区用于生产工作负载以保持高可用性。
	对于开发和测试，您可以选择不创建备用实例。
	有关更多信息，请参阅“配置和管理 Amazon RDS 的多可用区部署”。
		CLI 选项：
		`--multi-az`
		`--no-multi-az`
		RDS API 参数：
		`MultiAZ`
			All

控制台设置	设置说明	CLI 选项和 RDS API 参数	支持的数据库引擎

国家/地区字符集 (NCHAR)	数据库实例的国家/地区字符集，通常称为 NCHAR 字符集。您可以将国家/地区字符集设置为 AL16UTF16（默认值）或 UTF-8。创建数据库实例后，无法更改国家/地区字符集。
	
	国家/地区字符集不同于数据库字符集。与数据库字符集不同，国家/地区字符集仅指定 NCHAR 数据类型（NCHAR、NVARCHAR2 和 NCLOB）列的编码，而不影响数据库元数据。
	
	有关更多信息，请参阅“RDS for Oracle 字符集”。
		CLI 选项：
		
		`--nchar-character-set-name`
		
		API 参数：
		
		`NcharCharacterSetName`
			Oracle

网络类型	数据库实例支持的 IP 寻址协议。
	
	IPv4（默认值），规定资源只能通过 Internet 协议版本 4 (IPv4) 寻址协议与数据库实例通信。
	
	双堆栈模式，规定资源可通过 IPv4 和/或 Internet 协议版本 6 (IPv6) 与数据库实例通信。如果您有任何必须通过 IPv6 寻址协议与数据库实例通信的资源，请使用双堆栈模式。此外，请确保将 IPv6 CIDR 块与指定的数据库子网组中的所有子网进行关联。
	
	有关更多信息，请参阅 Amazon RDS IP 寻址。
		CLI 选项：
		
		`--network-type`
		
		RDS API 参数：
		
		`NetworkType`
			全部

控制台设置	设置说明	CLI 选项和 RDS API 参数	支持的数据库引擎

选项组	数据库实例的选项组。您可以选择默认选项组，也可以创建自定义选项组。
	
	有关更多信息，请参阅“使用选项组”。
		CLI 选项：
		
		`--option-group-name`
		
		RDS API 参数：
		
		`OptionGroupName`
			All

Performance Insights	启用 Performance Insights 以监控数据库实例负载，以便您可以分析数据库性能和解决数据库性能问题。
	
	选择保留期以确定要保留的 Performance Insights 历史记录数量。免费套餐中的保留设置为 Default (7 days) [默认设置（7 天）]。要将性能数据保留更长时间，请指定 1–24 个月。有关保留期的更多信息，请参阅性能详情的定价和数据留存。
	
	选择一个 KMS 密钥以用来保护用于加密该数据库卷的密钥。请从您的账户的 KMS 密钥中进行选择，或输入来自其他账户的密钥。
	
	有关更多信息，请参阅 在 Amazon RDS 上使用性能详情监控数据库负载。
		CLI 选项：
		
		`--enable-performance-insights`
		
		`--no-enable-performance-insights`
		
		`--performance-insights-retention-period`
		
		`--performance-insights-kms-key-id`
		
		RDS API 参数：
		
		`EnablePerformanceInsights`
		
		`PerformanceInsightsRetentionPeriod`
		
		`PerformanceInsightsKMSKeyId`
			全部，但 Db2 除外

控制台设置	设置说明	CLI 选项和 RDS API 参数	支持的数据库引擎

预置 IOPS	数据库实例的预调配 IOPS（每秒 I/O 操作数）值。仅当您为 Storage type（存储类型）选择以下类型之一时，此设置才可用：
	
	• General purpose SSD (gp3) [通用型 SSD（gp3）]
	• Provisioned IOPS SSD (io1) [预调配 IOPS（io1）]
	• 预调配 IOPS SSD（io2）
	
	有关更多信息，请参阅 Amazon RDS 数据库实例存储。
		CLI 选项：
		
		`--iops`
		
		RDS API 参数：
		
		`Iops`
			全部

控制台设置	设置说明	CLI 选项和 RDS API 参数	支持的数据库引擎

公有访问权限	Yes (是) 为数据库实例提供公有 IP 地址，这表示可以在 VPC 外部访问该实例。要可供公开访问，数据库实例还必须在 VPC 的公有子网中。
	
	No 让数据库实例只在 VPC 内部是可访问的。
	
	有关更多信息，请参阅 对互联网隐藏 VPC 中的数据库实例。
	
	要从 VPC 外部连接到数据库实例，该数据库实例必须可公开访问。此外，必须使用数据库实例安全组的入站规则授予访问权限。此外，还必须满足其他要求。有关更多信息，请参阅 无法连接到 Amazon RDS 数据库实例。
	
	如果数据库实例不可供公开访问，则使用 Amazon Site-to-Site VPN 连接或 Amazon Direct Connect 连接从私有网络访问该实例。有关更多信息，请参阅 互联网络流量隐私。
		CLI 选项：
		
		`--publicly-accessible`
		
		`--no-publicly-accessible`
		
		RDS API 参数：
		
		`PubliclyAccessible`
			全部

控制台设置	设置说明	CLI 选项和 RDS API 参数	支持的数据库引擎

RDS Extended Support	选择启用 RDS Extended Support，以允许受支持的主要引擎版本在 RDS 标准支持终止日期后继续运行。
	
	当您创建数据库实例时，Amazon RDS 默认使用 RDS Extended Support。为了防止在 RDS 标准支持终止日期后创建新的数据库实例并避免支付 RDS Extended Support 费用，请禁用此设置。在 RDS Extended Support 定价开始日期之前，您的现有数据库实例不会产生费用。
	
	有关更多信息，请参阅 Amazon RDS 的 Amazon RDS 扩展支持。
		CLI 选项：
		
		`--engine-lifecycle-support`
		
		RDS API 参数：
		
		`EngineLifecycleSupport`
			MySQL
			
			PostgreSQL

RDS Proxy（RDS 代理）	选择 Create an RDS Proxy（创建 RDS 代理），以便为您的数据库实例创建代理。Amazon RDS 会自动为代理创建 IAM 角色和 Secrets Manager 密钥。 有关更多信息，请参阅 的 Amazon RDS 代理。	创建数据库实例时不可用。	MariaDB MySQL PostgreSQL

Storage autoscaling (存储自动扩展)	Enable storage autoscaling (启用存储自动扩展)，使 Amazon RDS 能够在需要时自动增加存储空间，以避免数据库实例耗尽存储空间。
	
	使用最大存储阈值设置 Amazon RDS 自动将数据库实例存储增加到的上限。默认值为 1,000 GiB。
	
	有关更多信息，请参阅"使用 Amazon RDS 存储自动扩展功能自动管理容量"。
		CLI 选项：
		
		`--max-allocated-storage`
		
		RDS API 参数：
		
		`MaxAllocatedStorage`
			全部

控制台设置	设置说明	CLI 选项和 RDS API 参数	支持的数据库引擎

Storage throughput（存储吞吐量）	数据库实例的存储吞吐量值。仅当您为 Storage type（存储类型）选择 General purpose SSD (gp3) [通用型 SSD（gp3）] 时，此设置才可用：
	
	有关更多信息，请参阅 gp3 存储（推荐）。
		CLI 选项：
		
		`--storage-throughput`
		
		RDS API 参数：
		
		`StorageThroughput`
			All

存储类型	数据库实例的存储类型。
	
	如果您选择 General Purpose SSD (gp3) [通用型 SSD（gp3）]，则可以在 Advanced settings（高级设置）下预调配额外的预调配 IOPS 和存储吞吐量。
	
	如果您选择预调配 IOPS SSD（io1）或预调配 IOPS SSD（io2），请输入预调配 IOPS 值。
	
	有关更多信息，请参阅 Amazon RDS 存储类型。
		CLI 选项：
		
		`--storage-type`
		
		RDS API 参数：
		
		`StorageType`
			All

子网组	与此数据库实例关联的数据库子网组。
	
	有关更多信息，请参阅 使用数据库子网组。
		CLI 选项：
		
		`--db-subnet-group-name`
		
		RDS API 参数：
		
		`DBSubnetGroupName`
			全部

控制台设置	设置说明	CLI 选项和 RDS API 参数	支持的数据库引擎

租户数据库名称	在 Oracle 架构的多租户配置中，您的初始 PDB 的名称。只有在架构配置中选择多租户配置时，此设置才可用。
	
	租户数据库的名称必须与名为 `RDSCDB` CDB 的名称不同。您无法更改 CDB 名称。
		CLI 选项：
		
		`--db-name`
		
		RDS API 参数：
		
		`DBName`
			Oracle

租户数据库主用户名	用作登录租户数据库（PDB）的主用户名，拥有所有数据库权限。只有在架构配置中选择多租户配置时，此设置才可用。
	
	注意以下命名限制：
	
	• 名称可以包含 1–16 个字母数字字符和下划线。
	• 第一个字符必须是字母。
	• 名称不能是数据库引擎的保留字。
	
	您无法执行以下操作：
	
	• 创建租户数据库后更改租户主用户名。
	• 使用租户主用户名登录 CDB。
		CLI 选项：
		
		`--master-username`
		
		RDS API 参数：
		
		`MasterUsername`
			Oracle

租户数据库主密码	租户数据库（PDB）的主用户帐户的密码。只有在架构配置中选择多租户配置时，此设置才可用。
	
	密码必须包含 8–30 个可打印的 ASCII 字符（不包括 `/`、`"`、空格和 `@`。）。
		CLI 选项：
		
		`--master-password`
		
		RDS API 参数：
		
		`MasterPassword`
			Oracle

控制台设置	设置说明	CLI 选项和 RDS API 参数	支持的数据库引擎

租户数据库字符集	初始租户数据库的字符集。只有在架构配置中选择多租户配置时，此设置才可用。仅支持 RDS for Oracle CDB 实例。
	
	租户数据库字符集的默认值 AL32UTF8 适用于 Unicode 5.0 UTF-8 Universal 字符集。您可以选择不同于 CDB 字符集的租户数据库字符集。
	
	有关更多信息，请参阅 RDS for Oracle 字符集。
		CLI 选项：
		
		`--character-set-name`
		
		RDS API 参数：
		
		`CharacterSetName`
			Oracle

租户数据库国家字符集	租户数据库的国家字符集，通常称为 NCHAR 字符集。只有在架构配置中选择多租户配置时，此设置才可用。仅支持 RDS for Oracle CDB 实例。
	
	您可以将国家字符集设置为 AL16UTF16（默认值）或 UTF-8。创建租户数据库后，无法更改国家字符集。
	
	租户数据库国家字符集不同于租户数据库字符集。国家字符集仅为使用 NCHAR 数据类型（NCHAR、NVARCHAR2 、和 NCLOB）且不影响数据库元数据的列指定编码。
	
	有关更多信息，请参阅 RDS for Oracle 字符集。
		CLI 选项：
		
		`--nchar-character-set-name`
		
		API 参数：
		
		`NcharCharacterSetName`
			Oracle

控制台设置	设置说明	CLI 选项和 RDS API 参数	支持的数据库引擎

时区	数据库实例的时区。如果您未选择时区，则数据库实例将使用默认时区。您无法在创建数据库实例之后更改时区。
	
	有关更多信息，请参阅Amazon RDS for Db2 数据库实例的本地时区 和Microsoft SQL Server 数据库实例的本地时区。
		CLI 选项：
		
		`--timezone`
		
		RDS API 参数：
		
		`Timezone`
			Db2
			
			SQL Server
			
			RDS Custom for SQL Server

Virtual Private Cloud (VPC)	基于要与此数据库实例关联的 Amazon VPC 服务的 VPC。 有关更多信息，请参阅 Amazon VPC 和 Amazon RDS。	对于 CLI 和 API，您可以指定 VPC 安全组 ID。	全部

VPC security group (firewall) [VPC 安全组（防火墙）]	要与数据库实例关联的安全组。
	
	有关更多信息，请参阅 VPC 安全组概述。
		CLI 选项：
		
		`--vpc-security-group-ids`
		
		RDS API 参数：
		
		`VpcSecurityGroupIds`
			全部

使用 Amazon CloudFormation 创建 Amazon RDS 资源

Amazon RDS 与 Amazon CloudFormation 集成，后者是一项服务，可帮助您对Amazon资源进行建模和设置，这样您就可以花较少的时间来创建和管理资源与基础设施。您可以创建一个模板，描述所需的所有 Amazon 资源（例如数据库实例和数据库参数组），而且 Amazon CloudFormation 会为您预置和配置这些资源。

使用 Amazon CloudFormation 时，可以重复使用您的模板来不断地重复设置您的 RDS 资源。仅描述您的资源一次，然后在多个Amazon账户和区域中反复配置相同的资源。

RDS 和 Amazon CloudFormation 模板

Amazon CloudFormation 模板是 JSON 或 YAML 格式的文本文件。这些模板可描述您要在 Amazon CloudFormation 堆栈中调配的资源。如果您不熟悉 JSON 或 YAML，可以在 Amazon CloudFormation Designer 的帮助下开始使用 Amazon CloudFormation 模板。有关更多信息，请参阅 Amazon CloudFormation 用户指南中的什么是 Amazon CloudFormation Designer？。

RDS 支持在 Amazon CloudFormation 中创建资源。有关更多信息（包括这些资源的 JSON 和 YAML 模板示例），请参阅 Amazon CloudFormation 用户指南中的 RDS 资源类型参考。

了解有关 Amazon CloudFormation 的更多信息

要了解有关 Amazon CloudFormation 的更多信息，请参阅以下资源：

- Amazon CloudFormation
- Amazon CloudFormation 用户指南
- Amazon CloudFormation API Reference
- Amazon CloudFormation 命令行界面用户指南

连接到 Amazon RDS 数据库实例

在连接到数据库实例之前，您必须先创建数据库实例。有关信息，请参阅 创建 Amazon RDS 数据库实例。在 Amazon RDS 预调配数据库实例后，使用标准客户端应用程序或实用程序将数据库引擎连接到数据库实例。在连接字符串中，指定数据库实例端点中的 DNS 地址作为主机参数。此外，还可以将数据库实例端点中的端口号指定为端口参数。

有关查找 Amazon RDS 数据库实例的连接信息或访问 VPC 中数据库实例的场景的更多信息，请参阅以下主题。

- 查找 Amazon RDS 数据库实例的连接信息
- 在 VPC 中访问数据库实例的方案

查找 Amazon RDS 数据库实例的连接信息

数据库实例的连接信息包括其端点、端口和有效数据库用户，如主用户。例如，对于 MySQL 数据库实例，假设端点值是 `mydb.123456789012.us-east-1.rds.amazonaws.com`。在这种情况下，端口值是 `3306`，数据库用户是 `admin`。鉴于此信息，您可以在连接字符串中指定以下值：

- 对于主机或主机名或 DNS 名称，请指定 `mydb.123456789012.us-east-1.rds.amazonaws.com`。
- 对于端口，请指定 `3306`。
- 对于用户，请指定 `admin`。

端点对于每个数据库实例都是唯一的，端口和用户的值可能会有所不同。以下列表显示了每个数据库引擎的最常见端口：

- Db2 – 50000
- MariaDB – 3306
- Microsoft SQL Server – 1433
- MySQL – 3306
- Oracle – 1521
- PostgreSQL – 5432

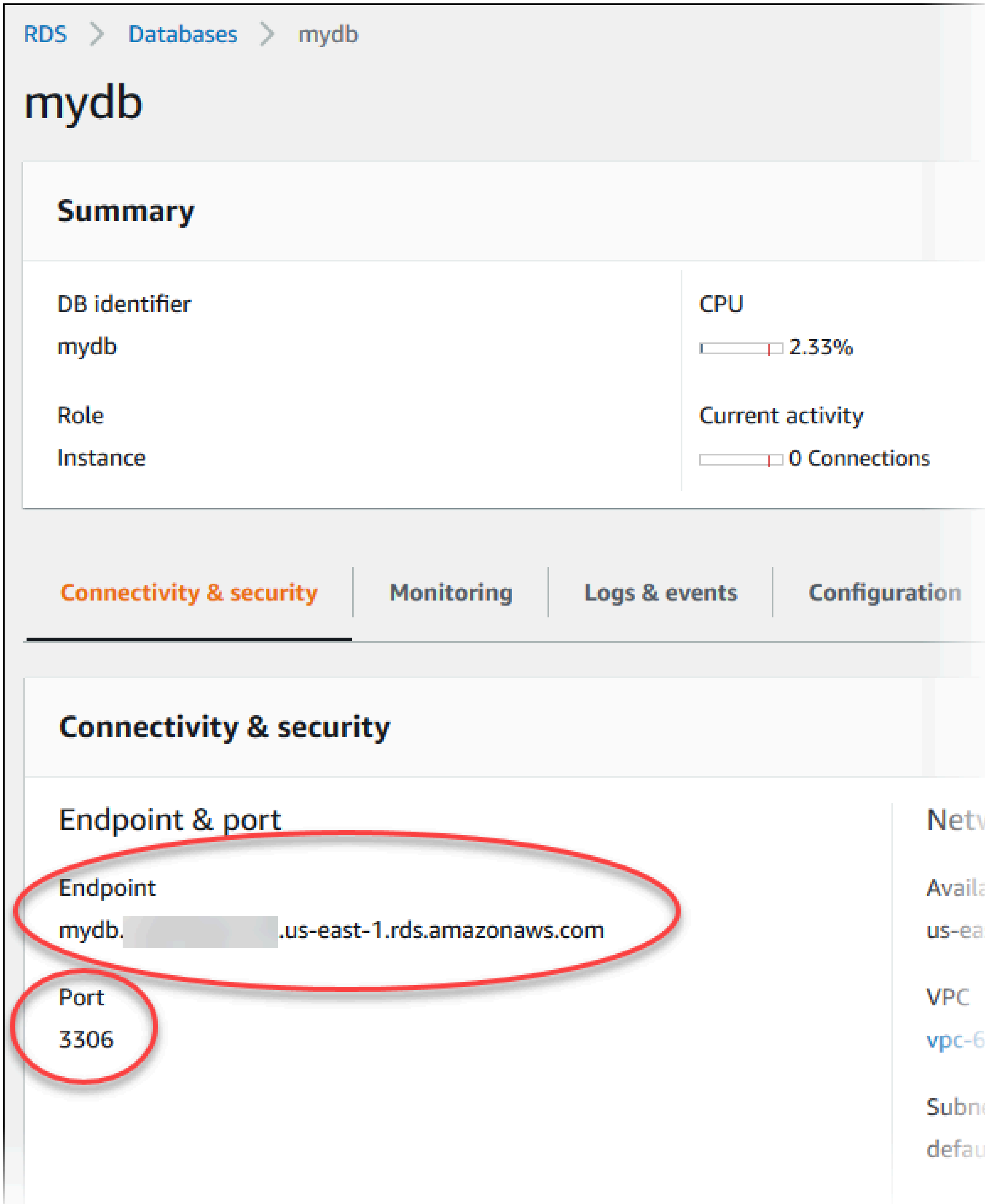
要连接到数据库实例，请使用数据库引擎的任何客户端。例如，您可以使用 mysql 实用程序连接到 MariaDB 或 MySQL 数据库实例。您可以使用 Microsoft SQL Server Management Studio 连接到 SQL Server 数据库实例。您可以使用 Oracle SQL Developer 连接到 Oracle 数据库实例。类似地，可以使用 psql 命令行实用程序连接到 PostgreSQL 数据库实例。

要查找数据库实例的连接信息，请使用 Amazon Web Services Management Console。还可以使用 Amazon Command Line Interface（Amazon CLI）[describe-db-instances](https://docs.aws.amazon.com/cli/latest/reference/rds/describe-db-instances.html) 命令或 RDS API [DescribeDBInstances](https://docs.aws.amazon.com/AmazonRDS/latest/APIReference/API_DescribeDBInstances.html) 操作。

控制台

在 Amazon Web Services Management Console 中查找数据库实例的连接信息

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 数据库 以显示数据库实例的列表。
3. 选择数据库实例的名称以显示其详细信息。
4. 在连接和安全性选项卡上，复制端点。另请注意端口号。您需要端点和端口号才能连接到数据库实例。



5. 如果需要查找主用户名，请选择 Configuration (配置) 选项卡并查看 Master username (主用户名) 值。

Amazon CLI

要使用 Amazon CLI 查找数据库实例的连接信息，请调用 describe-db-instances 命令。在调用中，查询数据库实例 ID、端点、端口和主用户名。

对于 Linux、macOS 或 Unix：

```
aws rds describe-db-instances \
  --query "*[].[DBInstanceIdentifier,Endpoint.Address,Endpoint.Port,MasterUsername]"
```

对于 Windows：

```
aws rds describe-db-instances ^
  --query "*[].[DBInstanceIdentifier,Endpoint.Address,Endpoint.Port,MasterUsername]"
```

您的输出应类似于以下内容。

```
[
    [
        "mydb",
        "mydb.123456789012.us-east-1.rds.amazonaws.com",
        3306,
        "admin"
    ],
    [
        "myoracledb",
        "myoracledb.123456789012.us-east-1.rds.amazonaws.com",
        1521,
        "dbadmin"
    ],
    [
        "mypostgresqldb",
        "mypostgresqldb.123456789012.us-east-1.rds.amazonaws.com",
        5432,
        "postgresadmin"
    ]
]
```

RDS API

要使用 Amazon RDS API 查找数据库实例的连接信息，请调用 DescribeDBInstances 操作。在输出中，找到端点地址、端点端口和主用户名的值。

在 VPC 中访问数据库实例的方案

利用 Amazon Virtual Private Cloud (Amazon VPC)，您可以在 Virtual Private Cloud (VPC) 中启动 Amazon资源，如 Amazon RDS 数据库实例。使用 Amazon VPC 时，您的虚拟联网环境完全由您控制。您可以选择自己的 IP 地址范围、创建子网以及配置路由和访问控制列表。

VPC 安全组控制 VPC 内的数据库实例的访问权限。每个 VPC 安全组规则都允许特定的源访问 VPC 中与此 VPC 安全组关联的数据库实例。源可以是一个地址范围 (例如，203.0.113.0/24) 或另一个 VPC 安全组。指定作为源的 VPC 安全组后，就可以允许从使用此源 VPC 安全组的所有实例 (通常为应用程序服务器) 中传入流量。

在尝试连接到您的数据库实例之前，请为您的使用案例配置 VPC。以下是在 VPC 中访问数据库实例的常见方案：

- VPC 中的数据库实例被同一 VPC 中的 Amazon EC2 实例访问 – 数据库实例在 VPC 中的常见用途是与在同一 VPC 中的 EC2 实例中运行的应用程序服务器共享数据。EC2 实例可能会使用与数据库实例交互的应用程序运行 Web 服务器。
- 不同 VPC 中的 EC2 实例访问 VPC 中的数据库实例 – 在某些情况下，您的数据库实例与用于访问该实例的 EC2 实例位于不同的 VPC 中。如果是这样，您可以使用 VPC 对等连接访问该数据库实例。
- 客户端应用程序通过互联网访问 VPC 中的数据库实例 – 要通过互联网从客户端应用程序访问 VPC 中的数据库实例，您可配置包含一个公有子网的 VPC。还要配置一个互联网网关以实现通过互联网通信。

  要从 VPC 外部连接到数据库实例，该数据库实例必须可公开访问。此外，必须使用数据库实例安全组的入站规则授予访问权限，并且必须满足其他要求。有关更多信息，请参阅 无法连接到 Amazon RDS 数据库实例。
- 私有网络访问的 VPC 中的数据库实例 - 如果您的数据库实例不可公开访问，则可以使用以下选项之一从私有网络访问它：
  - Amazon Site-to-Site VPN 连接
  - Amazon Direct Connect 连接
  - Amazon Client VPN 连接

有关更多信息，请参阅 在 VPC 中访问数据库实例的场景。

使用 Amazon 驱动程序连接到数据库实例

借助 Amazon 驱动程序套件，可显著缩短切换和故障转移时间，并支持使用 Amazon Secrets Manager、Amazon Identity and Access Management（IAM）和联合身份进行身份验证。Amazon 驱动程序依靠监控数据库实例状态和了解实例拓扑，来确定新的主实例。这种方法将切换和故障转移时间缩短到几秒钟，而开源驱动程序的切换和故障转移时间则为几十秒。

下表列出了每个驱动程序支持的功能。随着新服务功能的推出，使用 Amazon 驱动程序套件可为这些服务功能提供内置支持。

功能	Amazon JDBC 驱动程序	Amazon Python 驱动程序	适用于 MySQL 的 Amazon ODBC 驱动程序
故障转移支持	是	是	是
增强了故障转移监控功能	是	是	是
读/写分离	是	是	否
驱动程序元数据连接	是	不适用	不适用
遥测	是	是	否
Secrets Manager	是	是	是
IAM 身份验证	是	是	是
联合身份验证（AD FS）	是	是	否
联合身份（Okta）	是	否	否
多可用区数据库集群	是	是	否

有关 Amazon 驱动程序的更多信息，请参阅 RDS for MariaDB、RDS for MySQL 或 RDS for PostgreSQL 数据库实例的相应语言驱动程序。

> **Note**
>
> RDS for MariaDB 支持的唯一功能是使用 Amazon Secrets Manager、Amazon Identity and Access Management（IAM）和联合身份进行身份验证。

连接到运行特定数据库引擎的数据库实例

要了解如何连接到运行特定数据库引擎的数据库实例，请按照数据库引擎的说明操作：

- RDS for Db2
- RDS for MariaDB
- RDS for SQL Server
- RDS for MySQL
- RDS for Oracle
- RDS for PostgreSQL

使用 RDS Proxy 管理连接

还可以使用 Amazon RDS 代理管理与 RDS for MariaDB、RDS for Microsoft SQL Server、RDS for MySQL 和 RDS for PostgreSQL 数据库实例的连接。RDS Proxy 允许应用程序池化和共享数据库连接，以提高可扩展性。有关更多信息，请参阅 的 Amazon RDS 代理。

数据库身份验证选项

Amazon RDS 支持以下对数据库用户进行身份验证的方法：

- 密码身份验证 – 数据库实例将执行用户账户的所有管理。您可以使用 SQL 语句创建用户并指定密码。您可以使用的 SQL 语句取决于数据库引擎。
- Amazon Identity and Access Management（IAM）数据库身份验证 – 您在连接到数据库实例时无需使用密码。而是使用身份验证令牌。
- Kerberos 身份验证 – 使用 Kerberos 和 Microsoft Active Directory 对数据库用户进行外部身份验证。Kerberos 是一种网络身份验证协议，它使用票证和对称密钥加密，而不再需要通过网络传输密码。Kerberos 已内置到 Active Directory 中，用于在网络资源（如数据库）中对用户进行身份验证。

IAM 数据库身份验证和 Kerberos 身份验证仅适用于特定的数据库引擎和版本。

有关更多信息，请参阅“Amazon RDS 的数据库身份验证”。

加密连接

您可以使用应用程序中的安全套接字层 (SSL) 或传输层安全 (TLS) 来加密与数据库实例的连接。每个数据库引擎都有自己的用于实施 SSL/TLS 的过程。有关更多信息，请参阅 使用 SSL/TLS 加密与数据库实例或集群的连接。

使用选项组

一些数据库引擎提供更易于管理数据和数据库的其他功能，并为您的数据库提供更多安全保护。Amazon RDS 利用选项组启用和配置这些功能。通过选项组，可以指定各种名为选项的功能，它们是专为特定的 Amazon RDS 数据库实例提供的。选项可能具有指定该选项工作方式的设置。当您将数据库实例与选项组相关联时，就会在数据库实例上启用指定的选项和选项设置。

Amazon RDS 支持以下数据库引擎选项：

数据库引擎	相关文档
Db2	RDS for Db2 数据库实例选项
MariaDB	适用于 MariaDB 数据库引擎的选项
Microsoft SQL Server	适用于 Microsoft SQL Server 数据库引擎的选项
MySQL	适用于 MySQL 数据库实例的选项
Oracle	向 Oracle 数据库实例添加选项
PostgreSQL	PostgreSQL 不使用选项和选项组。PostgreSQL 使用扩展和模块来提供其他功能。有关更多信息，请参阅“支持的 PostgreSQL 扩展版本”。

选项组概述

Amazon RDS 为每个新数据库实例提供空的默认选项组。您无法修改或删除此原定设置选项组，但可以修改您创建的通过原定设置选项组派生其设置的任何新选项组。要将选项应用于数据库实例，您必须执行以下操作：

1. 创建新的选项组，或者复制或修改现有选项组。
2. 向选项组中添加一个或多个选项。
3. 将选项组与数据库实例相关联。

   要将选项组与数据库实例关联，请修改数据库实例。有关更多信息，请参阅“修改 Amazon RDS 数据库实例”。

数据库实例和数据库快照可与选项组相关联。有些情况下，您可能从数据库快照还原或为数据库实例执行时间点还原。在这些情况下，与数据库快照或数据库实例关联的选项组默认与还原的数据库实例相关联。您可以将不同的选项组与还原的数据库实例相关联。但是，新选项组必须包含原始选项组中包含的所有持久性或永久性选项。持久性和永久性选项的说明如下。

选项需要额外的内存才能在数据库实例上运行。因此，您可能需要根据数据库实例当前的使用情况启动更大的实例来使用这些选项。例如，Oracle Enterprise Manager Database Control 使用约 300 MB 的 RAM。如果为小数据库实例启用该选项，则可能会遇到性能问题或内存不足错误。

持久性和永久性选项

将两种选项 (持久性和永久性) 添加到选项组时，这两种选项需要考虑一些特殊的注意事项。

如果数据库实例与选项组关联，持久性选项则无法从该选项组中删除。持久性选项的一个示例是 Microsoft SQL Server 透明数据加密 (TDE) 的 TDE 选项。您必须先将所有数据库实例与该选项组取消关联，然后才能从选项组删除持久性选项。有些情况下，您可能从数据库快照还原或执行时间点还原。在这些情况下，如果与该数据库快照关联的选项组包含持久性选项，那么您只能将还原的数据库实例与该选项组相关联。

永久性选项（如适用于 Oracle 高级安全 TDE 的 TDE 选项）绝不能从选项组中删除。您可以更改使用永久性选项的数据库实例的选项组。但是，与数据库实例关联的选项组必须包括相同的永久性选项。有些情况下，您可能从数据库快照还原或执行时间点还原。在这些情况下，如果与该数据库快照关联的选项组包含永久性选项，那么您只能将还原的数据库实例与具有该永久性选项的选项组相关联。

对于 Oracle 数据库实例，您可以复制具有选项 `Timezone` 和/或 `OLS` 的共享数据库快照。为此，在复制数据库快照时，指定一个包含这些选项的目标选项组。OLS 选项仅对于运行 Oracle 12.2 或更高版本的 Oracle 数据库实例具有永久性和持久性。有关这些选项的更多信息，请参阅 Oracle 时区 和 Oracle Label Security。

VPC 注意事项

与该数据库实例关联的选项组会链接到该数据库实例的 VPC。这意味着，如果您尝试将实例还原到不同的 VPC，则无法使用为该数据库实例分配的选项组。如果您将数据库实例还原到不同的 VPC，可以执行以下操作之一：

- 将默认选项组分配到数据库实例。
- 分配关联到该 VPC 的选项组。
- 创建新选项组并将其分配到数据库实例。

对于持久性或永久性选项（如 Oracle TDE），您必须创建新的选项组。在将数据库实例还原到不同的 VPC 中时，此选项组必须包括持久或永久选项。

选项设置可控制选项的行为。例如，Oracle 高级安全选项 `NATIVE_NETWORK_ENCRYPTION` 包含可用于为数据库实例的传入和传出网络流量指定加密算法的设置。部分选项设置已针对与 Amazon RDS 配合使用而优化，并且不能更改。

互斥选项

一些选项是互斥的。您可使用其中一个选项，而不能同时使用两个选项。以下选项是互斥的：

- Oracle Enterprise Manager Database Express、 和 适用于 Enterprise Manager Cloud Control 的 Oracle Management Agent
- Oracle 本机网络加密、 和 Oracle 安全套接字层

创建选项组

您可以创建一个新的选项组，该选项组从原定设置选项组派生其设置。然后，向新的选项组中添加一个或多个选项。或者，如果您已经有一个现有选项组，则可复制带有您已添加到新选项组的所有选项的选项组。有关更多信息，请参阅 复制选项组。

创建新选项组后，新选项组中没有选项。要了解如何将选项添加到选项组，请参阅 将选项添加到选项组。添加了您所需的选项后，即可将选项组与数据库实例相关联。这样，就可以在数据库实例上使用这些选项了。有关将选项组与数据库实例相关联的信息，请参阅 使用选项组 中适用于您的引擎的文档。

控制台

一种创建选项组的方法是使用 Amazon Web Services Management Console。

通过使用控制台创建新选项组

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择选项组。
3. 选择创建组。
4. 在创建选项组窗口中，执行以下操作：

    a. 对于 Name (名称)，输入选项组的名称，在 Amazon 账户内该名称需具有唯一性。此名称只能包含字母、数字和连字符。

b. 对于说明，输入选项组的简要说明。此说明用于显示说明。
c. 对于引擎，选择需要的数据库引擎。
d. 对于主引擎版本，请选择所需的数据库引擎主版本。

5. 要继续，请选择创建。要取消操作，请选择取消。

Amazon CLI

要创建选项组，请使用带下列必需参数的 Amazon CLI `create-option-group` 命令。

- `--option-group-name`
- `--engine-name`
- `--major-engine-version`
- `--option-group-description`

Example

下例创建了名为 `testoptiongroup` 的选项组，它是与 Oracle 企业版数据库引擎相关联的选项组。引号中的内容是描述。

对于 Linux、macOS 或 Unix：

```
aws rds create-option-group \
    --option-group-name testoptiongroup \
    --engine-name oracle-ee \
    --major-engine-version 19 \
    --option-group-description "Test option group for Oracle Database 19c EE"
```

对于 Windows：

```
aws rds create-option-group ^
    --option-group-name testoptiongroup ^
    --engine-name oracle-ee ^-
    --major-engine-version 19 ^
    --option-group-description "Test option group for Oracle Database 19c EE"
```

RDS API

要创建选项组，请调用 Amazon RDS API CreateOptionGroup 操作。包括以下参数：

- OptionGroupName
- EngineName
- MajorEngineVersion
- OptionGroupDescription

复制选项组

您可以使用 Amazon CLI 或 Amazon RDS API 复制选项组。复制选项组可能很方便。例如，当您具有现有选项组并且想在新的选项组中包含其大部分自定义参数和值时。您还可以为用于生产的选项组创建副本，然后修改此副本以测试其他选项设置。

> **Note**
> 目前，无法将选项组复制到其他 Amazon 区域。

Amazon CLI

要复制选项组，请使用 Amazon CLI copy-option-group 命令。包括以下必需选项：

- --source-option-group-identifier
- --target-option-group-identifier
- --target-option-group-description

Example

以下示例创建一个名为 new-option-group 的选项组，它是选项组 my-option-group 的本地副本。

对于 Linux、macOS 或 Unix：

```
aws rds copy-option-group \
```

```
    --source-option-group-identifier my-option-group \
    --target-option-group-identifier new-option-group \
    --target-option-group-description "My new option group"
```

对于 Windows：

```
aws rds copy-option-group ^
    --source-option-group-identifier my-option-group ^
    --target-option-group-identifier new-option-group ^
    --target-option-group-description "My new option group"
```

RDS API

要复制选项组，请调用 Amazon RDS API CopyOptionGroup 操作。包括以下必需参数。

- `SourceOptionGroupIdentifier`
- `TargetOptionGroupIdentifier`
- `TargetOptionGroupDescription`

将选项添加到选项组

您可将选项添加到现有选项组。添加了您所需的选项后，即可将选项组与数据库实例相关联，以便在数据库实例上使用其中的选项。有关将选项组与数据库实例相关联的信息，请参阅 使用选项组 中列出的适用于特定数据库引擎的文档。

在以下两种情况下，必须立即应用选项组更改：

- 当您添加用于添加或更新端口值的选项时，如 `OEM` 选项。
- 当您添加或删除某个具有含端口值选项的选项组时。

在这些情况下，请选择控制台中的 Apply Immediately (立即应用) 选项。或者，您可以在使用 Amazon CLI 时加入 `--apply-immediately` 选项，或在使用 Amazon RDS API 时将 `ApplyImmediately` 参数设置为 `true`。可立即应用不包括端口值的选项，也可在数据库实例的下一维护时段内应用此类选项。

> **Note**
>
> 如果您将安全组指定为选项组中某个选项的值，则可以通过修改选项组来管理该安全组。您无法通过修改数据库实例来更改或删除此安全组。此外，安全组不会显示在 Amazon Web

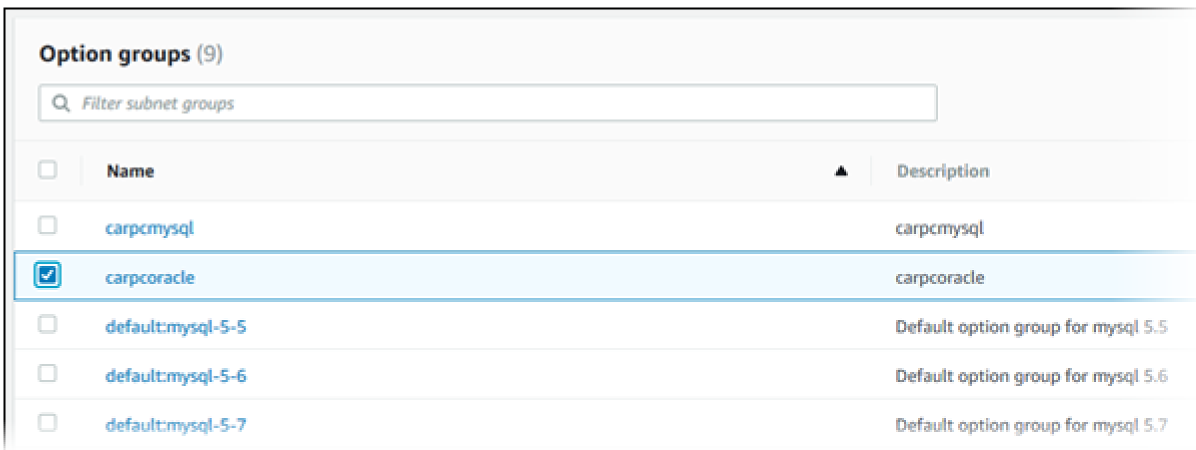
Services Management Console 的数据库实例详细信息中，也不会显示在 Amazon CLI 命令 `describe-db-instances` 的输出中。

控制台

您可以使用 Amazon Web Services Management Console将选项添加到选项组。

通过使用控制台将选项添加到选项组

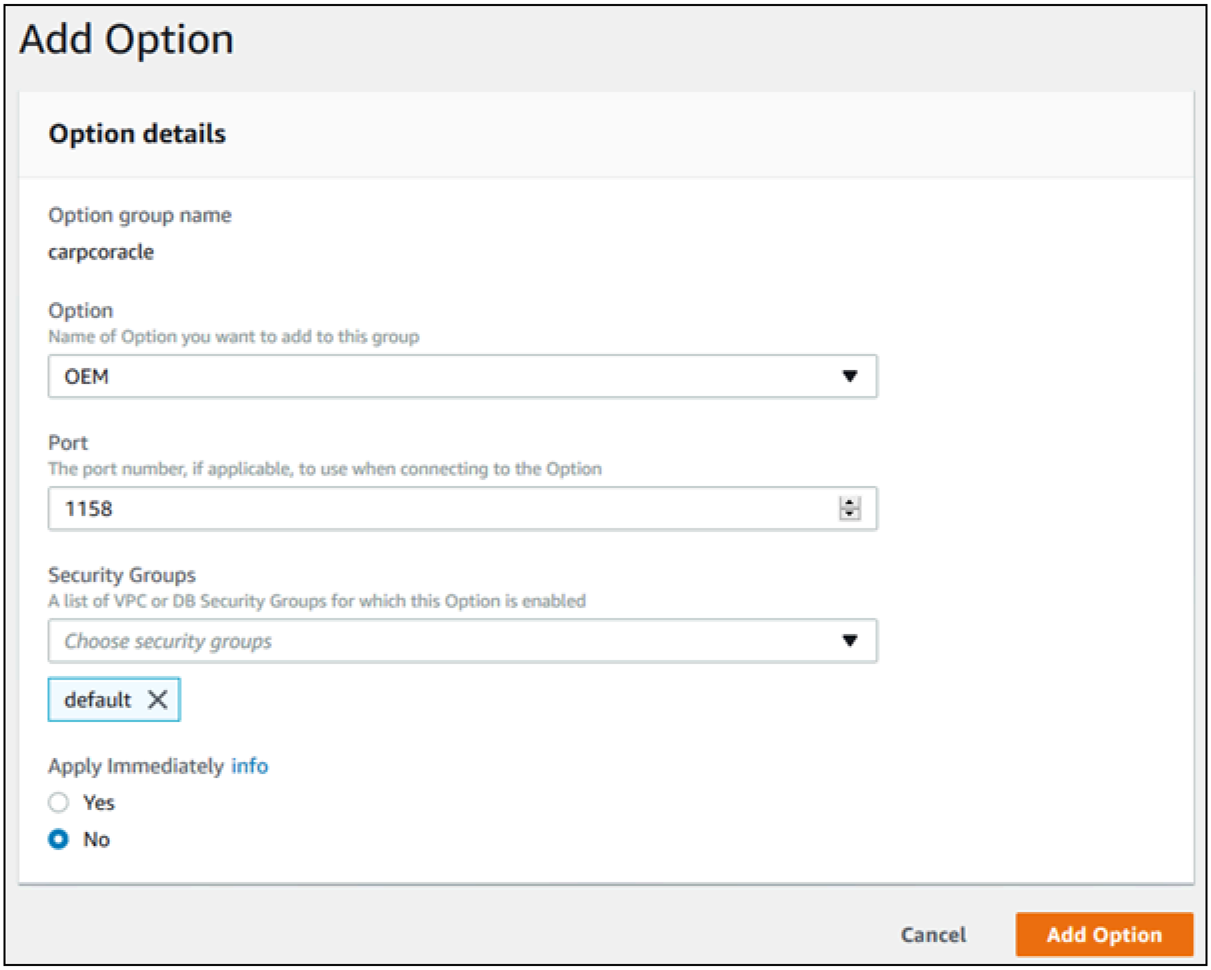
1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择选项组。
3. 选择要修改的选项组，然后选择 Add Option (添加选项)。



4. 在添加选项窗口中，执行以下操作：

   a. 选择要添加的选项。您可能需要根据您选择的选项提供其他值。例如，当您选择 `OEM` 选项时，必须同时输入端口值，并指定安全组。

   b. 要在添加选项后在所有关联数据库实例上启用该选项，对于立即应用，请选择是。如果选择否（默认），则会在下一个维护时段为每个关联数据库实例启用此选项。



5. 根据需要设置完毕后，选择 Add Option (添加选项)。

Amazon CLI

要将选项添加到选项组，请带所需选项运行 Amazon CLI[add-option-to-option-group](#) 命令。要立即在所有关联数据库实例上启用新选项，请加入 `--apply-immediately` 参数。默认情况下，会在每个关联数据库实例的下一维护时段启用此选项。包括以下必需参数：

- `--option-group-name`

Example

下例会立即将带有 `America/Los_Angeles` 设置的 `Timezone` 选项添加到名为 `testoptiongroup` 的选项组，并立即启用它。

对于 Linux、macOS 或 Unix：

```
aws rds add-option-to-option-group \
    --option-group-name testoptiongroup \
    --options "OptionName=Timezone,OptionSettings=[{Name=TIME_ZONE,Value=America/
Los_Angeles}]" \
    --apply-immediately
```

对于 Windows：

```
aws rds add-option-to-option-group ^
    --option-group-name testoptiongroup ^
    --options "OptionName=Timezone,OptionSettings=[{Name=TIME_ZONE,Value=America/
Los_Angeles}]" ^
    --apply-immediately
```

命令输出类似于以下内容：

```
...{
       "OptionName": "Timezone",
       "OptionDescription": "Change time zone",
       "Persistent": true,
       "Permanent": false,
       "OptionSettings": [
           {
               "Name": "TIME_ZONE",
               "Value": "America/Los_Angeles",
               "DefaultValue": "UTC",
               "Description": "Specifies the timezone the user wants to change the
 system time to",
               "ApplyType": "DYNAMIC",
               "DataType": "STRING",
               "AllowedValues": "Africa/Cairo,...",
               "IsModifiable": true,
               "IsCollection": false
           }
       ],
       "DBSecurityGroupMemberships": [],
       "VpcSecurityGroupMemberships": []
   }...
```

Example

以下示例将 Oracle OEM 选项添加到选项组。其中还指定了自定义端口和一对 Amazon EC2 VPC 安全组来用于该端口。

对于 Linux、macOS 或 Unix：

```
aws rds add-option-to-option-group \
    --option-group-name testoptiongroup \
    --options OptionName=OEM,Port=5500,VpcSecurityGroupMemberships="sg-test1,sg-test2"
 \
    --apply-immediately
```

对于 Windows：

```
aws rds add-option-to-option-group ^
    --option-group-name testoptiongroup ^
    --options OptionName=OEM,Port=5500,VpcSecurityGroupMemberships="sg-test1,sg-test2"
 ^
    --apply-immediately
```

命令输出类似于以下内容：

```
OPTIONGROUP  False  oracle-ee  19 arn:aws:rds:us-east-1:1234567890:og:testoptiongroup
 Test Option Group  testoptiongroup vpc-test
OPTIONS Oracle 12c EM Express   OEM     False   False   5500
VPCSECURITYGROUPMEMBERSHIPS     active  sg-test1
VPCSECURITYGROUPMEMBERSHIPS     active  sg-test2
```

Example

下例将 Oracle 选项 `NATIVE_NETWORK_ENCRYPTION` 添加到选项组，然后指定该选项的设置。如果未指定选项设置，则使用默认值。

对于 Linux、macOS 或 Unix：

```
aws rds add-option-to-option-group \
    --option-group-name testoptiongroup \
    --options '[{"OptionSettings":
[{"Name":"SQLNET.ENCRYPTION_SERVER","Value":"REQUIRED"},
{"Name":"SQLNET.ENCRYPTION_TYPES_SERVER","Value":"AES256,AES192,DES"}],"OptionName":"NATIVE_NET
 \
```

```
    --apply-immediately
```

对于 Windows：

```
aws rds add-option-to-option-group ^
    --option-group-name testoptiongroup ^
    --options "OptionSettings"=[{"Name"="SQLNET.ENCRYPTION_SERVER","Value"="REQUIRED"},
{"Name"="SQLNET.ENCRYPTION_TYPES_SERVER","Value"="AES256\,AES192\,DES"}],"OptionName"="NATIVE_N
 ^
    --apply-immediately
```

命令输出类似于以下内容：

```
...{
    "OptionName": "NATIVE_NETWORK_ENCRYPTION",
    "OptionDescription": "Native Network Encryption",
    "Persistent": false,
    "Permanent": false,
    "OptionSettings": [
        {
            "Name": "SQLNET.ENCRYPTION_TYPES_SERVER",
             "Value": "AES256,AES192,DES",
            "DefaultValue":
 "RC4_256,AES256,AES192,3DES168,RC4_128,AES128,3DES112,RC4_56,DES,RC4_40,DES40",
            "Description": "Specifies list of encryption algorithms in order of
 intended use",
            "ApplyType": "STATIC",
            "DataType": "STRING",
            "AllowedValues":
 "RC4_256,AES256,AES192,3DES168,RC4_128,AES128,3DES112,RC4_56,DES,RC4_40,DES40",
            "IsModifiable": true,
            "IsCollection": true
        },
        {
            "Name": "SQLNET.ENCRYPTION_SERVER",
            "Value": "REQUIRED",
            "DefaultValue": "REQUESTED",
            "Description": "Specifies the desired encryption behavior",
            "ApplyType": "STATIC",
            "DataType": "STRING",
            "AllowedValues": "ACCEPTED,REJECTED,REQUESTED,REQUIRED",
            "IsModifiable": true,
            "IsCollection": false
```

```
        },...
```

RDS API

要使用 Amazon RDS API 将选项添加到选项组，请使用所要添加的选项调用 ModifyOptionGroup 操作。要立即在所有关联数据库实例上启用新选项，请加入 `ApplyImmediately` 参数并将其设置为 `true`。默认情况下，会在每个关联数据库实例的下一维护时段启用此选项。包括以下必需参数：

- `OptionGroupName`

列出选项组的选项和选项设置

您可以列出选项组的所有选项和选项设置。

控制台

您可以使用 Amazon Web Services Management Console列出选项组的所有选项和选项设置。

列出选项组的选项和选项设置

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择选项组。
3. 选择选项组的名称可以显示其详细信息。此时将列出选项组中的选项和选项设置。

Amazon CLI

要列出选项组的选项和选项设置，请使用 Amazon CLI`describe-option-groups` 命令。指定您要查看其选项和设置的选项组的名称。如果不指定选项组名称，则会显示所有选项组的说明。

Example

以下示例列出了所有选项组的选项和选项设置。

```
aws rds describe-option-groups
```

Example

以下示例列出名为 `testoptiongroup` 的选项组的选项和选项设置。

```
aws rds describe-option-groups --option-group-name testoptiongroup
```

RDS API

要列出选项组的选项和选项设置，请使用 Amazon RDS API `DescribeOptionGroups` 操作。指定您要查看其选项和设置的选项组的名称。如果不指定选项组名称，则会显示所有选项组的说明。

修改选项设置

添加具有可修改的选项设置的选项后，可随时修改这些设置。如果更改选项组的选项或选项设置，这些更改将应用到与此选项组相关联的所有数据库实例。有关哪些设置对各种选项可用的详细信息，请参阅使用选项组 中适用于您的引擎的文档。

在以下两种情况下，必须立即应用选项组更改：

- 当您添加用于添加或更新端口值的选项时，如 `OEM` 选项。
- 当您添加或删除某个具有含端口值选项的选项组时。

在这些情况下，请选择控制台中的 Apply Immediately (立即应用) 选项。或者，您可以在使用 `--apply-immediately` 时加入 Amazon CLI 选项，或在使用 RDS API 时将 `ApplyImmediately` 参数设置为 `true`。可立即应用不包括端口值的选项，也可在数据库实例的下一维护时段内应用此类选项。

> **Note**
>
> 如果您将安全组指定为选项组中某个选项的值，则可以通过修改选项组来管理该安全组。您无法通过修改数据库实例来更改或删除此安全组。此外，安全组不会显示在 Amazon Web Services Management Console 的数据库实例详细信息中，也不会显示在 Amazon CLI 命令 `describe-db-instances` 的输出中。

控制台

您可以使用 Amazon Web Services Management Console来修改选项设置。

通过使用控制台来修改选项设置

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。

2. 在导航窗格中，选择选项组。
3. 选择要修改选项的选项组，然后选择修改选项。
4. 在修改选项窗口中，从安装的选项中选择要修改设置的选项。执行所需的更改。
5. 要在添加选项后启用该选项，对于立即应用，请选择是。如果选择否（默认），则会在下一个维护时段为每个关联数据库实例启用此选项。
6. 根据需要设置完毕后，选择修改选项。

Amazon CLI

要修改选项设置，请使用 Amazon CLI [add-option-to-option-group](https://docs.aws.amazon.com/cli/latest/reference/rds/add-option-to-option-group.html) 命令处理要修改的选项组和选项。默认情况下，会在每个关联数据库实例的下一维护时段启用此选项。要立即将更改应用到所有关联数据库实例，请包含 `--apply-immediately` 参数。要修改选项设置，请使用 `--settings` 参数。

Example

下例会修改在名为 `testoptiongroup` 的选项组中 Oracle Enterprise Manager Database Control (OEM) 使用的端口，并立即应用此更改。

对于 Linux、macOS 或 Unix：

```
aws rds add-option-to-option-group \
    --option-group-name testoptiongroup \
    --options OptionName=OEM,Port=5432,DBSecurityGroupMemberships=default \
    --apply-immediately
```

对于 Windows：

```
aws rds add-option-to-option-group ^
    --option-group-name testoptiongroup ^
    --options OptionName=OEM,Port=5432,DBSecurityGroupMemberships=default ^
    --apply-immediately
```

命令输出类似于以下内容：

```
OPTIONGROUP   False   oracle-ee   19   arn:aws:rds:us-east-1:1234567890:og:testoptiongroup
   Test Option Group    testoptiongroup
```

```
OPTIONS Oracle 12c EM Express   OEM     False   False   5432
DBSECURITYGROUPMEMBERSHIPS   default  authorized
```

Example

下例修改 Oracle 选项 NATIVE_NETWORK_ENCRYPTION，然后更改该选项的设置。

对于 Linux、macOS 或 Unix：

```
aws rds add-option-to-option-group \
    --option-group-name testoptiongroup \
    --options '[{"OptionSettings":
[{"Name":"SQLNET.ENCRYPTION_SERVER","Value":"REQUIRED"},
{"Name":"SQLNET.ENCRYPTION_TYPES_SERVER","Value":"AES256,AES192,DES,RC4_256"}],"OptionName":"NA
 \
    --apply-immediately
```

对于 Windows：

```
aws rds add-option-to-option-group ^
    --option-group-name testoptiongroup ^
    --options "OptionSettings"=[{"Name"="SQLNET.ENCRYPTION_SERVER","Value"="REQUIRED"},
{"Name"="SQLNET.ENCRYPTION_TYPES_SERVER","Value"="AES256\,AES192\,DES
\,RC4_256"}],"OptionName"="NATIVE_NETWORK_ENCRYPTION" ^
    --apply-immediately
```

命令输出类似于以下内容：

```
OPTIONGROUP   False  oracle-ee  19  arn:aws:rds:us-east-1:1234567890:og:testoptiongroup
   Test Option Group    testoptiongroup
OPTIONS Oracle Advanced Security - Native Network Encryption
 NATIVE_NETWORK_ENCRYPTION       False   False
OPTIONSETTINGS
 RC4_256,AES256,AES192,3DES168,RC4_128,AES128,3DES112,RC4_56,DES,RC4_40,DES40 STATIC
 STRING
   RC4_256,AES256,AES192,3DES168,RC4_128,AES128,3DES112,RC4_56,DES,RC4_40,DES40
 Specifies list of encryption algorithms in order of intended use
   True     True    SQLNET.ENCRYPTION_TYPES_SERVER  AES256,AES192,DES,RC4_256
OPTIONSETTINGS  ACCEPTED,REJECTED,REQUESTED,REQUIRED    STATIC  STRING  REQUESTED
 Specifies the desired encryption behavior  False   True  SQLNET.ENCRYPTION_SERVER
 REQUIRED
```

```
OPTIONSETTINGS  SHA1,MD5   STATIC  STRING  SHA1,MD5    Specifies list of
 checksumming algorithms in order of intended use      True    True
 SQLNET.CRYPTO_CHECKSUM_TYPES_SERVER  SHA1,MD5
OPTIONSETTINGS  ACCEPTED,REJECTED,REQUESTED,REQUIRED  STATIC  STRING
 REQUESTED     Specifies the desired data integrity behavior   False   True
 SQLNET.CRYPTO_CHECKSUM_SERVER  REQUESTED
```

RDS API

要修改选项设置，请使用 Amazon RDS API `ModifyOptionGroup` 命令处理要修改的选项组和选项。默认情况下，会在每个关联数据库实例的下一维护时段启用此选项。要立即将更改应用到所有关联数据库实例，请加入 `ApplyImmediately` 参数并将其设置为 `true`。

从选项组中删除选项

有些选项可从选项组中删除，而另一些不可以。在与选项组关联的所有数据库实例未取消关联之前，不能从该选项组中删除持续性选项。永久选项无法从选项组中删除。有关可删除选项的更多信息，请参阅 使用选项组 中列出的适用于特定引擎的文档。

如果您删除某个选项组的所有选项，Amazon RDS 将不会删除该选项组。与空选项组相关联的数据库实例会继续与之相关联；只是不具有任何活动的选项。或者，要删除数据库实例的所有选项，您可以将数据库实例与默认 (空) 选项组关联。

控制台

您可以使用 Amazon Web Services Management Console删除选项组中的选项。

通过使用控制台删除选项组中的选项

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择选项组。
3. 选择要删除选项的选项组，然后选择删除选项。
4. 在删除选项窗口中，执行以下操作：
   - 选中您要删除的选项对应的复选框。
   - 要使删除在操作后立即生效，对于立即应用，请选择是。如果选择否（默认），则会在下一个维护时段为每个关联数据库实例删除此选项。

5. 根据需要完成所有设置后，选择是的，删除。

Amazon CLI

要从选项组中删除选项，请在 Amazon CLI `remove-option-from-option-group` 命令中指定要删除的选项。默认情况下，会在每个关联数据库实例的下一维护时段删除此选项。要立即应用更改，请加入 `--apply-immediately` 参数。

Example

以下示例会从名为 `testoptiongroup` 的选项组中移除 Oracle Enterprise Manager Database Control (OEM) 选项，并立即应用此更改。

对于 Linux、macOS 或 Unix：

```
aws rds remove-option-from-option-group \
    --option-group-name testoptiongroup \
    --options OEM \
    --apply-immediately
```

对于 Windows：

```
aws rds remove-option-from-option-group ^
```

```
    --option-group-name testoptiongroup ^
    --options OEM ^
    --apply-immediately
```

命令输出类似于以下内容：

```
OPTIONGROUP    testoptiongroup oracle-ee   19    Test option group
```

RDS API

要从选项组中删除选项，请使用 Amazon RDS API ModifyOptionGroup 操作。默认情况下，会在每个关联数据库实例的下一维护时段删除此选项。要立即应用更改，请加入 ApplyImmediately 参数，并将其设置为 true。

包括以下参数：

- OptionGroupName
- OptionsToRemove.OptionName

删除选项组

只有在选项组满足以下条件时，才能删除该选项组：

- 它未与任何 Amazon RDS 资源关联。选项组可以与数据库实例、手动数据库快照或自动数据库快照关联。
- 它不是默认选项组。

要识别数据库实例和数据库快照使用的选项组，您可以使用以下 CLI 命令：

```
aws rds describe-db-instances \
    --query 'DBInstances[*].
[DBInstanceIdentifier,OptionGroupMemberships[].OptionGroupName]'

aws rds describe-db-snapshots | jq -r '.DBSnapshots[] | "\(.DBInstanceIdentifier),
\(.OptionGroupName)"' | sort | uniq
```

如果您尝试删除与 RDS 资源关联的选项组，将返回类似于下面的错误。

```
An error occurred (InvalidOptionGroupStateFault) when calling the DeleteOptionGroup
 operation: The option group 'optionGroupName' cannot be deleted because it is in use.
```

查找与选项组关联的 Amazon RDS 资源

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择选项组。
3. 选择选项组的名称可以显示其详细信息。
4. 请查看关联的实例和快照部分，了解关联的 Amazon RDS 资源。

如果数据库实例已与选项组关联，请修改数据库实例来使用其他选项组。有关更多信息，请参阅 修改 Amazon RDS 数据库实例。

如果手动数据库快照与选项组关联，请修改数据库快照以使用其他选项组。您可以使用 Amazon CLI `modify-db-snapshot` 命令来执行此操作。

> **Note**
> 您无法修改自动数据库快照的选项组。

控制台

一种删除选项组的方法是使用 Amazon Web Services Management Console。

使用控制台删除选项组

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择选项组。
3. 选择选项组。
4. 选择 Delete group (删除组)。
5. 在确认页面，选择 Delete (删除) 删除选项组，或选择 Cancel (取消) 取消删除。

Amazon CLI

要删除选项组，请使用带下列必需参数的 Amazon CLI `delete-option-group` 命令。

- `--option-group-name`

Example

以下示例将删除名为 `testoptiongroup` 的选项组。

对于 Linux、macOS 或 Unix：

```
aws rds delete-option-group \
    --option-group-name testoptiongroup
```

对于 Windows：

```
aws rds delete-option-group ^
    --option-group-name testoptiongroup
```

RDS API

要删除选项组，请调用 Amazon RDS API `DeleteOptionGroup` 操作。包括以下参数：

- `OptionGroupName`

Amazon RDS 的参数组

数据库参数指定数据库的配置方式。例如，数据库参数可以指定要分配给数据库的资源量（如内存）。

您可以通过将数据库实例和多可用区 数据库集群与参数组进行关联来管理数据库配置。Amazon RDS 使用默认设置定义参数组。您还可以使用自定义设置定义您自己的参数组。

> **Note**
>
> 某些数据库引擎提供了附加功能，您可以将这些功能作为选项组中的选项添加到数据库中。有关选项组的信息，请参阅使用选项组。

主题

- 参数组概述
- Amazon RDS 数据库实例的数据库参数组
- 使用多可用区数据库集群的数据库集群参数组
- 比较数据库参数组
- 指定数据库参数

参数组概述

数据库参数组就像是引擎配置值的容器，这些值应用于一个或多个数据库实例。

数据库集群参数组仅适用于多可用区数据库集群。在多可用区数据库集群中，数据库集群参数组中的设置应用于集群中的所有数据库实例。数据库引擎和数据库引擎版本的原定设置数据库参数组用于数据库集群中的每个数据库实例。

主题

- 原定设置和自定义参数组
- 静态和动态数据库实例参数
- 静态和动态数据库集群参数
- 字符集参数
- 支持的参数和参数值

原定设置和自定义参数组

如果创建的数据库实例未指定数据库参数组，数据库实例将使用默认的数据库参数组。同样，如果您在创建多可用区 数据库集群时未指定数据库集群参数组，数据库集群将使用默认数据库集群参数组。每个默认参数组包含数据库引擎默认值和 Amazon RDS 系统默认值，具体根据引擎、计算等级及实例的分配存储空间而定。

默认参数组的参数设置无法修改。您可以改而执行以下操作：

1. 创建新参数组。
2. 更改所需参数的设置。并非参数组中的所有数据库引擎参数都可供修改。
3. 修改数据库实例或数据库集群以关联新的参数组。

   将新数据库参数组与数据库实例进行关联时，关联将立即发生。有关修改数据库实例的信息，请参阅修改 Amazon RDS 数据库实例。有关修改多可用区数据库集群的信息，请参阅修改 Amazon RDS 的多可用区数据库集群。

> **Note**
> 如果您修改了数据库实例以使用自定义参数组，然后启动数据库实例，RDS 会在启动过程中自动重启数据库实例。对于启用了 AlwaysOn 或镜像选项的 RDS for SQL Server 多可用区实例，在启动过程结束后重启实例时，预计会进行失效转移。

仅在重启数据库实例后，RDS 才会在新关联的参数组中应用修改后的静态和动态参数。但是，如果在将数据库参数组与数据库实例关联之后修改数据库参数组中的动态参数，这些更改将立即得到应用，而无需重启。有关更改数据库集群参数组的信息，请参阅 修改 Amazon RDS 数据库实例。

如果您更新了数据库参数组内的参数，更改将应用于与该参数组关联的所有数据库实例。同样，如果您更新了多可用区数据库参数组内的参数，更改将应用于与该数据库集群参数组关联的所有 Aurora 数据库集群。

如果您不想从头开始创建参数组，您可以使用 Amazon CLI copy-db-parameter-group 命令或 copy-db-cluster-parameter-group 命令复制现有的参数组。您可能会发现复制参数组在某些情况下很有用。例如，您可能希望在新的数据库参数组中包含现有数据库参数组的大多数自定义参数和值。

静态和动态数据库实例参数

数据库实例参数是静态的，也可以是动态的。它们在以下方面有所不同：

- 更改静态参数并保存数据库参数组时，参数更改将在手动重启关联的数据库实例后生效。对于静态参数，控制台始终将 `pending-reboot` 用于 `ApplyMethod`。
- 当您更改动态参数时，原定设置情况下，参数更改将立即生效，而无需重启。当您使用 Amazon Web Services Management Console 更改数据库实例参数值时，对于动态参数，它始终使用 `immediate` 作为 `ApplyMethod`。要将参数更改推迟到重启关联的数据库实例之后，请使用 Amazon CLI 或 RDS API。将 `ApplyMethod` 设置为 `pending-reboot` 以进行参数更改。

> **Note**
> 将 `pending-reboot` 与 Amazon CLI 中的参数或 RDS for SQL Server 数据库实例上的 RDS API 搭配使用，会产生错误。在 RDS for SQL Server 上使用 `apply-immediately`。

有关使用 Amazon CLI 更改参数值的更多信息，请参阅 modify-db-parameter-group。有关使用 RDS API 更改参数值的更多信息，请参阅 ModifyDBParameterGroup。

如果数据库实例未使用对其关联的数据库参数组所做的最新更改，控制台将针对数据库参数组显示 pending-reboot（待重启）状态。此状态不会在下一个维护时段期间导致自动重启。要向该数据库实例应用最新的参数更改，请手动重启数据库实例。

静态和动态数据库集群参数

数据库集群参数是静态的，也可以是动态的。它们在以下方面有所不同：

- 更改静态参数并保存数据库集群参数组时，参数更改将在手动重启关联数据库集群后生效。对于静态参数，控制台始终将 `pending-reboot` 用于 `ApplyMethod`。
- 当您更改动态参数时，原定设置情况下，参数更改将立即生效，而无需重启。当您使用 Amazon Web Services Management Console 更改数据库集群参数值时，对于动态参数，它始终使用 `immediate` 作为 `ApplyMethod`。要将参数更改推迟到重启关联的数据库集群之后，请使用 Amazon CLI 或 RDS API。将 `ApplyMethod` 设置为 `pending-reboot` 以进行参数更改。

有关使用 Amazon CLI 更改参数值的更多信息，请参阅 modify-db-cluster-parameter-group。有关使用 RDS API 更改参数值的更多信息，请参阅 ModifyDBClusterParameterGroup。

字符集参数

在创建数据库实例或多可用区数据库集群之前，在参数组中设置与数据库的字符集或排序规则相关的任何参数。在其中创建数据库之前也要这样做。这样，您可确保原定设置数据库以及新数据库使用您指定的字符集和排序规则值。如果您更改字符集或排序规则参数，则参数更改不会应用于现有数据库。

对于某些数据库引擎，您可使用 ALTER DATABASE 命令更改现有数据库的字符集或排序规则值，例如：

```
ALTER DATABASE database_name CHARACTER SET character_set_name COLLATE collation;
```

有关更改数据库字符集或排序规则值的更多信息，请查阅数据库引擎文档。

支持的参数和参数值

要确定数据库引擎支持的参数，请查看数据库实例或数据库集群使用的数据库参数组和数据库集群参数组中的参数。有关更多信息，请参阅在 Amazon RDS 中查看数据库参数组的参数值和查看数据库集群参数组的参数值。

在许可情况下，您可以使用表达式、公式和函数指定整数和布尔参数值。函数可以包含数学对数表达式。但是，并非所有参数都支持对参数值使用表达式、公式和函数。有关更多信息，请参阅 指定数据库参数。

在参数组内设置参数不恰当可能会产生意外的不利影响，包括性能降低和系统不稳定。修改数据库参数时应始终保持谨慎，且在修改参数组前备份数据。将参数组更改应用于生产数据库实例或数据库集群前，请在测试数据库实例或数据库集群上尝试进行这些参数组设置更改。

Amazon RDS 数据库实例的数据库参数组

数据库实例使用数据库参数组。以下各节介绍配置和管理数据库实例参数组。

主题

- 在 Amazon RDS 中创建数据库参数组
- 在 Amazon RDS 中将数据库参数组与数据库实例关联
- 在 Amazon RDS 中修改数据库参数组中的参数
- 在 Amazon RDS 中将数据库参数组中的参数重置为其默认值
- 在 Amazon RDS 中复制数据库参数组
- 在 Amazon RDS 中列出数据库参数组

- 在 Amazon RDS 中查看数据库参数组的参数值
- 在 Amazon RDS 中删除数据库参数组

在 Amazon RDS 中创建数据库参数组

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 创建新数据库参数组。

以下限制适用于数据库参数组名称：

- 名称必须为 1 到 255 个字母、数字或连字符。

  原定设置参数组名称可以包含句点，例如 `default.mysql8.0`。但是，自定义参数组名称不能包含句点。
- 第一个字符必须是字母。
- 名称不能以连字符结束，也不能包含两个连续的连字符。

控制台

创建数据库参数组

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择参数组。
3. 选择创建参数组。
4. 对于参数组名称，请输入新数据库参数组的名称。
5. 对于描述，请输入对新数据库参数组的描述。
6. 对于引擎类型，请选择所需的数据库引擎。
7. 对于参数组系列，请选择一个数据库参数组系列。
8. 对于类型，如果适用，请选择数据库参数组。
9. 选择创建。

Amazon CLI

要创建数据库参数组，请使用 Amazon CLI `create-db-parameter-group` 命令。以下示例为 8.0 版 MySQL 创建名为 mydbparametergroup 的数据库参数组，其说明为“我的新参数组”。

包括以下必需参数：

- `--db-parameter-group-name`
- `--db-parameter-group-family`
- `--description`

要列出所有可用的参数组系列，请使用以下命令：

```
aws rds describe-db-engine-versions --query "DBEngineVersions[].DBParameterGroupFamily"
```

> **Note**
>
> 输出包含重复项。

Example

对于 Linux、macOS 或 Unix：

```
aws rds create-db-parameter-group \
    --db-parameter-group-name mydbparametergroup \
    --db-parameter-group-family MySQL8.0 \
    --description "My new parameter group"
```

对于 Windows：

```
aws rds create-db-parameter-group ^
    --db-parameter-group-name mydbparametergroup ^
    --db-parameter-group-family MySQL8.0 ^
    --description "My new parameter group"
```

此命令生成类似于下述信息的输出：

```
DBPARAMETERGROUP  mydbparametergroup  mysql8.0  My new parameter group
```

RDS API

要创建数据库参数组，请使用 RDS API `CreateDBParameterGroup` 操作。

包括以下必需参数：

- `DBParameterGroupName`
- `DBParameterGroupFamily`
- `Description`

在 Amazon RDS 中将数据库参数组与数据库实例关联

您可以使用自定义设置创建自己的数据库参数组。之后可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 关联数据库参数组与数据库实例。在创建或修改数据库实例时可以执行此操作。

有关创建数据库参数组的信息，请参阅在 Amazon RDS 中创建数据库参数组。有关创建数据库实例的信息，请参阅。创建 Amazon RDS 数据库实例有关修改数据库实例的信息，请参阅修改 Amazon RDS 数据库实例。

> **Note**
>
> 将新数据库参数组与数据库实例关联时，修改后的静态和动态参数仅在数据库实例重新启动后得到应用。但是，如果在将数据库参数组与数据库实例关联之后修改数据库参数组中的动态参数，这些更改将立即得到应用，而无需重启。

控制台

关联数据库参数组与数据库实例

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择数据库，然后选择要修改的数据库实例。
3. 选择修改。将显示修改数据库实例页面。
4. 更改数据库参数组设置。
5. 选择继续，查看修改摘要。
6. （可选）选择立即应用以立即应用更改。选择此选项在某些情况下可能导致中断。有关更多信息，请参阅使用计划修改设置。
7. 在确认页面上，检查您的更改。如果更改正确无误，请选择 Modify DB Instance（修改数据库实例）保存更改。

   也可以选择 Back (返回) 编辑您的更改，或选择 Cancel (取消) 取消更改。

Amazon CLI

要关联数据库参数组与数据库实例，请使用 Amazon CLI `modify-db-instance` 命令及以下选项：

- `--db-instance-identifier`
- `--db-parameter-group-name`

以下示例关联了 `mydbpg` 数据库参数组与 `database-1` 数据库实例。使用 `--apply-immediately` 可立即应用更改。使用 `--no-apply-immediately` 可在下一维护时段内应用更改。有关更多信息，请参阅使用计划修改设置。

Example

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier database-1 \
    --db-parameter-group-name mydbpg \
    --apply-immediately
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier database-1 ^
    --db-parameter-group-name mydbpg ^
    --apply-immediately
```

RDS API

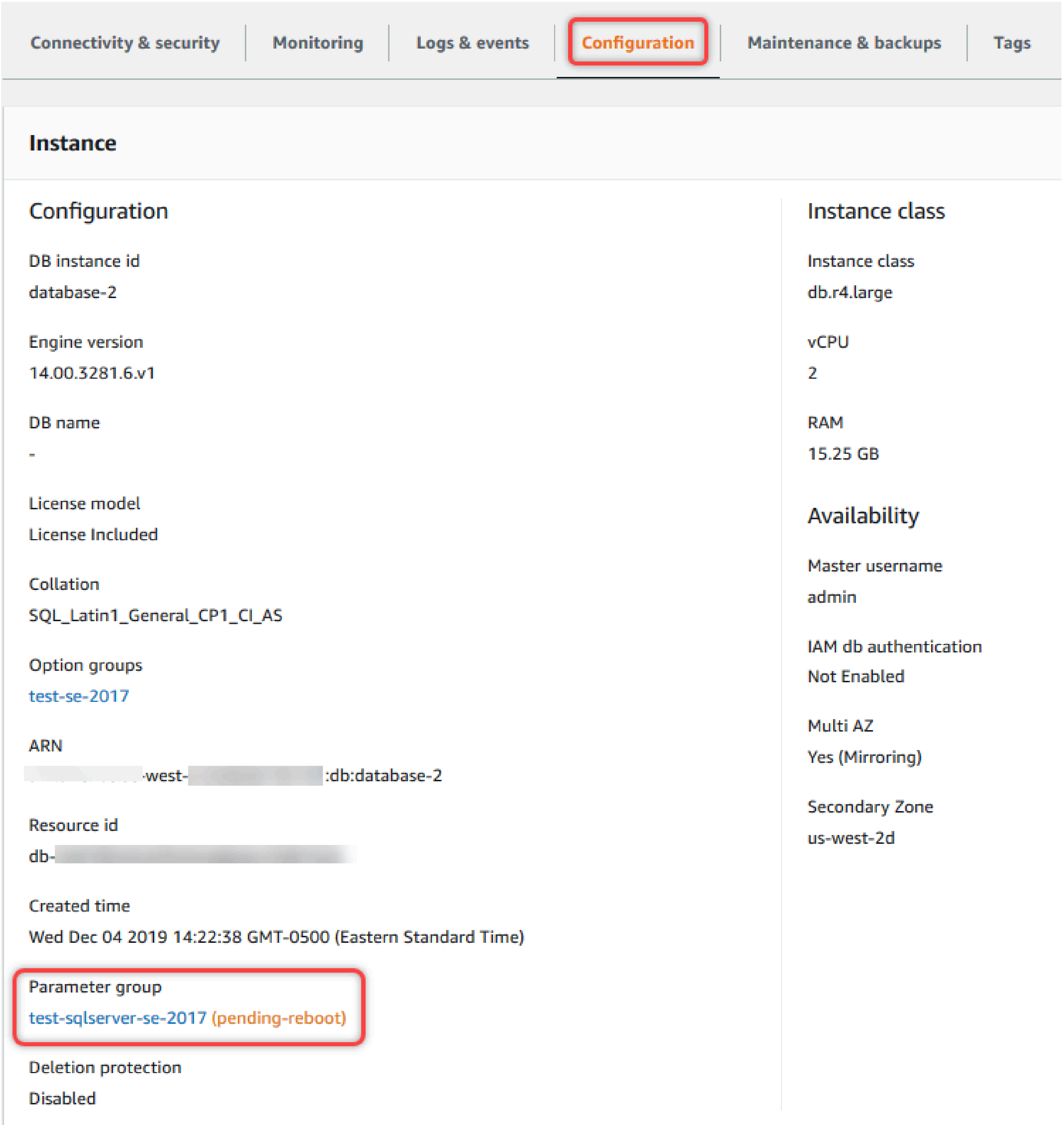
要关联数据库参数组与数据库实例，请使用 RDS API `ModifyDBInstance` 操作及以下参数：

- `DBInstanceName`
- `DBParameterGroupName`

在 Amazon RDS 中修改数据库参数组中的参数

您可以修改客户创建的数据库参数组中的参数值，但不能更改默认数据库参数组中的参数值。对客户创建的数据库参数组中的参数所做的更改将应用于与此数据库参数组关联的所有数据库实例。

对某些参数的更改将立即应用于数据库实例，而无需重新启动。而对其他一些参数进行的更改，只有在重新启动数据库实例之后，才会应用。RDS 控制台在 Configuration (配置) 选项卡上显示与数据库实例关联的数据库参数组的状态。例如，假设数据库实例未使用对其关联的数据库参数组所做的最新更改。如果是这样，RDS 控制台将显示状态为 pending-reboot（待重启）的数据库参数组。要向该数据库实例应用最新的参数更改，请手动重启数据库实例。我们建议您在修改参数后，始终重启数据库实例。如果您不重启数据库实例，则失效转移操作需要的时间可能比正常情况下更长。


Connectivity & security
Monitoring
Logs & events
Configuration
Maintenance & backups
Tags
Instance
Configuration
DB instance id
database-2
Engine version
14.00.3281.6.v1
DB name
-
License model
License Included
Collation
SQL_Latin1_General_CP1_CI_AS
Option groups
test-se-2017
ARN
-west- :db:database-2
Resource id
db-
Created time
Wed Dec 04 2019 14:22:38 GMT-0500 (Eastern Standard Time)
Parameter group
test-sqlserver-se-2017 (pending-reboot)
Deletion protection
Disabled
Instance class
Instance class
db.r4.large
vCPU
2
RAM
15.25 GB
Availability
Master username
admin
IAM db authentication
Not Enabled
Multi AZ
Yes (Mirroring)
Secondary Zone
us-west-2d

控制台

修改数据库参数组中的参数

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择参数组。
3. 在此列表中，选择要修改的参数组的名称。
4. 对于 Parameter group actions (参数组操作)，选择 Edit (编辑)。
5. 更改要修改的参数的值。您可使用对话框右上方的箭头键滚动参数。

   您无法更改默认参数组中的值。
6. 选择保存更改。

Amazon CLI

要修改数据库参数组，请使用 Amazon CLI `modify-db-parameter-group` 命令及以下必需选项：

- `--db-parameter-group-name`
- `--parameters`

以下示例修改了名为 mydbparametergroup 的数据库参数组的 `max_connections` 和 `max_allowed_packet` 值。

Example

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-parameter-group \
    --db-parameter-group-name mydbparametergroup \
    --parameters
 "ParameterName=max_connections,ParameterValue=250,ApplyMethod=immediate" \

 "ParameterName=max_allowed_packet,ParameterValue=1024,ApplyMethod=immediate"
```

对于 Windows：

```
aws rds modify-db-parameter-group ^
```

```
    --db-parameter-group-name mydbparametergroup ^
    --parameters
 "ParameterName=max_connections,ParameterValue=250,ApplyMethod=immediate" ^

 "ParameterName=max_allowed_packet,ParameterValue=1024,ApplyMethod=immediate"
```

该命令产生类似下面的输出：

```
DBPARAMETERGROUP  mydbparametergroup
```

RDS API

要修改数据库参数组，请使用 RDS API `ModifyDBParameterGroup` 操作及以下必要参数：

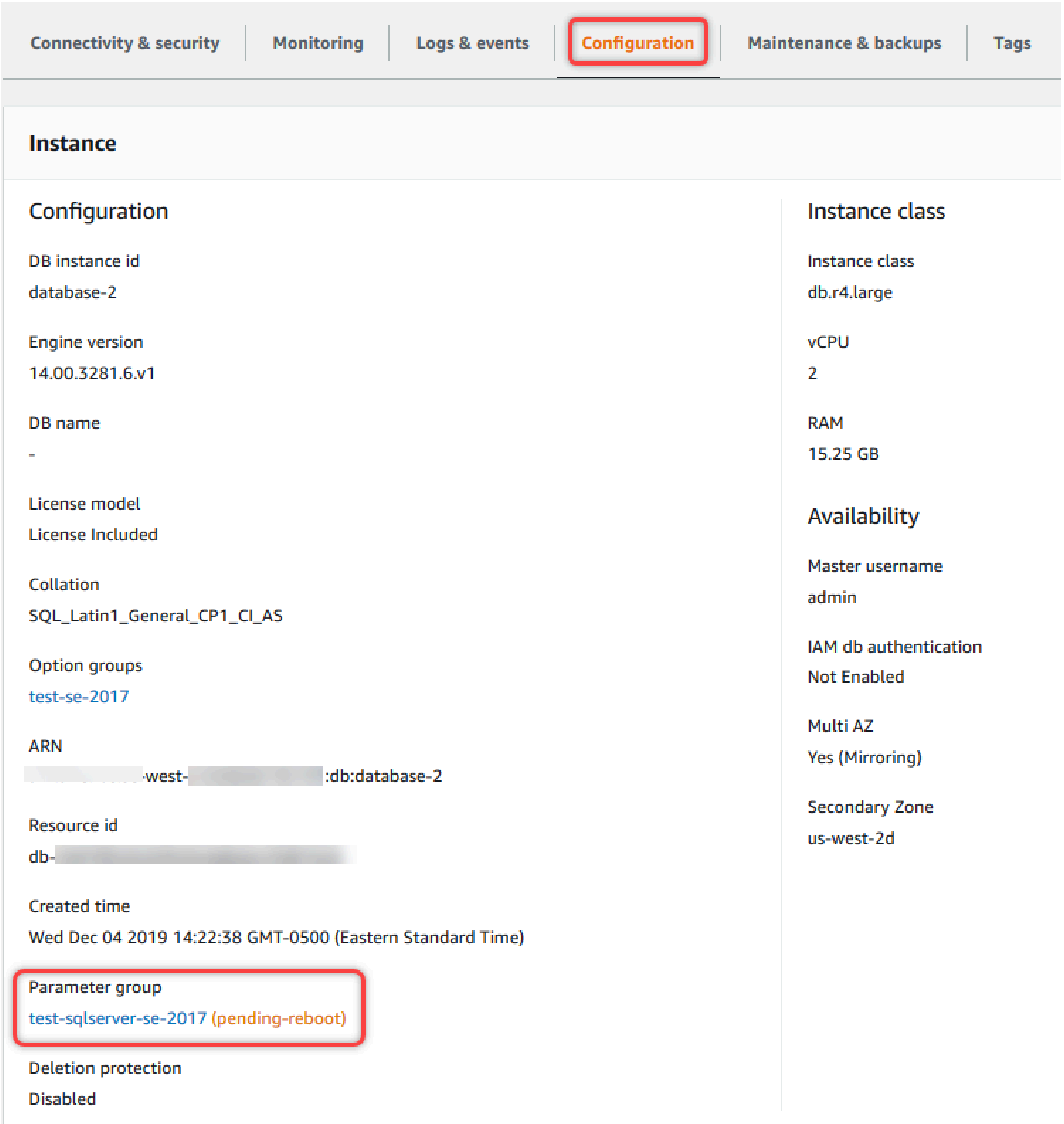
- `DBParameterGroupName`
- `Parameters`

在 Amazon RDS 中将数据库参数组中的参数重置为其默认值

您可以将客户创建的数据库参数组中的参数值重置为默认值。对客户创建的数据库参数组中的参数所做的更改将应用于与此数据库参数组关联的所有数据库实例。

使用控制台时，可以将特定参数重置为其默认值。但是，您无法轻松地一次性重置数据库参数组中的所有参数。使用 Amazon CLI 或 RDS API 时，可以将特定参数重置为其默认值。您还可以一次性重置数据库参数组中的所有参数。

对某些参数的更改将立即应用于数据库实例，而无需重新启动。而对其他一些参数进行的更改，只有在重新启动数据库实例之后，才会应用。RDS 控制台在 Configuration (配置) 选项卡上显示与数据库实例关联的数据库参数组的状态。例如，假设数据库实例未使用对其关联的数据库参数组所做的最新更改。如果是这样，RDS 控制台将显示状态为 pending-reboot（待重启）的数据库参数组。要向该数据库实例应用最新的参数更改，请手动重启数据库实例。



> **Note**
>
> 在默认数据库参数组中，参数始终设置为默认值。

控制台

将数据库参数组中的参数重置为默认值的方法

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择参数组。
3. 在列表中，选择参数组。
4. 对于 Parameter group actions (参数组操作)，选择 Edit (编辑)。
5. 选择要重置为默认值的参数。您可使用对话框右上方的箭头键滚动参数。

   您无法重置默认参数组中的值。
6. 选择重置，然后通过选择重置参数进行确认。

Amazon CLI

要重置数据库参数组中的部分或所有参数，请使用带以下必需选项的 Amazon CLI `reset-db-parameter-group` 命令：`--db-parameter-group-name`。

要重置数据库参数组中的所有参数，请指定 `--reset-all-parameters` 选项。要重置特定参数，请指定 `--parameters` 选项。

以下示例将名为 mydbparametergroup 的数据库参数组中的所有参数重置为其默认值。

Example

对于 Linux、macOS 或 Unix：

```
aws rds reset-db-parameter-group \
    --db-parameter-group-name mydbparametergroup \
    --reset-all-parameters
```

对于 Windows：

```
aws rds reset-db-parameter-group ^
    --db-parameter-group-name mydbparametergroup ^
    --reset-all-parameters
```

以下示例展示了在名为 mydbparametergroup 的数据库参数组中将 `max_connections` 和 `max_allowed_packet` 选项重置为各自的默认值。

Example

对于 Linux、macOS 或 Unix：

```
aws rds reset-db-parameter-group \
    --db-parameter-group-name mydbparametergroup \
    --parameters "ParameterName=max_connections,ApplyMethod=immediate" \
                 "ParameterName=max_allowed_packet,ApplyMethod=immediate"
```

对于 Windows：

```
aws rds reset-db-parameter-group ^
    --db-parameter-group-name mydbparametergroup ^
    --parameters "ParameterName=max_connections,ApplyMethod=immediate" ^
                 "ParameterName=max_allowed_packet,ApplyMethod=immediate"
```

该命令产生类似下面的输出：

```
DBParameterGroupName  mydbparametergroup
```

RDS API

要将数据库参数组中的参数重置为默认值，请使用带以下必需参数的 RDS API `ResetDBParameterGroup` 命令：`DBParameterGroupName`。

要重置数据库参数组中的所有参数，请将 `ResetAllParameters` 参数设置为 `true`。要重置特定参数，请指定 `Parameters` 参数。

在 Amazon RDS 中复制数据库参数组

您可以复制您创建的自定义数据库参数组。复制参数组可能是一种方便的解决方案。例如，您创建了一个数据库参数组，并且想在新的数据库参数组中包含其大部分自定义参数和值。您可以使用 Amazon Web Services Management Console复制数据库参数组。还可以使用 Amazon CLI copy-db-parameter-group 命令或 RDS API CopyDBParameterGroup 操作。

复制数据库参数组之后，请至少等待 5 分钟，再创建使用该数据库参数组作为默认参数组的第一个数据库实例。这样，在使用参数组前，Amazon RDS 可以完成全部复制操作。这对于在为数据库实例创

建默认数据库时十分关键的参数非常重要。示例如 `character_set_database` 参数定义的默认数据库的字符集。请使用 [Amazon RDS 控制台](https://console.aws.amazon.com/rds/)的参数组选项或使用 describe-db-parameters 命令来验证是否已创建数据库参数组。

> **Note**
>
> 您无法复制默认参数组。不过，您可以创建基于默认参数组的新参数组。
>
> 您无法将数据库参数组复制到其他 Amazon Web Services 账户或 Amazon Web Services 区域。

控制台

复制数据库参数组

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择参数组。
3. 在列表中，选择要复制的自定义参数组。
4. 对于 Parameter group actions (参数组操作)，选择 Copy (复制)。
5. 在 New DB parameter group identifier (新数据库参数组标识符) 中，输入新参数组的名称。
6. 在 Description (描述) 中，输入新参数组的描述。
7. 选择 Copy (复制)。

Amazon CLI

要复制数据库参数组，请使用 Amazon CLI `copy-db-parameter-group` 命令及以下必要选项：

- `--source-db-parameter-group-identifier`
- `--target-db-parameter-group-identifier`
- `--target-db-parameter-group-description`

以下示例创建一个名为 `mygroup2` 的新数据库参数组，它是数据库参数组 `mygroup1` 的副本。

Example

对于 Linux、macOS 或 Unix：

```
aws rds copy-db-parameter-group \
    --source-db-parameter-group-identifier mygroup1 \
    --target-db-parameter-group-identifier mygroup2 \
    --target-db-parameter-group-description "DB parameter group 2"
```

对于 Windows：

```
aws rds copy-db-parameter-group ^
    --source-db-parameter-group-identifier mygroup1 ^
    --target-db-parameter-group-identifier mygroup2 ^
    --target-db-parameter-group-description "DB parameter group 2"
```

RDS API

要复制数据库参数组，请将 RDS API `CopyDBParameterGroup` 操作与下列必需参数配合使用：

- `SourceDBParameterGroupIdentifier`
- `TargetDBParameterGroupIdentifier`
- `TargetDBParameterGroupDescription`

在 Amazon RDS 中列出数据库参数组

您可以列出为Amazon 账户创建的数据库参数组。

> **Note**
> 当您为特定数据库引擎和版本创建数据库实例时，将自动从默认参数模板创建默认参数组。这些默认参数组包含首选参数设置，并且无法修改。当您创建自定义参数组时，可以修改参数设置。

控制台

列出 Amazon 账户的所有数据库参数组

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择参数组。

数据库参数组将显示在列表中。

Amazon CLI

要列出 Amazon 账户的所有数据库参数组，请使用 Amazon CLI describe-db-parameter-groups 命令。

Example

下例列出了 Amazon 账户的所有可用数据库参数组。

```
aws rds describe-db-parameter-groups
```

此命令会返回类似以下内容的响应：

```
DBPARAMETERGROUP  default.mysql8.0     mysql8.0  Default parameter group for MySQL8.0
DBPARAMETERGROUP  mydbparametergroup   mysql8.0  My new parameter group
```

以下示例描述了 mydbparamgroup1 参数组。

对于 Linux、macOS 或 Unix：

```
aws rds describe-db-parameter-groups \
    --db-parameter-group-name mydbparamgroup1
```

对于 Windows：

```
aws rds describe-db-parameter-groups ^
    --db-parameter-group-name mydbparamgroup1
```

此命令会返回类似以下内容的响应：

```
DBPARAMETERGROUP  mydbparametergroup1  mysql8.0  My new parameter group
```

RDS API

要列出 Amazon 账户的所有数据库参数组，请使用 RDS API DescribeDBParameterGroups 操作。

在 Amazon RDS 中查看数据库参数组的参数值

您可获得数据库参数组内所有参数的列表及它们的值。

控制台

查看数据库参数组的参数值

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择参数组。

   数据库参数组将显示在列表中。
3. 选择参数组名称以查看其参数列表。

Amazon CLI

要查看数据库参数组的参数值，请使用带下列必需参数的 Amazon CLI `describe-db-parameters` 命令。

- `--db-parameter-group-name`

Example

以下示例列出名为 mydbparametergroup 的数据库参数组的参数和参数值。

```
aws rds describe-db-parameters --db-parameter-group-name mydbparametergroup
```

此命令会返回类似以下内容的响应：

```
DBPARAMETER  Parameter Name            Parameter Value  Source           Data Type
 Apply Type  Is Modifiable
DBPARAMETER  allow-suspicious-udfs                      engine-default   boolean
 static      false
DBPARAMETER  auto_increment_increment                   engine-default   integer
 dynamic     true
DBPARAMETER  auto_increment_offset                      engine-default   integer
 dynamic     true
DBPARAMETER  binlog_cache_size         32768            system           integer
 dynamic     true
```

```
DBPARAMETER   socket                    /tmp/mysql.sock   system           string
 static       false
```

RDS API

要查看数据库参数组的参数值，请使用带下列所需参数的 RDS API `DescribeDBParameters` 命令。

- `DBParameterGroupName`

在 Amazon RDS 中删除数据库参数组

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 删除数据库参数组。仅当参数组未与数据库实例关联时，才能将其删除。

控制台

删除数据库参数组

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择参数组。

   数据库参数组将显示在列表中。
3. 选择要删除的参数组的名称。
4. 选择操作，然后选择删除。
5. 查看参数组名称，然后选择删除。

Amazon CLI

要删除数据库参数组，请使用带以下必需参数的 Amazon CLI `delete-db-parameter-group` 命令。

- `--db-parameter-group-name`

Example

以下示例删除一个名为 mydbparametergroup 的数据库参数组。

```
aws rds delete-db-parameter-group --db-parameter-group-name mydbparametergroup
```

RDS API

要删除数据库参数组，请使用带以下必需参数的 RDS API DeleteDBParameterGroup 命令。

- DBParameterGroupName

使用多可用区数据库集群的数据库集群参数组

多可用区数据库集群使用数据库集群参数组。以下各节介绍配置和管理数据库集群参数组。

主题

创建数据库集群参数组

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 创建新数据库集群参数组。

创建数据库集群参数组之后，至少等待 5 分钟，然后创建使用该数据库集群参数组的数据库集群。这样，Amazon RDS 就可以在新数据库集群使用参数组之前完全创建此参数组。您可以使用 Amazon RDS 控制台的 Parameter groups（参数组）页面，或者使用 describe-db-cluster-parameters 命令，以验证是否创建了数据库集群参数组。

以下限制适用于数据库集群参数组名称：

- 名称必须为 1 到 255 个字母、数字或连字符。

  原定设置参数组名称可以包含句点，例如 `default.mysql5.7`。但是，自定义参数组名称不能包含句点。
- 第一个字符必须是字母。

- 名称不能以连字符结束，也不能包含两个连续的连字符。

控制台

若要创建数据库集群参数组

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择参数组。
3. 选择创建参数组。
4. 对于参数组名称，输入新数据库集群参数组的名称。
5. 对于描述，输入新数据库集群参数组的描述。
6. 对于引擎类型，请选择数据库引擎。
7. 对于参数组系列，请选择一个数据库参数组系列。
8. 选择创建。

Amazon CLI

要创建数据库集群参数组，请使用 Amazon CLI `create-db-cluster-parameter-group` 命令。

以下示例为 RDS for MySQL 版本 8.0 创建名为 mydbclusterparametergroup 的数据库集群参数组，其说明为“我的新集群参数组”。

包括以下必需参数：

- `--db-cluster-parameter-group-name`
- `--db-parameter-group-family`
- `--description`

要列出所有可用的参数组系列，请使用以下命令：

```
aws rds describe-db-engine-versions --query "DBEngineVersions[].DBParameterGroupFamily"
```

> **Note**
>
> 输出包含重复项。

Example

对于 Linux、macOS 或 Unix：

```
aws rds create-db-cluster-parameter-group \
    --db-cluster-parameter-group-name mydbclusterparametergroup \
    --db-parameter-group-family mysql8.0 \
    --description "My new cluster parameter group"
```

对于 Windows：

```
aws rds create-db-cluster-parameter-group ^
    --db-cluster-parameter-group-name mydbclusterparametergroup ^
    --db-parameter-group-family mysql8.0 ^
    --description "My new cluster parameter group"
```

此命令生成类似于下述信息的输出：

```
{
    "DBClusterParameterGroup": {
        "DBClusterParameterGroupName": "mydbclusterparametergroup",
        "DBParameterGroupFamily": "mysql8.0",
        "Description": "My new cluster parameter group",
        "DBClusterParameterGroupArn": "arn:aws:rds:us-east-1:123456789012:cluster-
pg:mydbclusterparametergroup2"
    }
}
```

RDS API

要创建数据库集群参数组，请使用 RDS API `CreateDBClusterParameterGroup` 操作。

包括以下必需参数：

- `DBClusterParameterGroupName`
- `DBParameterGroupFamily`
- `Description`

修改数据库集群参数组中的参数

您可以修改客户创建的数据库集群参数组中的参数值。您无法更改默认数据库集群参数组中的参数值。对客户创建的数据库集群参数组中的参数所做的更改将应用于与此数据库集群参数组关联的所有数据库集群。

控制台

修改数据库集群参数组

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择参数组。
3. 在此列表中，选择要修改的参数组。
4. 对于 Parameter group actions (参数组操作)，选择 Edit (编辑)。
5. 更改要修改的参数的值。您可使用对话框右上方的箭头键滚动参数。

   您无法更改默认参数组中的值。
6. 选择保存更改。
7. 重启集群中的以对其应用更改。

   如果您不重启集群，则失效转移操作需要的时间可能比正常情况下更长。

Amazon CLI

要修改数据库集群参数组，请使用带以下必需参数的 Amazon CLI `modify-db-cluster-parameter-group` 命令：

- `--db-cluster-parameter-group-name`
- `--parameters`

以下示例修改了名为 mydbclusterparametergroup 的数据库集群参数组的 `server_audit_logging` 和 `server_audit_logs_upload` 值。

Example

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-cluster-parameter-group \
    --db-cluster-parameter-group-name mydbclusterparametergroup \
    --parameters
 "ParameterName=server_audit_logging,ParameterValue=1,ApplyMethod=immediate" \

 "ParameterName=server_audit_logs_upload,ParameterValue=1,ApplyMethod=immediate"
```

对于 Windows：

```
aws rds modify-db-cluster-parameter-group ^
    --db-cluster-parameter-group-name mydbclusterparametergroup ^
    --parameters
 "ParameterName=server_audit_logging,ParameterValue=1,ApplyMethod=immediate" ^

 "ParameterName=server_audit_logs_upload,ParameterValue=1,ApplyMethod=immediate"
```

该命令产生类似下面的输出：

```
DBCLUSTERPARAMETERGROUP  mydbclusterparametergroup
```

RDS API

要修改数据库集群参数组，请使用带以下必需参数的 RDS API `ModifyDBClusterParameterGroup` 命令：

- `DBClusterParameterGroupName`
- `Parameters`

重置数据库集群参数组中的参数

您可以在客户创建的数据库集群参数组中将参数重置为其默认值。对客户创建的数据库集群参数组中的参数所做的更改将应用于与此数据库集群参数组关联的所有数据库集群。

> **Note**
> 在默认数据库集群参数组中，参数始终设置为默认值。

控制台

将数据库集群参数组中的参数重置为其默认值

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择参数组。
3. 在列表中，选择参数组。
4. 对于 Parameter group actions (参数组操作)，选择 Edit (编辑)。
5. 选择要重置为默认值的参数。您可使用对话框右上方的箭头键滚动参数。

   您无法重置默认参数组中的值。
6. 选择重置，然后通过选择重置参数进行确认。
7. 重启。

Amazon CLI

要将数据库集群参数组中的参数重置为其默认值，请使用带以下必要选项的 Amazon CLI [`reset-db-cluster-parameter-group`]() 命令：`--db-cluster-parameter-group-name`。

要重置数据库集群参数组中的所有参数，请指定 `--reset-all-parameters` 选项。要重置特定参数，请指定 `--parameters` 选项。

以下示例将名为 mydbparametergroup 的数据库参数组中的所有参数重置为其默认值。

Example

对于 Linux、macOS 或 Unix：

```
aws rds reset-db-cluster-parameter-group \
    --db-cluster-parameter-group-name mydbparametergroup \
    --reset-all-parameters
```

对于 Windows：

```
aws rds reset-db-cluster-parameter-group ^
    --db-cluster-parameter-group-name mydbparametergroup ^
```

```
    --reset-all-parameters
```

以下示例展示了在名为 mydbclusterparametergroup 的数据库集群参数组中将 server_audit_logging 和 server_audit_logs_upload 重置为其默认值。

Example

对于 Linux、macOS 或 Unix：

```
aws rds reset-db-cluster-parameter-group \
    --db-cluster-parameter-group-name mydbclusterparametergroup \
    --parameters "ParameterName=server_audit_logging,ApplyMethod=immediate" \
                 "ParameterName=server_audit_logs_upload,ApplyMethod=immediate"
```

对于 Windows：

```
aws rds reset-db-cluster-parameter-group ^
    --db-cluster-parameter-group-name mydbclusterparametergroup ^
    --parameters
 "ParameterName=server_audit_logging,ParameterValue=1,ApplyMethod=immediate" ^

 "ParameterName=server_audit_logs_upload,ParameterValue=1,ApplyMethod=immediate"
```

该命令产生类似下面的输出：

```
DBClusterParameterGroupName  mydbclusterparametergroup
```

RDS API

要将数据库集群参数组中的参数重置为其默认值，请使用带以下必需参数的 RDS API [ResetDBClusterParameterGroup](#) 命令：DBClusterParameterGroupName。

要重置数据库集群参数组中的所有参数，请将 ResetAllParameters 参数设置为 true。要重置特定参数，请指定 Parameters 参数。

复制数据库集群参数组

您可以复制您创建的自定义数据库集群参数组。当您已创建一个数据库集群参数组并且想在新的数据库集群参数组中包含该组中的大部分自定义参数和值时，复制参数组是一个方便的解决方案。您可通过

使用 Amazon CLI copy-db-cluster-parameter-group 命令或 RDS API CopyDBClusterParameterGroup 操作来复制数据库集群参数组。

复制数据库集群参数组之后，至少等待 5 分钟，然后创建使用该数据库集群参数组的数据库集群。这样，Amazon RDS 就可以在新数据库集群使用参数组之前完全复制此参数组。您可以使用 Amazon RDS 控制台的 Parameter groups（参数组）页面，或者使用 describe-db-cluster-parameters 命令，以验证是否创建了数据库集群参数组。

> **Note**
> 您无法复制默认参数组。不过，您可以创建基于默认参数组的新参数组。
> 您无法将数据库集群参数组复制到其他 Amazon Web Services 账户或 Amazon Web Services 区域。

控制台

复制数据库集群参数组

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择参数组。
3. 在列表中，选择要复制的自定义参数组。
4. 对于 Parameter group actions (参数组操作)，选择 Copy (复制)。
5. 在 New DB parameter group identifier (新数据库参数组标识符) 中，输入新参数组的名称。
6. 在 Description (描述) 中，输入新参数组的描述。
7. 选择 Copy (复制)。

Amazon CLI

要复制数据库集群参数组，请使用 Amazon CLI `copy-db-cluster-parameter-group` 命令及以下必要参数：

- `--source-db-cluster-parameter-group-identifier`
- `--target-db-cluster-parameter-group-identifier`
- `--target-db-cluster-parameter-group-description`

以下示例创建一个名为 mygroup2 的新数据库参集群数组，它是数据库集群参数组 mygroup1 的副本。

Example

对于 Linux、macOS 或 Unix：

```
aws rds copy-db-cluster-parameter-group \
    --source-db-cluster-parameter-group-identifier mygroup1 \
    --target-db-cluster-parameter-group-identifier mygroup2 \
    --target-db-cluster-parameter-group-description "DB parameter group 2"
```

对于 Windows：

```
aws rds copy-db-cluster-parameter-group ^
    --source-db-cluster-parameter-group-identifier mygroup1 ^
    --target-db-cluster-parameter-group-identifier mygroup2 ^
    --target-db-cluster-parameter-group-description "DB parameter group 2"
```

RDS API

要复制数据库集群参数组，请将 RDS API `CopyDBClusterParameterGroup` 操作与下列必需参数配合使用：

- `SourceDBClusterParameterGroupIdentifier`
- `TargetDBClusterParameterGroupIdentifier`
- `TargetDBClusterParameterGroupDescription`

列出数据库集群参数组

您可以列出为 Amazon 账户创建的数据库集群参数组。

> **Note**
> 当您为特定数据库引擎和版本创建数据库集群时，将自动从默认参数模板创建默认参数组。这些默认参数组包含首选参数设置，并且无法修改。当您创建自定义参数组时，可以修改参数设置。

控制台

列出 Amazon 账户的所有数据库集群参数组

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择参数组。

   数据库集群参数组出现在数据库集群参数组的类型列表中。

Amazon CLI

要列出 Amazon 账户的所有数据库集群参数组，请使用 Amazon CLI describe-db-cluster-parameter-groups 命令。

Example

下例列出了 Amazon 账户的所有可用数据库集群参数组。

```
aws rds describe-db-cluster-parameter-groups
```

以下示例描述了 mydbclusterparametergroup 参数组。

对于 Linux、macOS 或 Unix：

```
aws rds describe-db-cluster-parameter-groups \
    --db-cluster-parameter-group-name mydbclusterparametergroup
```

对于 Windows：

```
aws rds describe-db-cluster-parameter-groups ^
    --db-cluster-parameter-group-name mydbclusterparametergroup
```

此命令会返回类似以下内容的响应：

```
{
    "DBClusterParameterGroups": [
        {
            "DBClusterParameterGroupName": "mydbclusterparametergroup2",
```

```
            "DBParameterGroupFamily": "mysql8.0",
            "Description": "My new cluster parameter group",
            "DBClusterParameterGroupArn": "arn:aws:rds:us-east-1:123456789012:cluster-pg:mydbclusterparametergroup"
        }
    ]
}
```


RDS API

要列出 Amazon 账户的所有数据库集群参数组，请使用 RDS API `DescribeDBClusterParameterGroups` 操作。

查看数据库集群参数组的参数值

您可获得数据库集群参数组内所有参数的列表及它们的值。

控制台

查看数据库集群参数组的参数值

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择参数组。

   数据库集群参数组出现在数据库集群参数组的类型列表中。
3. 选择数据库集群参数组的名称以查看其参数列表。

Amazon CLI

要查看数据库集群参数组的参数值，请使用 Amazon CLI `describe-db-cluster-parameters` 命令及以下必要参数。

- `--db-cluster-parameter-group-name`

Example

以下示例以 JSON 格式列出名为 mydbclusterparametergroup 的数据库集群参数组的参数和参数值。

此命令会返回类似以下内容的响应：

```
aws rds describe-db-cluster-parameters --db-cluster-parameter-group-
name mydbclusterparametergroup
```


```
{
    "Parameters": [
        {
            "ParameterName": "activate_all_roles_on_login",
            "ParameterValue": "0",
            "Description": "Automatically set all granted roles as active after the
 user has authenticated successfully.",
            "Source": "engine-default",
            "ApplyType": "dynamic",
            "DataType": "boolean",
            "AllowedValues": "0,1",
            "IsModifiable": true,
            "ApplyMethod": "pending-reboot",
            "SupportedEngineModes": [
                "provisioned"
            ]
        },
        {
            "ParameterName": "allow-suspicious-udfs",
            "Description": "Controls whether user-defined functions that have only an
 xxx symbol for the main function can be loaded",
            "Source": "engine-default",
            "ApplyType": "static",
            "DataType": "boolean",
            "AllowedValues": "0,1",
            "IsModifiable": false,
            "ApplyMethod": "pending-reboot",
            "SupportedEngineModes": [
                "provisioned"
            ]
        },
...
```


RDS API

要查看数据库集群参数组的参数值，请使用用带下列所需参数的 RDS API DescribeDBClusterParameters 命令。

- `DBClusterParameterGroupName`

在某些情况下，不显示参数的允许值。这些始终是源为数据库引擎原定设置值的参数。

要查看这些参数的值，可以运行以下 SQL 语句：

- MySQL：

```
-- Show the value of a particular parameter
mysql$ SHOW VARIABLES LIKE '%parameter_name%';

-- Show the values of all parameters
mysql$ SHOW VARIABLES;
```

- PostgreSQL：

```
-- Show the value of a particular parameter
postgresql=> SHOW parameter_name;

-- Show the values of all parameters
postgresql=> SHOW ALL;
```

删除数据库集群参数组

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 删除数据库集群参数组。仅当数据库集群参数组未与数据库集群关联时，才能将其删除。

控制台

删除参数组

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择参数组。

   参数组显示在列表中。
3. 选择要删除的数据库集群参数组的名称。
4. 选择操作，然后选择删除。
5. 查看参数组名称，然后选择删除。

Amazon CLI

要删除数据库集群参数组，请使用带以下必需参数的 Amazon CLI `delete-db-cluster-parameter-group` 命令。

- `--db-parameter-group-name`

Example

以下示例删除了名为 mydbparametergroup 的数据库集群参数组。

```
aws rds delete-db-cluster-parameter-group --db-parameter-group-name mydbparametergroup
```

RDS API

要删除数据库集群参数组，请使用带以下必需参数的 RDS API `DeleteDBClusterParameterGroup` 命令。

- `DBParameterGroupName`

比较数据库参数组

您可以使用 Amazon Web Services Management Console查看两个数据库参数组之间的区别。

指定的参数组必须都是数据库参数组，或者必须都是数据库集群参数组。甚至当数据库引擎和版本相同时，也是如此。例如，您无法比较 `aurora-mysql8.0`（Aurora MySQL 版本 3）数据库参数组和 `aurora-mysql8.0` 数据库集群参数组。

您可以比较 Aurora MySQL 和 RDS for MySQL 数据库参数组（即使是不同版本），但您无法不能比较 Aurora PostgreSQL 和 RDS for PostgreSQL 数据库参数组。

比较两个数据库参数组

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择参数组。
3. 在此列表中，选择要比较的两个参数组。

> **Note**
> 要将默认参数组与自定义参数组进行比较，首先请在默认选项卡上选择默认参数组，然后在自定义选项卡上选择自定义参数组。

4. 从操作中选择比较。

指定数据库参数

数据库参数类型包括：

- 整数
- Boolean
- String
- 长整型
- Double
- Timestamp
- 其他定义的数据类型的对象
- 整数、布尔值、字符串、long、double、时间戳或对象类型的值的数组

您还可以使用表达式、公式和函数指定整数和布尔参数。

对于 Oracle 引擎，您可以使用 `DBInstanceClassHugePagesDefault` 公式变量指定布尔值数据库参数。请参阅 数据库参数公式变量。

对于 PostgreSQL 引擎，您可以使用表达式指定布尔值数据库参数。请参阅 布尔值数据库参数表达式。

目录

- 数据库参数日志表达式
- 数据库参数值示例

数据库参数公式

数据库参数公式是一种可解析为整数值或布尔值的表达式。可以将表达式用大括号括起来：{}。可以为数据库参数值使用公式，也可以将公式用作数据库参数函数的参数。

语法

```
{FormulaVariable}
{FormulaVariable*Integer}
{FormulaVariable*Integer/Integer}
{FormulaVariable/Integer}
```

数据库参数公式变量

每个公式变量会返回一个整数或布尔值。变量的名称是区分大小写的。

AllocatedStorage

返回一个整数，它表示数据卷的大小（以字节为单位）。

DBInstanceClassHugePagesDefault

返回一个布尔值。当前，仅 Oracle 引擎支持它。

有关更多信息，请参阅“为 RDS for Oracle 实例开启大页”。

DBInstanceClassMemory

该变量会返回一个整数，表示数据库进程可用的内存字节数。这个数字是以数据库实例类的内存总量开始在内部计算得出的。从此处，计算结果将减去为操作系统以及管理实例的 RDS 进程预留的内存。因此，该数字总是略低于 数据库实例类 中实例类表所示的内存数字。确切的值取决于多种因素的组合。其中包括实例类、数据库引擎，以及其适用于 RDS 实例还是适用于属于 Aurora 集群一部分的实例。

DBInstanceVCPU

返回一个整数，它表示 Amazon RDS 用于管理实例的虚拟中央处理器 (vCPU) 的数量。

EndPointPort

返回一个整数，它表示连接到数据库实例时使用的端口。

TrueIfReplica

如果数据库实例是只读副本，返回 1，如果不是只读副本，则返回 0。这是 MySQL 中 `read_only` 参数的默认值。

数据库参数公式运算符

数据库参数公式支持两个运算符：除法和乘法。

除法运算符：/

用除数除以被除数，返回整数型商。商中的小数不四舍五入，直接截断。

语法

```
dividend / divisor
```

被除数和除数参数必须是整数型表达式。

乘法运算符：*

将表达式乘以表达式，返回表达式的乘积。表达式中的小数不四舍五入，直接截断。

语法

```
expression * expression
```

两个表达式必须都是整数型。

数据库参数函数

您可以将数据库参数函数的参数指定为整数或公式。每个函数必须具有至少一个参数。将多个参数指定为逗号分隔的列表。列表不能拥有任何空成员，如 argument1、argument3。函数名称区分大小写。

IF

返回一个参数。

当前，只有 Oracle 引擎支持它，并且唯一支持的第一个参数是 `{DBInstanceClassHugePagesDefault}`。有关更多信息，请参阅"为 RDS for Oracle 实例开启大页"。

语法

```
IF(argument1, argument2, argument3)
```

如果第一个参数的计算结果为 true，则返回第二个参数。否则返回第三个参数。

GREATEST

返回整数型或者参数公式列表中最大的值。

语法

```
GREATEST(argument1, argument2,...argumentn)
```

返回整数。

LEAST

返回整数型或者参数公式列表中最小的值。

语法

```
LEAST(argument1, argument2,...argumentn)
```

返回整数。

SUM

添加指定整数型或者参数公式的值。

语法

```
SUM(argument1, argument2,...argumentn)
```

返回整数。

布尔值数据库参数表达式

布尔值数据库参数表达式解析为布尔值 1 或 0。引号中的内容是表达式。

> **Note**
>
> 仅 PostgreSQL 引擎支持布尔值数据库参数表达式。

语法

```
"expression operator expression"
```

两个表达式都必须解析为整数。表达式可以是：

- 整型常量
- 数据库参数公式
- 数据库参数函数
- 数据库参数变量

布尔值数据库参数表达式支持以下不等式运算符：

大于运算符：>

语法

```
"expression > expression"
```

小于运算符：<

语法

```
"expression < expression"
```

大于或等于运算符：>=、=>

语法

```
"expression >= expression"
"expression => expression"
```

小于或等于运算符:<=、=<

语法

```
"expression <= expression"
"expression =< expression"
```

Example 使用布尔值数据库参数表达式

以下布尔数据库参数表达式示例将参数公式的结果与整数进行比较。这样做是为了修改 PostgreSQL 数据库实例的布尔数据库参数 `wal_compression`。参数表达式将 vCPU 的数量与值 2 进行比较。如果 vCPU 的数量大于 2，`wal_compression` 数据库参数则会设置为 true。

```
aws rds modify-db-parameter-group --db-parameter-group-name group-name \
--parameters "ParameterName=wal_compression,ParameterValue=\"{DBInstanceVCPU} > 2\" "
```

数据库参数日志表达式

您可以为日志表达式设置整数数据库参数值。可以将表达式用大括号括起来：{}。例如：

```
{log(DBInstanceClassMemory/8187281418)*1000}
```

`log` 函数表示日志基数 2。此示例还使用了 `DBInstanceClassMemory` 公式变量。请参阅 数据库参数公式变量。

> **Note**
> 目前，您只能将 MySQL `innodb_log_file_size` 参数指定为一个整数。

数据库参数值示例

这些示例展示了如何使用公式、函数和表达式来表达数据库参数的值。

> **Warning**
> 在数据库参数组内设置参数不当可能会产生意外的不利影响。可能包括性能下降和系统不稳定。修改数据库参数时应保持谨慎，且修改数据库参数组前要备份数据。将参数组更改应用到生产数据库实例前，请在使用时间点还原创建的测试数据库实例上试用这些参数组更改。

Example 使用数据库参数函数 GREATEST

您可以指定 Oracle 进程参数中的 `GREATEST` 函数。使用它可以将用户进程数设置为 80 或 `DBInstanceClassMemory`/9868951 中的较大值。

```
GREATEST({DBInstanceClassMemory/9868951},80)
```

Example 使用数据库参数函数 LEAST

您可以指定 MySQL `LEAST` 参数值中的 `max_binlog_cache_size` 函数。使用它将事务可以在 MySQL 实例中使用的最大缓存大小设置为 1 MB 或`DBInstanceClass`/256 中的较小值。

```
LEAST({DBInstanceClassMemory/256},10485760)
```

使用 Amazon RDS 数据库实例设置创建 Amazon ElastiCache 缓存

ElastiCache 是一项完全托管式内存缓存服务，提供的微秒级读写延迟可支持灵活的实时使用案例。ElastiCache 可以帮助提高应用程序和数据库的性能。您可以将 ElastiCache 用作不需要数据持久性的使用案例的主要数据存储，例如游戏排行榜、流媒体和数据分析。ElastiCache 可以帮助消除与部署和管理分布式计算环境相关的复杂性。有关更多信息，请参阅适用于 Memcached 的 Common ElastiCache Use Cases and How ElastiCache Can Help 和适用于 Redis OSS 的 Common ElastiCache Use Cases and How ElastiCache Can Help。您可以使用 Amazon RDS 控制台创建 ElastiCache 缓存。

您可以通过两种方式来使用 Amazon ElastiCache。您可以从无服务器缓存开始入手，也可以选择设计自己的缓存群集。如果您选择设计自己的缓存集群，ElastiCache 将同时使用 Redis OSS 和 Memcached 引擎。如果您不确定要使用哪个引擎，请参阅 Comparing Memcached and Redis OSS。有关 Amazon ElastiCache 的更多信息，请参阅 Amazon ElastiCache 用户指南。

主题

使用 RDS 数据库实例设置创建 ElastiCache 缓存的概述

您可以使用与新创建的或现有的 RDS 数据库实例相同的配置设置，从 Amazon RDS 创建 ElastiCache 缓存。

将 ElastiCache 缓存与您的数据库实例关联的一些使用案例：

- 与单独在 RDS 上运行相比，将 ElastiCache 与 RDS 结合使用可以节省成本并提高性能。

  例如，将 ElastiCache 和 RDS for MySQL 结合使用，与单独使用 RDS for MySQL 相比，可以节省多达 55% 的成本并将读取性能提高多达 80 倍。
- 您可以使用 ElastiCache 缓存作为不需要数据持久性的应用程序的主数据存储。使用 Redis OSS 或 Memcached 的应用程序几乎不需要任何修改就能使用 ElastiCache。

当您从 RDS 创建 ElastiCache 缓存时，ElastiCache 缓存会从关联的 RDS 数据库实例继承以下设置：

- ElastiCache 连接设置
- ElastiCache 安全设置

您也可以根据要求设置缓存配置设置。

在应用程序中设置 ElastiCache

您的应用程序必须设置为使用 ElastiCache 缓存。还可以根据要求将应用程序设置为使用缓存策略，从而优化和提高缓存性能。

- 要访问并开始使用 ElastiCache 缓存，请参阅 Getting started with ElastiCache (Redis OSS) 和 Getting started with ElastiCache (Memcached)。
- 有关缓存策略的更多信息，请参阅适用于 Memcached 的 Caching strategies and best practices 和适用于 Redis 的 Caching strategies and best practices。
- 有关 ElastiCache（Redis OSS）集群中的高可用性的更多信息，请参阅 High availability using replication groups。
- 您可能会产生与备份存储、区域内或跨区域数据传输或使用 Amazon Outposts 相关的成本。有关定价详细信息，请参阅 Amazon ElastiCache 定价。

使用 RDS 数据库实例中的设置创建 ElastiCache 缓存

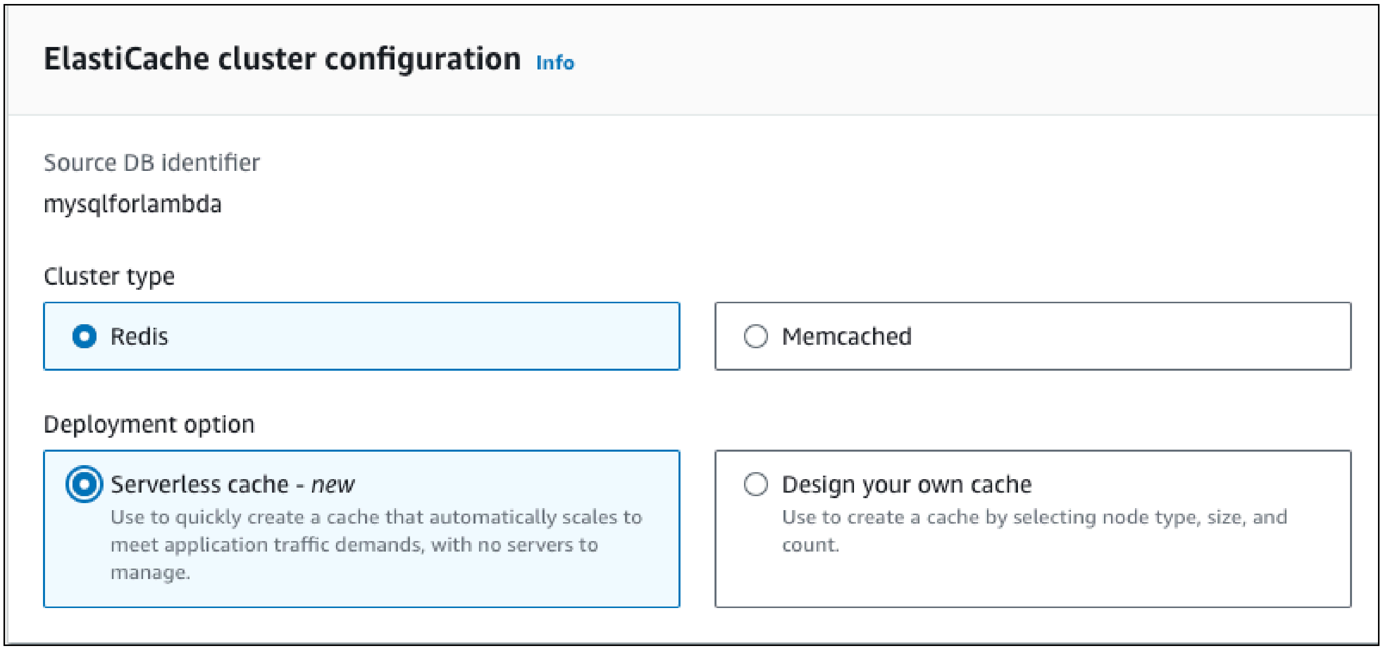
您可以使用继承自数据库实例的设置为 RDS 数据库实例创建 ElastiCache 缓存。

使用数据库实例中的设置创建 ElastiCache 缓存

1. 要创建数据库实例，请按照创建 Amazon RDS 数据库实例中的说明操作。
2. 创建 RDS 数据库实例后，控制台将显示建议的插件窗口。选择使用您的数据库设置从 RDS 创建 ElastiCache 集群。

   对于现有数据库，在数据库页面中，选择所需的数据库实例。在操作下拉菜单中，选择创建 ElastiCache 集群，以在 RDS 中创建与现有 RDS 数据库实例具有相同设置的 ElastiCache 缓存。

   在 ElastiCache 配置部分，源数据库标识符显示 ElastiCache 缓存从哪个数据库实例继承设置。
3. 选择要创建 Redis OSS 集群还是 Memcached 集群。有关更多信息，请参阅 Comparing Memcached and Redis OSS。



4. 之后，选择是要创建无服务器缓存还是设计自己的缓存。有关更多信息，请参阅选择部署选项。

   如果您选择无服务器缓存：

   a. 在缓存设置中，输入名称和描述的值。
   b. 在查看默认设置下，保留默认设置以在缓存和数据库实例之间建立连接。
   c. 您也可以通过选择自定义默认设置来编辑默认设置。选择 ElastiCache 连接设置、ElastiCache 安全设置和最大使用限制。

5. 如果您选择设计自己的缓存：

   a. 如果您选择 Redis OSS 集群，请选择要将集群模式保持为启用还是禁用。有关更多信息，请参阅 Replication: Redis OSS (Cluster Mode Disabled) vs. Redis OSS (Cluster Mode Enabled)。
   b. 输入名称、描述和引擎版本的值。

      对于引擎版本，建议的原定设置值是最新的引擎版本。您也可以为 ElastiCache 缓存选择最符合您要求的引擎版本。
   c. 在节点类型选项中选择节点类型。有关更多信息，请参阅管理节点。

      如果您选择在集群模式设置为启用的情况下创建 Redis OSS 集群，请在分片数量选项中输入分片数（分区/节点组）。

      在副本数量中输入每个分片的副本数量。

> **Note**
>
> 所选节点类型、分片数量和副本数量都会影响您的缓存性能和资源成本。确保这些设置符合您的数据库需求。有关定价信息，请参阅 Amazon ElastiCache 定价。

   d. 选择 ElastiCache 连接设置和 ElastiCache 安全设置。您可以保留默认设置或根据需要自定义这些设置。

6. 验证 ElastiCache 缓存的默认设置和继承设置。某些设置在创建后无法更改。

> **Note**
>
> RDS 可能会调整您的 ElastiCache 缓存的备份时段，以满足 60 分钟的最低时段要求。源数据库的备份时段保持不变。

7. 准备就绪后，请选择创建 ElastiCache 缓存。

控制台显示创建 ElastiCache 缓存的确认横幅。点击横幅中指向 ElastiCache 控制台的链接可查看缓存的详细信息。ElastiCache 控制台显示新创建的 ElastiCache 缓存。

使用 Amazon Database Migration Service 将 EC2 数据库自动迁移到 Amazon RDS

您可以使用 RDS 控制台将 EC2 数据库迁移到 RDS。RDS 使用 Amazon Database Migration Service（Amazon DMS）迁移您的源 EC2 数据库。Amazon DMS 允许您将关系数据库迁移到 Amazon 云中。有关 Amazon Database Migration Service 的更多信息，请参阅《Amazon Database Migration Service 用户指南》中的[什么是 Amazon Database Migration Service？](#)。

要开始迁移，您必须创建等效的 RDS 数据库实例，以将数据迁移到其中。创建目标数据库后，您可以将 EC2 数据库导入其中。对于小于 1 TiB 的源数据库，此迁移操作可减少将数据迁移到 RDS 所需的时间和资源。

概览

RDS 控制台允许您将 EC2 数据库迁移到等效的 RDS 数据库中。您必须创建 RDS 数据库，才能从控制台进行迁移。

您可以为以下数据库引擎迁移 EC2 数据库：

- MySQL
- MariaDB
- PostgreSQL

该迁移过程涉及到以下步骤：

- 在 RDS 中创建等效的数据库。为了使这些数据库具有等效性，它们必须具有相同的数据库引擎和兼容的引擎版本。它们还必须位于同一 VPC 中。有关创建数据库的说明，请参阅 [创建 Amazon RDS 数据库实例](#)。
- 选择数据库的复制类型：
  - 完全加载迁移 – RDS 将完整的源数据库复制到目标数据库，并在必要时在目标中创建新表。

> **Note**
>
> 此选项会导致您的 RDS 数据库中断。

  - 完全加载和更改数据捕获（CDC）迁移 – 与完全加载迁移类似，使用此选项，RDS 会将完整的源数据库复制到目标数据库。但是，在完全加载迁移之后，RDS 会将源中捕获的任何更改应用于目标数据库。更改数据捕获通过使用数据库引擎的本机 API 来收集对数据库日志的更改。

    > **Note**
    >
    > 此选项会导致您的 RDS 数据库中断。

  - 更改数据捕获（CDC）– 使用此选项可使您的目标数据库在迁移过程中保持可用。RDS 会将源数据库中正在进行的更改迁移到目标数据库。

- RDS 会创建必要的联网资源来促进迁移。RDS 在创建所需资源后，会通知您已创建的资源并允许您启动数据传输。

  完成迁移所需的时间取决于复制的类型和源数据库的大小。

先决条件

MySQL 和 MariaDB

在开始将 MySQL 或 MariaDB 数据库用作源数据库之前，请确保满足以下先决条件。这些先决条件适用于 Amazon 托管的源。

您必须拥有具有复制管理员角色的 Amazon DMS 的账户。该角色需要以下权限：

- REPLICATION CLIENT – 仅 CDC 任务需要此权限。换而言之，full-load-only 任务不需要此权限。
- REPLICATION SLAVE – 仅 CDC 任务需要此权限。换而言之，full-load-only 任务不需要此权限。

Amazon DMS 用户还必须具有为复制指定的源表的 SELECT 权限。

如果您使用 MySQL 特定的迁移前评测功能，请授予以下权限。

```
grant select on mysql.user to <dms_user>;
grant select on mysql.db to <dms_user>;
grant select on mysql.tables_priv to <dms_user>;
grant select on mysql.role_edges to <dms_user>  #only for MySQL version 8.0.11 and
 higher
```

PostgreSQL

在从 Amazon 托管的 PostgreSQL 源数据库迁移数据之前，请执行以下操作：

- 我们建议您使用具有 PostgreSQL 数据库实例所需最低权限的 Amazon 用户账户作为 Amazon DMS 的 PostgreSQL 源端点的用户账户。建议不要使用主账户。该账户必须具有 `rds_superuser` 和 `rds_replication` 角色。`rds_replication` 角色可授予管理逻辑时隙和使用逻辑时隙流式处理数据的权限。

> **Note**
> 一些 Amazon DMS 事务在被 DMS 引擎再次使用之前会空闲一段时间。在 PostgreSQL 版本 9.6 及更高版本中使用参数 `idle_in_transaction_session_timeout`，能够导致空闲事务超时并失败。

限制

以下限制适用于自动迁移过程：

- 您的目标数据库状态必须为可用才能开始源数据库迁移。
- 当从 MySQL 源数据库迁移时，您的 RDS 账户必须具有复制管理员角色。您还必须为该角色应用适当的权限。
- EC2 实例和目标数据库必须位于同一 VPC 中。
- 使用从 EC2 数据库迁移数据操作时，您无法将 EC2 数据库迁移到以下目标数据库：
  - 作为集群成员的数据库
  - Oracle、SQL Server 和 Db2 数据库
  - MySQL 版本低于 5.7 的数据库
  - PostgreSQL 版本低于 10.4 的数据库
  - MariaDB 版本低于 10.2 的数据库

为同构迁移创建 IAM 资源

RDS 使用 Amazon DMS 迁移数据。为了访问数据库并迁移数据，Amazon DMS 会为同构数据迁移创建无服务器环境。在此环境中，Amazon DMS 需要访问 VPC 对等连接、路由表、安全组和

其他 Amazon 资源。此外，Amazon DMS 将每个数据迁移的日志、指标和进度存储在 Amazon CloudWatch 中。要创建数据迁移项目，Amazon DMS 需要访问这些服务。

此外，Amazon DMS 需要访问代表一组用户凭证的密钥，以验证源和目标连接的数据库连接。

> **Note**
>
> 通过使用从 EC2 实例迁移数据操作，您可以使用 RDS 控制台生成这些 IAM 资源。如果您使用控制台生成的 IAM 资源，则跳过此步骤。

此过程需要以下 IAM 资源：

主题

为同构数据迁移创建 IAM 策略

在此步骤中，创建 IAM policy，向 Amazon DMS 提供 Amazon EC2 和 CloudWatch 资源的访问权限。接下来，创建 IAM 角色并附加此策略。

为数据迁移创建 IAM 策略

1. 登录到 Amazon Web Services Management Console，然后通过以下网址打开 IAM 控制台：https://console.aws.amazon.com/iam/。
2. 在导航窗格中，选择策略。
3. 选择创建策略。
4. 在创建策略页面中，选择 JSON 选项卡。
5. 将下面的 JSON 粘贴到编辑器中。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Action": [
                "ec2:DescribeRouteTables",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeVpcPeeringConnections",
                "ec2:DescribeVpcs",
                "ec2:DescribePrefixLists",
                "logs:DescribeLogGroups"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "servicequotas:GetServiceQuota"
            ],
            "Resource": "arn:aws:servicequotas:*:*:vpc/L-0EA8095F"
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:DescribeLogStreams"
            ],
            "Resource": "arn:aws:logs:*:*:log-group:dms-data-migration-*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "arn:aws:logs:*:*:log-group:dms-data-migration-*:log-
stream:dms-data-migration-*"
        },
        {
            "Effect": "Allow",
            "Action": "cloudwatch:PutMetricData",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateRoute",
                "ec2:DeleteRoute"
```

```
            ],
            "Resource": "arn:aws:ec2:*:*:route-table/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:security-group/*",
                "arn:aws:ec2:*:*:security-group-rule/*",
                "arn:aws:ec2:*:*:route-table/*",
                "arn:aws:ec2:*:*:vpc-peering-connection/*",
                "arn:aws:ec2:*:*:vpc/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AuthorizeSecurityGroupEgress",
                "ec2:AuthorizeSecurityGroupIngress"
            ],
            "Resource": "arn:aws:ec2:*:*:security-group-rule/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AuthorizeSecurityGroupEgress",
                "ec2:AuthorizeSecurityGroupIngress",
                "ec2:RevokeSecurityGroupEgress",
                "ec2:RevokeSecurityGroupIngress"
            ],
            "Resource": "arn:aws:ec2:*:*:security-group/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AcceptVpcPeeringConnection",
                "ec2:ModifyVpcPeeringConnectionOptions"
            ],
            "Resource": "arn:aws:ec2:*:*:vpc-peering-connection/*"
        },
        {
            "Effect": "Allow",
```

```
                "Action": "ec2:AcceptVpcPeeringConnection",
                "Resource": "arn:aws:ec2:*:*:vpc/*"
            }
        ]
    }
```

6. 选择下一步：标签和下一步：审核。
7. 在名称*中输入 **HomogeneousDataMigrationsPolicy**，然后选择创建策略。

为同构数据迁移创建 IAM 角色

在此步骤中，您将创建 IAM 角色，该角色提供对 Amazon Secrets Manager、Amazon EC2 和 CloudWatch 的访问权限。

为数据迁移创建 IAM 角色

1. 登录 Amazon Web Services Management Console，然后通过以下网址打开 IAM 控制台：https://console.aws.amazon.com/iam/。
2. 在导航窗格中，选择角色。
3. 选择创建角色。
4. 在选择可信实体页面中，在可信实体类型下选择 Amazon 服务。对于其他 Amazon 服务的应用场景，选择 DMS。
5. 选中 DMS 复选框，然后选择下一步。
6. 在添加权限页面上，选择之前创建的 HomogeneousDataMigrationsPolicy。选择下一步。
7. 在命名、检查并创建页面上，在角色名称中输入 **HomogeneousDataMigrationsRole**，然后选择创建角色。
8. 在角色页面上，在角色名称中输入 **HomogeneousDataMigrationsRole**。选择 HomogeneousDataMigrationsRole。
9. 在 HomogeneousDataMigrationsRole 页面上，选择信任关系选项卡。选择编辑信任策略。
10. 在编辑信任策略页面上，将以下 JSON 粘贴到编辑器中，替换现有文本。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
```

```
                "Principal": {
                    "Service": [
                        "dms-data-migrations.amazonaws.com",
                        "dms.your_region.amazonaws.com"
                    ]
                },
                "Action": "sts:AssumeRole"
            }
        ]
}
```

在以上示例中，请用您的 Amazon Web Services 区域名称替换 *your_region*。

上述基于资源的策略为 Amazon DMS 服务主体提供了根据客户管理型 HomogeneousDataMigrationsPolicy 策略执行任务的权限。

11. 选择更新策略。

创建密钥访问策略和角色

按照以下步骤创建您的密钥访问策略和角色，允许 DMS 访问源数据库和目标数据库的用户凭证。

创建密钥访问策略和角色，以允许 Amazon RDS 访问 Amazon Secrets Manager，从而访问相应的密钥

1. 登录到 Amazon Web Services Management Console，然后通过以下网址打开 Amazon Identity and Access Management（IAM）控制台：https://console.aws.amazon.com/iam/。
2. 选择策略，然后选择创建策略。
3. 选择 JSON 并输入以下策略，以启用对密钥的访问和解密。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "secretsmanager:GetSecretValue",
            "Resource": secret_arn,
        },
        {
             "Effect": "Allow",
             "Action": [
```

```
                        "kms:Decrypt",
                        "kms:DescribeKey"
                      ],
              "Resource": kms_key_arn,
          }
      ]
}
```

此处，*secret_arn* 是密钥的 ARN，您可以根据需要从任一 SecretsManagerSecretId 中获取，*kms_key_arn* 是您用于加密密钥的 Amazon KMS 密钥的 ARN，如以下示例所示。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "arn:aws:secretsmanager:us-
east-2:123456789012:secret:MySQLTestSecret-qeHamH"
        },
        {
             "Effect": "Allow",
             "Action": [
                        "kms:Decrypt",
                        "kms:DescribeKey"
                      ],
             "Resource": "arn:aws:kms:us-
east-2:123456789012:key/761138dc-0542-4e58-947f-4a3a8458d0fd"
        }
    ]
}
```

> **Note**
>
> 如果您使用 Amazon Secrets Manager 创建的默认加密密钥，则无需为 *kms_key_arn* 指定 Amazon KMS 权限。
>
> 如果您希望策略提供对两个密钥的访问权限，只需为另一个 *secret_arn* 指定一个额外的 JSON 资源对象。

4. 查看并创建具友好名称和描述（可选）的策略。
5. 选择角色，然后选择创建角色。

6. 选择 Amazon 服务作为可信实体的类型。
7. 从服务列表中选择 DMS 作为可信服务，然后选择下一步：权限。
8. 查找并附加您在步骤 4 中创建的策略，然后继续添加所有标签并查看角色。此刻，编辑角色的信任关系，以使用 Amazon RDS 区域服务主体作为可信实体。此主体采用以下格式。

```
dms.region-name.amazonaws.com
```

此处，*region-name* 是区域的名称，如 `us-east-1`。因此，此区域的 Amazon RDS 区域服务主体如下。

```
dms.us-east-1.amazonaws.com
dms-data-migrations.amazonaws.com
```

为 Amazon DMS 创建 IAM 角色以管理 Amazon VPC

您必须为 Amazon DMS 创建 IAM 角色，才能管理资源的 VPC 设置。要成功迁移，此角色必须可用。

为数据库迁移创建 **dms-vpc-role**

<result>

这样将为 DMS 创建角色来管理用于迁移的 VPC 设置。
</result>

1. 登录 Amazon Web Services Management Console，然后打开 IAM 控制台：https://console.aws.amazon.com/iam/。
2. 在控制台的导航窗格中，选择 Roles，然后选择 Create role。
3. 对于选择可信实体选项，选择 Amazon Web Services 服务选项。

   对于使用案例，选择 DMS。
4. 对于添加权限步骤，选择 `AmazonDMSVPCManagementRole` 并选择下一步。
5. 在命名、查看和创建页面中，将角色名称设置为 `dms-vpc-role`，并选择创建角色。

为 EC2 数据库设置数据迁移

要从 EC2 源数据库开始迁移数据，您必须创建一个等效的 RDS 数据库。有关创建数据库的说明，请参阅 [创建 Amazon RDS 数据库实例](#)。

创建目标数据库后，使用以下步骤设置数据迁移：

设置数据迁移项目

1. 在 RDS 控制台的数据库页面上选择目标数据库。
2. 选择操作下拉列表并选择从 EC2 数据库迁移数据选项。要查看支持的目标数据库，请参阅 限制。
3. 在选择源 EC2 数据库部分下：

    1. 检查引擎类型并确保它与源数据库相同。

        另外，请检查引擎版本是否兼容。
    2. 对于 EC2 实例，请选择您的源数据库所驻留的 EC2 实例。
    3. 在端口中，输入源数据库允许流量的端口。
    4. 对于密钥，如果您没有现有密钥，请选择创建和使用新密钥。为您的源数据库输入用户名和密码。此外，选择要用于加密该密钥的 KMS 密钥。

        如果您有现有密钥，请选择使用现有密钥，然后从下拉列表中选择密钥。
    5. 对于密钥的 IAM 角色，如果您有现有的 IAM 角色，请选择使用现有的 IAM 角色，然后从下拉列表中选择一个可以访问上一步中的密钥 ID 的 IAM 角色。

        如果您没有现有的 IAM 角色，请选择创建和使用新的 IAM 角色。为 IAM 角色名称输入您的角色的新名称。您可以在下面的链接中查看与此角色关联的权限。
4. 在查看目标 RDS 数据库部分下：

    1. 在该部分顶部，确认目标数据库的设置。
    2. 对于密钥，如果您没有保存目标数据库凭证的现有密钥，请选择创建和使用新密钥。

        如果您有现有密钥，请从下拉列表中选择该密钥。
    3. 对于密钥的 IAM 角色，选择可以访问上一步中密钥的 IAM 角色。如果您没有现有 IAM 角色，还可以创建一个新 IAM 角色。

        如果下拉列表未填充 IAM 角色，请以 `arn:aws:iam:`*`account_id`*`:role/`*`roleName`* 格式指定 IAM 角色 ARN。
5. 在配置数据迁移部分下：

    1. 从完全加载、完全加载和更改数据捕获（CDC）或更改数据捕获（CDC）中选择数据迁移的类型。有关这些选项的详细信息，请参阅概览。

迁移开始后，您无法修改迁移类型。

2. 对于数据迁移的 IAM 角色，如果您有现有的 IAM 角色，请选择使用现有的 IAM 角色，然后从下拉列表中选择一个 IAM 角色，该角色将授予 DMS 创建迁移所需资源的权限。如果您没有现有的 IAM 角色，请选择创建和使用新的 IAM 角色。

6. 确认查看迁移设置选项卡显示了成功设置数据迁移所需的设置。

7. 选择迁移以完成迁移设置。

完成这些步骤后，您可以通过在控制台的进度横幅中选择查看详细信息来查看正在为数据迁移设置的资源。设置好所需资源后，迁移将自动开始。如果您创建

要将多个数据库迁移到目标数据库，请使用有关新 EC2 数据库的详细信息重新开始此过程。

管理数据迁移

从 RDS 控制台使用从 EC2 数据库迁移数据操作后，RDS 会自动启动迁移。

如果您使用 Amazon DMS 控制台创建迁移资源，则可以启动迁移过程。

启动数据迁移

请按照以下步骤启动数据迁移：

启动数据迁移

1. 在 RDS 控制台的数据库页面上选择目标数据库。
2. 在数据库详细信息页面上，选择数据迁移选项卡。
3. 在数据迁移选项卡下，关联的数据迁移列出了可用的数据迁移。

   设置所需资源后，使用 RDS 控制台设置的迁移会自动启动。

   使用 DMS 控制台设置的迁移将设置为就绪。

   要开始这些迁移，请选择操作下拉列表并选择启动。
4. 这样将开始 EC2 数据库的数据迁移。

停止数据迁移

对于复制类型为完全加载的数据迁移，停止迁移会导致迁移过程停止且无法恢复。停止迁移后，必须重新启动迁移。

对于复制类型设置为更改数据捕获（CDC）或完全加载和 CDC 的迁移，您可以停止持续复制过程，稍后再恢复该过程。

停止数据迁移

1. 在 RDS 控制台的数据库页面上选择目标数据库。
2. 在数据库详细信息页面上，选择数据迁移选项卡。
3. 在数据迁移选项卡下，关联的数据迁移列出了正在进行的数据迁移。

   要停止迁移，请选择数据迁移，然后在操作下拉列表中选择停止。
4. 这样将停止 EC2 数据库的数据迁移。

恢复数据迁移

对于复制类型为完全加载和更改数据捕获（CDC）或更改数据捕获（CDC）迁移的数据迁移，您可以从最后一个停止点恢复 CDC 流程。

恢复数据迁移

1. 在 RDS 控制台的数据库页面上选择目标数据库。
2. 在数据库详细信息页面上，选择数据迁移选项卡。
3. 在数据迁移选项卡下，关联的数据迁移列出了已停止的数据迁移。

   要恢复迁移，请选择数据迁移，然后在操作下拉列表中选择恢复处理。
4. 这样将恢复 EC2 数据库的数据迁移。

删除数据迁移

要删除关联的数据迁移，请使用以下说明：

删除数据迁移

1. 在 RDS 控制台的数据库页面上选择目标数据库。

2. 在数据库详细信息页面上，选择数据迁移选项卡。
3. 要删除迁移，请选择数据迁移，然后在操作下拉列表中选择删除。
4. 这样将删除相应的数据迁移。

删除正在进行的数据迁移不会影响已加载到目标数据库的任何数据。

重新启动数据迁移

要从 CDC 起点重新启动关联的数据迁移，请按照以下说明进行操作。

重新启动数据迁移

1. 在 RDS 控制台的数据库页面上选择目标数据库。
2. 在数据库详细信息页面上，选择数据迁移选项卡。
3. 要重新启动迁移，请选择数据迁移，然后在操作下拉列表中选择重新启动。
4. 这样将从 CDC 起点重新启动数据迁移。

重新启动正在进行的数据迁移不会影响已加载到目标数据库的任何数据。

监控数据迁移

数据迁移开始后，您可以监控其状态和进度。大型数据集的数据迁移需要数小时才能完成。为了保持数据迁移的可靠性、可用性和高性能，请定期监控迁移进度。

检查数据迁移的状态和进度

1. 在 RDS 控制台的数据库页面上选择目标数据库。
2. 在数据库详细信息页面上，选择数据迁移选项卡。
3. 关联的数据迁移部分列出了您的数据迁移。检查状态列。
4. 对于正在运行的数据迁移，迁移过程列显示了已迁移数据的百分比。
5. 要在 CloudWatch 中监控进程，请使用 CloudWatch 列中的链接。

迁移状态

对于您运行的每个数据迁移，RDS 控制台都会显示状态。以下列表包括状态：

- Ready：数据迁移已准备好开始。

- `Starting`：RDS 正在为您的数据迁移创建无服务器环境。
- `Load running`：RDS 正在执行完全加载迁移。
- `Load complete, replication ongoing`：RDS 已完成完全加载，现在复制正在进行的更改。此状态仅适用于完全加载和 CDC 类型的迁移。
- `Replication ongoing`：RDS 正在复制正在进行的更改。此状态仅适用于 CDC 类型的迁移。
- `Stopping`：RDS 正在停止数据迁移。当您从操作菜单中选择停止数据迁移时，此状态适用。
- `Stopped`：RDS 已停止数据迁移。
- `Failed`：数据迁移已失败。有关更多信息，请参阅日志文件。
- `Restarting`：数据迁移已从 CDC 起点重新启动了正在进行的数据复制。

教程：使用自定义参数和新选项组创建 MySQL 数据库实例

在本教程中，您将使用自定义参数和新选项组，来创建 MySQL 数据库实例。有关自定义参数和选项组的更多信息，请参阅 Amazon RDS 的参数组 和使用选项组。

> **Important**
>
> 创建 Amazon 账户并不会收费；但是，在完成本教程过程中使用 Amazon 资源可能会产生费用。完成本教程后，如果不再需要这些资源，可以将其删除。

要使用自定义配置和设置创建数据库实例，您可以使用自定义参数和选项组。如果您使用多个数据库并希望为实例集统一配置设置，则自定义参数和新选项组特别有用。

通过完成这些步骤，您将学习：

- 如何使用 Amazon RDS，通过自定义参数和新选项组创建 MySQL 数据库实例。
- 如何为 MySQL 数据库实例使用特定的参数和选项。

要完成本教程，请执行以下任务：

1. 使用 MySQL 参数 `default_password_lifetime` 和 `disconnect_on_expired_password` 创建一个自定义参数组。
2. 使用 MySQL 选项功能 `MariaDB Audit Plugin` 创建新的选项组。有关创建选项组的步骤，请参阅使用选项组。
3. 使用您创建的自定义参数组和新选项组创建 MySQL 数据库实例。

主题

先决条件

本教程要求您拥有一个 Amazon Web Services 账户和一个具有管理权限的用户。如果您还没有设置这些内容，请完成以下各节中的步骤：

- 注册 Amazon Web Services 账户
- 保护 IAM 用户

创建 Amazon RDS参数组

在本教程中，您将了解如何在控制台中为 MySQL 数据库实例创建一个具有 default_password_lifetime 和 disconnect_on_expired_password 的自定义参数组。`default_password_lifetime` 参数控制客户端密码自动过期之前的天数。`disconnect_on_expired_password` 参数控制密码到期后 MySQL 数据库实例是否断开与客户端的连接。有关可用于 MySQL 数据库实例的其它参数的更多信息，请参阅 MySQL 的参数。

创建参数组

1. 打开 Amazon RDS 控制台，然后选择参数组。
2. 对于自定义参数组，选择创建参数组。
3. 设置参数组详细信息。
    1. 输入参数组的名称。
    2. 输入参数组的描述。
    3. 对于引擎类型，选择 MySQL 社区。 支持蓝绿部署。
    4. 对于参数组系列，选择 MySQL 8.0。
4. 选择创建。

新的参数组将显示在 Amazon RDS 控制台的参数组页面上。以下步骤说明如何修改参数值以自定义参数组。

修改自定义参数组中的参数值

使用以下步骤修改您在创建 Amazon RDS参数组中创建的参数组中的参数值。

修改参数组中的参数值

1. 打开 Amazon RDS 控制台，然后选择参数组。
2. 对于自定义参数组，请选择您创建的参数组的名称。
3. 选择编辑。
4. 在筛选参数搜索框中，搜索自定义参数 `default_password_lifetime`。
5. 选中参数旁边的复选框，并输入要为此密码生命周期参数设置的天数值。
6. 选择保存更改。
7. 对参数 `disconnect_on_expired_password` 重复相同的步骤。当您选择此参数时，系统会提示您从下拉菜单中选择 0 或 1 的值。选择 1 可在密码过期时断开连接。

自定义参数组现在可以与 Amazon RDS for MySQL 8.0 数据库实例相关联。接下来，为数据库实例创建新选项组。

创建新的 Amazon RDS 选项组

使用 MariaDB 审计插件选项创建新选项组。此插件记录服务器活动以确保安全性和合规性。有关可用于 MySQL 数据库实例的其它自定义选项的更多信息，请参阅适用于 MySQL 数据库实例的选项。

创建选项组

1. 打开 Amazon RDS 控制台，然后选择选项组。
2. 对于选项组，选择创建组。
3. 设置选项组详细信息。
   - 输入选项组的名称。
   - 输入选项组的描述。
   - 对于引擎，选择 mysql。
   - 对于主要引擎版本，选择 8.0。
4. 选择创建。

新的选项组将显示在 Amazon RDS 控制台的选项组页面上。以下步骤显示了如何将选项添加到选项组。

将选项添加到新选项组

使用以下步骤将选项添加到您在创建新的 Amazon RDS 选项组中创建的新选项组。

将选项添加到选项组

1. 打开 Amazon RDS 控制台，然后选择选项组。
2. 对于选项组，选择您创建的选项组的名称。
3. 在选项下，选择添加选项。
4. 设置选项组详细信息。
    - 对于选项名称，选择选项 MariaDB 审计插件 MARIADB_AUDIT_PLUGIN。
    - 对于选项设置，使所有默认选项保持选中状态。
    - 对于立即应用，选择是。
5. 选择创建选项。

现在，新选项组应可用于所有关联的数据库实例。接下来，使用自定义参数和新选项组创建一个 MySQL 数据库实例。

使用自定义参数和新选项组创建 MySQL 数据库实例

最后，使用您在前面的步骤中创建的自定义参数和新选项组，创建 MySQL 数据库实例。以下步骤显示了如何使用自定义参数和新选项组创建 MySQL 数据库实例。

使用自定义参数和新选项组创建数据库实例

1. 打开 Amazon RDS 控制台并选择数据库。
2. 选择创建数据库。
3. 对于选择数据库创建方法，选择标准创建。
4. 对于引擎选项，选择 MySQL。
5. 对于可用性和耐久性，选择单个数据库实例。此步骤是支持自定义参数或新选项组所必需的。
6. 选择其它配置。
    - 对于初始数据库名称，为数据库实例选择名称。
    - 在数据库参数组下拉列表中，选择您之前创建的自定义参数组的名称。
    - 在选项组下拉列表下，选择您之前创建的新选项组的名称。

7. 对于本教程，可以保留任何其它数据库设置的默认设置，也可以根据需要对其进行修改。
8. 选择创建数据库。

RDS 使用自定义参数和新选项组创建新的 MySQL 数据库实例。要查看有关此数据库的更多信息，请参阅 Amazon RDS 控制台的数据库页面。

在本教程中，您使用自定义参数和新选项组，通过定制的设置配置了 MySQL 数据库实例。这个新创建的 MySQL 数据库实例使用 `default_password_lifetime` 参数来管理用户密码生命周期。此数据库实例还使用参数 `disconnect_on_expired_password` 断开使用过期密码进行连接的用户。还可以使用选项 `MariaDB Audit Plugin` 来保持跟踪服务器活动。要优化数据库，可以在自定义参数组中应用其它设置并添加选项。

创建完自定义数据库实例后，应删除资源，以免产生不必要的费用。要删除数据库实例，请按照[删除数据库实例](#)中的说明操作。

管理 Amazon RDS 数据库实例

在下文中您可以找到有关管理和维护 Amazon RDS 数据库实例的说明。

主题

暂时停止 Amazon RDS 数据库实例

您可以间歇性地停止数据库实例，来进行临时测试或执行日常开发活动，最长可以连续停止 7 天。最常见的使用案例是成本优化。

停止数据库实例的时间因实例类、网络状态、数据库引擎类型和数据库状态等因素而异。此过程可能需要几分钟时间。该服务必须执行以下操作：

- 关闭数据库引擎进程。
- 关闭 RDS 平台进程。
- 分离与数据库实例关联的 EBS 存储卷。
- 终止底层 Amazon EC2 实例。

> **Warning**
>
> 启动数据库实例需要实例恢复，且可能需要几分钟到几小时的时间。因此，如果担心实例可用性，请谨慎处理临时停止生产实例的事宜。有关更多信息，请参阅 启动之前停止的 Amazon RDS 数据库实例。

要在相同的操作中停止和启动数据库实例，请重启该数据库实例。有关更多信息，请参阅 重启中的数据库实例。

主题

停止数据库实例的使用案例

与创建数据库快照、删除数据库实例、然后在需要访问实例时恢复快照相比，停止和启动数据库实例要更加快速。停止实例的常见使用案例包括：

- 成本优化 – 对于非生产数据库，您可以暂时停止 Amazon RDS 数据库实例以节省资金。实例停止后，您无需为所用数据库实例的小时数付费。

> **Important**
>
> 当数据库实例停止后，您需要为预调配的存储（包括预调配 IOPS）付费。您还需要为备份存储付费，包括指定保留时段内的手动快照和自动备份。但是，您无需为所用数据库实例小时数付费。有关更多信息，请参阅账单常见问题。

- 日常开发 – 如果您出于开发目的维护数据库实例，则可以在需要时启动该实例，然后在不需要时将其关闭。
- 测试 – 您可能需要一个临时数据库实例来测试备份和恢复程序、迁移、应用程序升级或相关活动。在这些使用案例中，您可以在不需要数据库实例时将其停止。
- 培训 – 如果您在 RDS 中进行培训，则可能需要在培训期间启动数据库实例，然后再将其关闭。

支持的数据库引擎、实例类和区域

您可以停止和启动运行以下数据库引擎的 Amazon RDS 数据库实例：

- Db2
- MariaDB
- Microsoft SQL Server，包括 RDS Custom for SQL Server
- MySQL
- Oracle
- PostgreSQL

所有数据库实例类和所有 Amazon 区域都支持停止并启动数据库实例。

停止多可用区部署中的数据库实例

您可以停止和启动多可用区部署中的数据库实例。请注意以下限制：

- 您只能在数据库引擎支持的情况下创建多可用区部署。有关引擎版本的更多信息，请参阅支持 Amazon RDS 中多可用区数据库集群的区域和数据库引擎。
- RDS for SQL Server 不支持停止多可用区部署中的数据库实例。有关更多信息，请参阅 Microsoft SQL Server 多可用区部署限制、说明和建议。
- 停止数据库实例可能需要较长的时间。如果在上次故障转移之后至少进行了一次备份，那么可以通过执行一次包括故障转移操作的重启来加快停止操作。有关更多信息，请参阅 重启中的数据库实例。

停止数据库实例的工作原理

停止操作分为以下几个阶段：

1. 数据库实例启动正常关闭过程。

   数据库实例的状态更改为 `stopping`。
2. 实例停止运行，最多连续 7 天。

   数据库实例的状态更改为 `stopped`。

已停止数据库实例的特征

处于停止状态时，数据库实例具有以下特征：

- 当您停止数据库实例时，它会保留以下行为：
  - 实例 ID
  - 域名服务器（DNS）端点
  - 参数组
  - 安全组
  - 选项组
  - Amazon S3 事务日志（时间点还原所必需）

  当您重启数据库实例时，它的配置与您停止它时相同。
- 所有存储卷仍附加到数据库实例，而且其数据将会保留。RDS 删除存储在数据库实例的 RAM 中的所有数据。

  当数据库实例停止后，您需要为预调配的存储（包括预调配 IOPS）付费。您还需要为备份存储付费，包括指定保留时段内的手动快照和自动备份。

- RDS 将删除待处理的操作（包括计划进行的维护更新），但针对数据库实例的选项组或数据库参数组的待处理操作除外。

> **Note**
>
> 有时，RDS for PostgreSQL 数据库实例不会完全关闭。如果发生这种情况，您将看到实例在稍后重启时会经历恢复过程。这是数据库引擎的预期行为，旨在保护数据库完整性。一些基于内存的统计数据和计数器不会保留历史记录，重启后会重新初始化，以便捕获接下来的操作工作负载。

自动重启已停止的数据库实例

如果您的数据库实例在停止连续七天后没有手动启动，RDS 会自动为您启动数据库实例。这样，您的实例就可以及时获得任何所需的维护更新。要了解如何按计划停止和启动您的实例，请参阅如何使用 Step Functions 将 Amazon RDS 实例停止 7 天以上？。

停止数据库实例的限制

以下是停止操作的一些限制：

- 您无法停止具有只读副本或作为只读副本的数据库实例。
- 您无法修改已停止的数据库实例。
- 您无法删除与停止的数据库实例关联的选项组。
- 您无法删除与已停止的数据库实例关联的数据库参数组。
- 在多可用区部署中，请注意以下限制：
  - 无法停止 RDS for SQL Server 数据库实例。
  - 在启动数据库实例后，主要可用区和辅助可用区可能会切换。

其他限制适用于 RDS Custom for SQL Server。有关更多信息，请参阅 启动和停止 RDS Custom for SQL Server 数据库实例。

选项和参数组注意事项

如果存在与一个选项组关联的数据库实例，则无法从该选项组中删除持久选项 (包括永久选项)。该功能也适用于状态为 `stopping`、`stopped` 或 `starting` 的任何数据库实例。

您可以更改与已停止的数据库实例关联的选项组或数据库参数组。但是，直到下次启动数据库实例时，才会发生更改。如果您选择立即应用更改，则更改将在您启动数据库实例时生效。否则，更改将在下个维护时段内您启动数据库实例之后才会生效。

公有 IP 地址注意事项

当您停止一个数据库实例时，它将保留其 DNS 端点。如果您停止具有公有 IP 地址的数据库实例，请 Amazon RDS 公开其公有 IP 地址。重新启动数据库实例时，它具有不同的公有 IP 地址。

> **Note**
>
> 您应始终使用 DNS 端点而不是 IP 地址来连接到数据库实例。

暂时停止数据库实例：基本步骤

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 停止数据库。

控制台

停止数据库实例

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)，然后选择要停止的数据库实例。
3. 对于 Actions（操作），选择 Stop temporarily（临时停止）。
4. 在 Stop DB instance temporarily（临时停止数据库实例）窗口中，选择确认数据库实例将在 7 天后自动重启。
5. （可选）选择 Save the DB instance in a snapshot（在快照中保存数据库实例），然后为 Snapshot name（快照名称）输入快照名称。如果要在停止数据库实例之前创建其快照，请选择此选项。
6. 选择 Stop temporarily（临时停止）以停止数据库实例，或选择 Cancel（取消）以取消该操作。

Amazon CLI

要使用 Amazon CLI 停止数据库实例，请使用以下选项调用 stop-db-instance 命令：

- `--db-instance-identifier` – 数据库实例的名称。

Example

```
aws rds stop-db-instance --db-instance-identifier mydbinstance
```

RDS API

要使用 Amazon RDS API 停止数据库实例，请使用以下参数调用 StopDBInstance 操作：

- `DBInstanceIdentifier` – 数据库实例的名称。

启动之前停止的 Amazon RDS 数据库实例

本主题介绍如何启动之前停止的 Amazon RDS 数据库实例，概述了恢复使用数据库的必要步骤和关键注意事项。

您可以暂时停止 Amazon RDS 数据库实例以节省资金。在停止数据库实例后，您可以重新启动它以再次开始使用。有关停止数据库实例的更多信息，请参阅暂时停止 Amazon RDS 数据库实例。

当您启动之前停止的数据库实例时，该数据库实例将保留如下信息:

- 实例 ID
- 域名服务器（DNS）端点
- 数据库参数组
- VPC 安全组
- 数据库选项组

Amazon RDS 以一秒为增量对数据库实例和附加存储进行计费。启动实例时，最低收费为 10 分钟。

要启动实例，Amazon RDS 服务会执行如下操作：

- 预置底层 Amazon EC2 实例
- 启动 RDS 进程
- 启动数据库引擎进程
- 连接 EBS 存储卷
- 启用性能详情（如果之前已启用）
- 恢复数据库实例（即使在正常关闭后也恢复）

启动数据库实例的时间因实例类、网络状态、数据库引擎类型、数据库大小和实例关闭时的数据库状态等因素而异。启动过程可能需要几分钟到几小时。我们建议您在创建可用性计划时考虑启动时间的可变性。

控制台

启动数据库实例

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。

2. 在导航窗格中，选择 Databases (数据库)，然后选择要启动的数据库实例。
3. 对于 Actions (操作)，选择 Start (开始)。

Amazon CLI

要使用 Amazon CLI 启动数据库实例，请使用以下参数调用 start-db-instance 命令：

- `--db-instance-identifier` – 数据库实例的名称。

Example

```
aws rds start-db-instance --db-instance-identifier mydbinstance
```

RDS API

要使用 Amazon RDS API 启动数据库实例，请使用以下参数调用 StartDBInstance 操作：

- `DBInstanceIdentifier` – 数据库实例的名称。

重启中的数据库实例

您可以通过一项名为重启的操作在 RDS 数据库实例上停止和启动数据库服务。可能需要重启才能应用配置更改、解决小问题或解决网络问题，而无需对数据库执行完全重启或迁移。

> **Note**
>
> 本主题仅适用于重启单可用区或多可用区数据库实例。有关重启多可用区数据库集群的说明，请参阅the section called “重启多可用区数据库集群”。

主题

重启中数据库实例的使用案例

通常出于维护原因而重启数据库实例，以使您的更改生效。以下是常见的使用案例：

- 关联新数据库参数组 – 将新数据库参数组与数据库实例关联时，RDS 将仅在数据库实例重启后才应用修改后的静态和动态参数。但是，如果在将数据库参数组与数据库实例关联之后修改数据库参数组中的动态参数，这些更改将立即得到应用，而无需重启。有关更多信息，请参阅 Amazon RDS 的参数组。
- 将对静态参数的更改应用于现有数据库参数组 - 当您更改静态参数并保存数据库参数组时，控制台中与该参数组关联的数据库实例的状态将更改为 pending-reboot。参数更改仅在重启关联的数据库实例之后才会生效。当您更改现有参数组中的动态参数时，更改默认情况下会立即生效，不需要重启。

> **Note**
>
> pending-reboot 状态不会在下一个维护时段导致自动重启。要向数据库实例应用最新的参数更改，请手动重启数据库实例。有关参数组的更多信息，请参阅 Amazon RDS 的参数组。

- 故障排除 - 您可能会遇到需要重启的性能或其他操作问题。例如，数据库实例可能没有响应。

如何重启中的数据库实例

当 Amazon RDS 重启您的数据库实例时，它会执行以下连续任务：

1. 在数据库实例中停止相应的数据库服务
2. 在数据库实例中启动相应的数据库服务

重启过程会导致短暂中断。在此中断期间，数据库实例的状态为重启。即使您通过失效转移进行重启，单可用区部署和多可用区数据库实例部署也会出现中断。

如何在多可用区部署中重启数据库实例

如果 Amazon RDS 数据库实例位于多可用区部署中，则可以通过失效转移进行重启。此操作对于模拟数据库实例的故障，或在效转移后将操作还原到原始可用区非常有用。

在包括失效转移的重启期间，Amazon RDS 将执行以下操作

- 立即中断数据库。数据库实例及其客户端会话可能来不及正常关闭。

> **Warning**
>
> 为避免数据丢失，建议先停止数据库实例上的事务，再使用失效转移进行重启。

- 必要时对数据库执行崩溃恢复。
- 自动切换到另一个可用区中的备用副本。可用区更改可能在几分钟内无法反映在 Amazon Web Services Management Console、对 Amazon CLI 的调用以及 RDS API 中。
- 更新数据库实例的 DNS 记录，使其指向备用数据库实例。因此，您需要清理并重新建立到数据库实例的任何现有连接。有关更多信息，请参阅 [配置和管理 Amazon RDS 的多可用区部署]。
- 在重启后创建 Amazon RDS 事件。

在 RDS for Microsoft SQL Server 上，失效转移仅重启主数据库实例。失效转移后，主数据库实例将成为新的备用数据库实例。多可用区实例的参数可能不更新。对于不进行失效转移的重新启动，主数据库实例和备用数据库实例将重新启动，并在重新启动后更新参数。如果数据库实例没有响应，我们建议重新启动而不进行失效转移。

重启中的数据库实例时的注意事项

在重启实例之前，请考虑以下事项：

- 对于具有只读副本的数据库实例，您可以独立地重启源数据库实例及其只读副本。重启完成后，复制将自动恢复。
- 重启时间取决于崩溃恢复过程、重启时的数据库活动以及特定数据库引擎的行为。为缩短重启时间，建议在重启期间尽可能减少数据库活动。此方法可减少传输中事务的回滚活动。

重启中数据库实例的先决条件

请确保满足以下先决条件：

- 数据库实例必须处于 `available` 状态。您的数据库可能会由于几个原因而不可用，例如，正在进行备份、以前请求的修改或维护时段操作。
- 如果您强制失效转移到其他可用区，则必须为多可用区配置数据库实例。
- 如果您强制失效转移到其他可用区，建议您先停止数据库实例上的事务，以防止可能丢失数据。

重启中的数据库实例：基本步骤

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 重启数据库实例。

控制台

重启数据库实例

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择数据库，然后选择要重启的数据库实例。
3. 对于操作，选择重启。

   将显示重启数据库实例页面。
4. （可选）选择重启失效转移?强制在可用区之间进行失效转移。
5. 选择重新引导以重新引导您的数据库实例。

   或者，选择取消。

Amazon CLI

要使用 Amazon CLI 重新引导数据库实例，请调用 [reboot-db-instance](#) 命令。

Example 简单重启

对于 Linux、macOS 或 Unix：

```
aws rds reboot-db-instance \
    --db-instance-identifier mydbinstance
```

对于 Windows：

```
aws rds reboot-db-instance ^
    --db-instance-identifier mydbinstance
```

Example 通过失效转移重启

若要在多可用区数据库集群中，从一个可用区强制失效转移到另一个可用区，请使用 `--force-failover` 参数。

对于 Linux、macOS 或 Unix：

```
aws rds reboot-db-instance \
    --db-instance-identifier mydbinstance \
    --force-failover
```

对于 Windows：

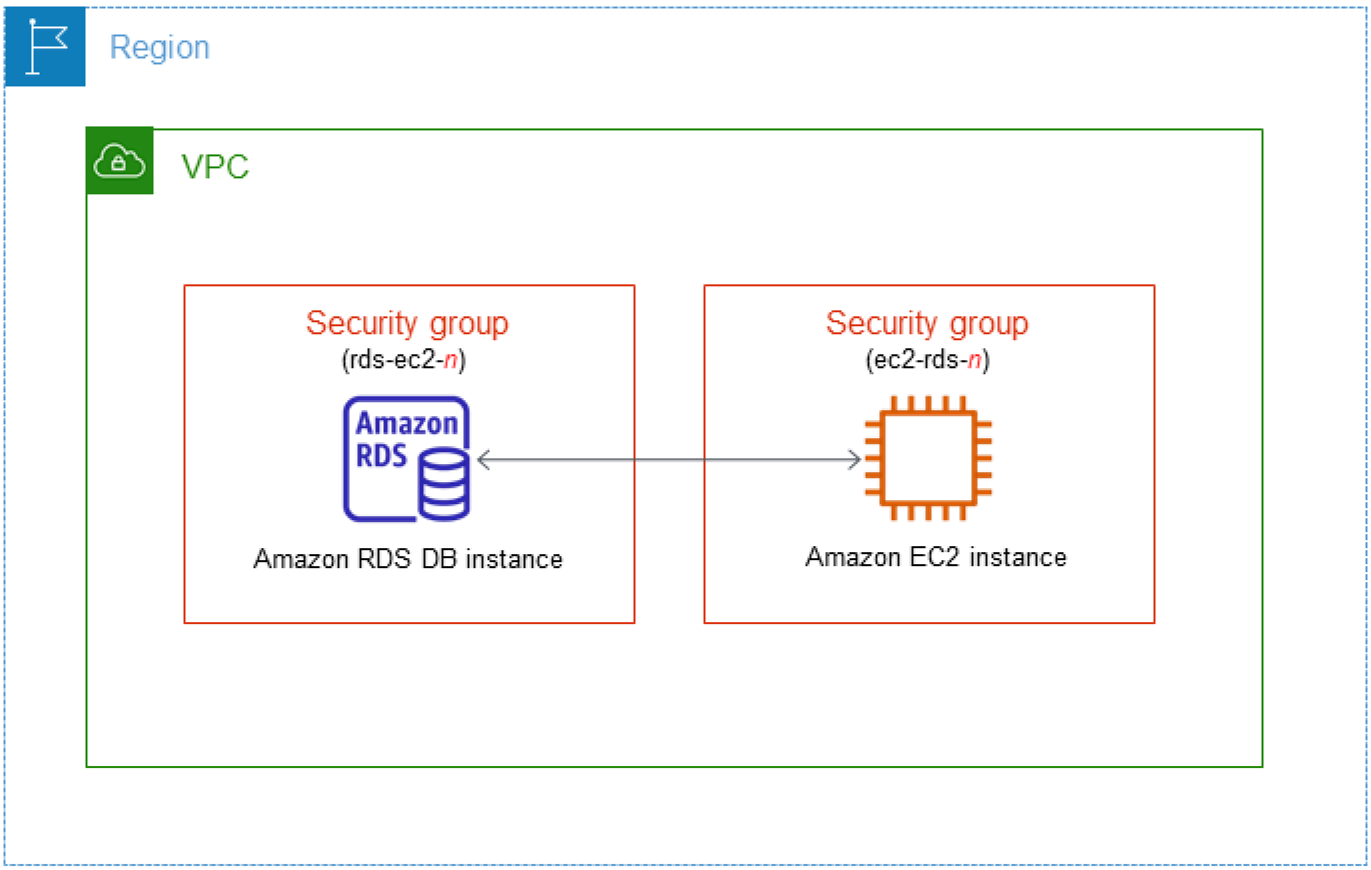
```
aws rds reboot-db-instance ^
    --db-instance-identifier mydbinstance ^
    --force-failover
```

RDS API

要使用 Amazon RDS API 重启数据库实例，请调用 [RebootDBInstance](#) 操作。

自动连接 EC2 实例和数据库实例

您可以使用 RDS 控制台来简化在 Amazon Elastic Compute Cloud（Amazon EC2）实例和数据库实例之间设置连接的过程。通常，数据库实例位于私有子网中，而 EC2 实例位于 VPC 内的公有子网中。您可以在 EC2 实例上使用 SQL 客户端连接到数据库实例。EC2 实例还可以运行访问私有数据库实例的 Web 服务器或应用程序。有关在 EC2 实例和多可用区数据库集群之间设置连接的说明，请参阅the section called “连接 EC2 实例和多可用区数据库集群”。



如果您想连接到与数据库实例不在同一 VPC 中的 EC2 实例，请参阅在 VPC 中访问数据库实例的场景中的相应方案。

主题

与 EC2 实例的自动连接概述

当您在 EC2 实例和 RDS 数据库之间设置连接时，Amazon RDS 会自动为您的 EC2 实例和 RDS 数据库配置 VPC 安全组。

以下是将 EC2 实例与 RDS 数据库连接的要求：

- EC2 实例必须与 RDS 数据库存在于同一 VPC 中。

  如果同一 VPC 中不存在任何 EC2 实例，则控制台将提供创建一个此类实例的链接。
- 设置连接的用户必须具有执行以下 Amazon EC2 操作的权限：
  - `ec2:AuthorizeSecurityGroupEgress`
  - `ec2:AuthorizeSecurityGroupIngress`
  - `ec2:CreateSecurityGroup`
  - `ec2:DescribeInstances`
  - `ec2:DescribeNetworkInterfaces`
  - `ec2:DescribeSecurityGroups`
  - `ec2:ModifyNetworkInterfaceAttribute`
  - `ec2:RevokeSecurityGroupEgress`

如果数据库实例和 EC2 实例位于不同的可用区，则您的账户可能会产生跨可用区成本。

当您设置与 EC2 实例的连接时，Amazon RDS 会根据与 RDS 数据库和 EC2 实例关联的安全组的当前配置采取操作，如下表所述。

当前 RDS 安全组配置	当前 EC2 安全组配置	RDS 操作
有一个或多个安全组与 RDS 数据库 [其名称与模式 `rds-ec2-`*n*（其中 *n* 是数字）相匹配] 关联。尚未修改与此模式匹配的安全组。该安全组只具有一条以 EC2 实例的 VPC 安全组作为源的入站规则。	有一个或多个安全组与 EC2 实例关联，此实例的名称与模式 `ec2-rds-`*n*（其中 *n* 是数字）相匹配。尚未修改与此模式匹配的安全组。该安全组只具有一条以 RDS 数据库的 VPC 安全组作为源的出站规则。	RDS 不执行任何操作。 已在 EC2 实例和 RDS 数据库之间自动配置了连接。由于 EC2 实例和 RDS 数据库之间已经存在连接，因此不会修改安全组。

当前 RDS 安全组配置	当前 EC2 安全组配置	RDS 操作
以下任一条件适用： • 没有与 RDS 数据库（其名称与模式 `rds-ec2-n` 匹配）关联的安全组。 • 有一个或多个安全组与 RDS 数据库（其名称与模式 `rds-ec2-n` 相匹配）关联。但是，Amazon RDS 不能使用这些安全组中的任何一个来连接 EC2 实例。如果安全组没有一条以 EC2 实例的 VPC 安全组作为源的入站规则，则 Amazon RDS 无法使用该安全组。Amazon RDS 也无法使用经过修改的安全组。修改示例包括添加规则或更改现有规则的端口。	以下任一条件适用： • 没有与 EC2 实例（其名称与模式 `ec2-rds-n` 匹配）关联的安全组。 • 有一个或多个安全组与 EC2 实例关联，此实例的名称与模式 `ec2-rds-n` 相匹配。但是，Amazon RDS 不能将其中任何安全组用于连接 RDS 数据库。如果安全组没有一条以 RDS 数据库的 VPC 安全组作为源的出站规则，则 Amazon RDS 无法使用该安全组。Amazon RDS 也无法使用经过修改的安全组。	RDS action: create new security groups
有一个或多个安全组与 RDS 数据库（其名称与模式 `rds-ec2-n` 相匹配）关联。尚未修改与此模式匹配的安全组。该安全组只具有一条以 EC2 实例的 VPC 安全组作为源的入站规则。	有一个或多个安全组与 EC2 实例关联，此实例的名称与模式 `ec2-rds-n` 相匹配。但是，Amazon RDS 不能将其中任何安全组用于连接 RDS 数据库。如果安全组没有一条以 RDS 数据库的 VPC 安全组作为源的出站规则，则 Amazon RDS 无法使用该安全组。Amazon RDS 也无法使用经过修改的安全组。	RDS action: create new security groups

当前 RDS 安全组配置	当前 EC2 安全组配置	RDS 操作
有一个或多个安全组与 RDS 数据库（其名称与模式 `rds-ec2-n` 相匹配）关联。尚未修改与此模式匹配的安全组。该安全组只具有一条以 EC2 实例的 VPC 安全组作为源的入站规则。	存在用于连接的有效 EC2 安全组，但它与 EC2 实例不关联。此安全组的名称与模式 `ec2-rds-n` 相匹配。尚未修改它。它只具有一条以 RDS 数据库的 VPC 安全组作为源的出站规则。	RDS action: associate EC2 security group
以下任一条件适用： • 没有与 RDS 数据库（其名称与模式 `rds-ec2-n` 匹配）关联的安全组。 • 有一个或多个安全组与 RDS 数据库（其名称与模式 `rds-ec2-n` 相匹配）关联。但是，Amazon RDS 不能使用这些安全组中的任何一个来连接 EC2 实例。如果安全组没有一条以 EC2 实例的 VPC 安全组作为源的入站规则，则 Amazon RDS 无法使用该安全组。Amazon RDS 也无法使用经过修改的安全组。	有一个或多个安全组与 EC2 实例关联，此实例的名称与模式 `ec2-rds-n` 相匹配。尚未修改与此模式匹配的安全组。该安全组只具有一条以 RDS 数据库的 VPC 安全组作为源的出站规则。	RDS action: create new security groups

RDS 操作：创建新的安全组

Amazon RDS 执行以下操作：

- 创建与模式 `rds-ec2-n` 匹配的新安全组。该安全组具有一条以 EC2 实例的 VPC 安全组作为源的入站规则。此安全组与 RDS 数据库关联，并允许 EC2 实例访问 RDS 数据库。
- 创建与模式 `ec2-rds-n` 匹配的新安全组。该安全组具有一条以 RDS 数据库的 VPC 安全组作为目标的出站规则。该安全组与 EC2 实例相关联，并允许 EC2 实例向 RDS 数据库发送流量。

RDS 操作：关联 EC2 安全组

Amazon RDS 将有效的现有 EC2 安全组与 EC2 实例关联。该安全组允许 EC2 实例向 RDS 数据库发送流量。

自动连接 EC2 实例和 RDS 数据库

在 EC2 实例与 RDS 数据库之间设置连接之前，请确保满足与 EC2 实例的自动连接概述中所述的要求。

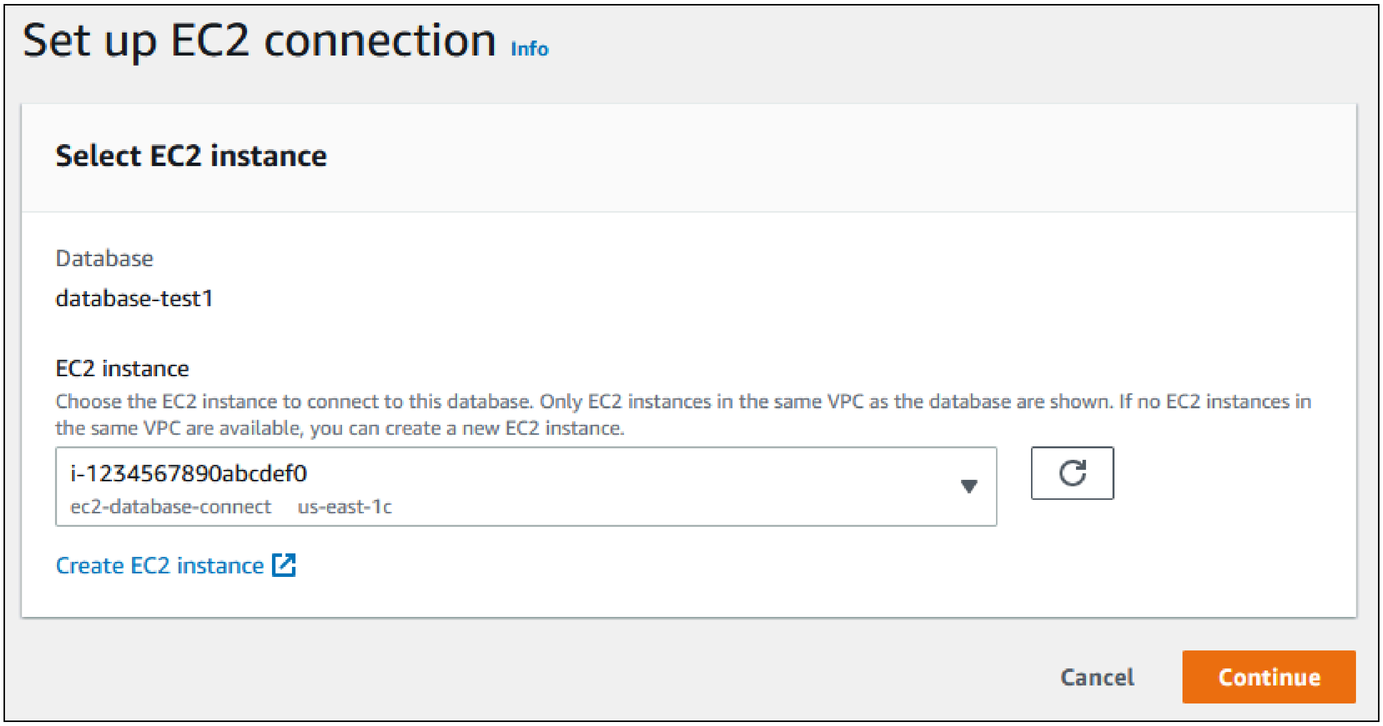
如果您在配置连接后更改安全组，则这些更改可能会影响 EC2 实例与 RDS 数据库之间的连接。

> **Note**
>
> 您只能使用 Amazon Web Services Management Console自动在 EC2 实例与 RDS 数据库之间设置连接。您无法使用 Amazon CLI 或 RDS API 自动设置连接。

自动连接 EC2 实例与 RDS 数据库

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases（数据库），然后选择 RDS 数据库。
3. 从操作中，选择设置 EC2 连接。

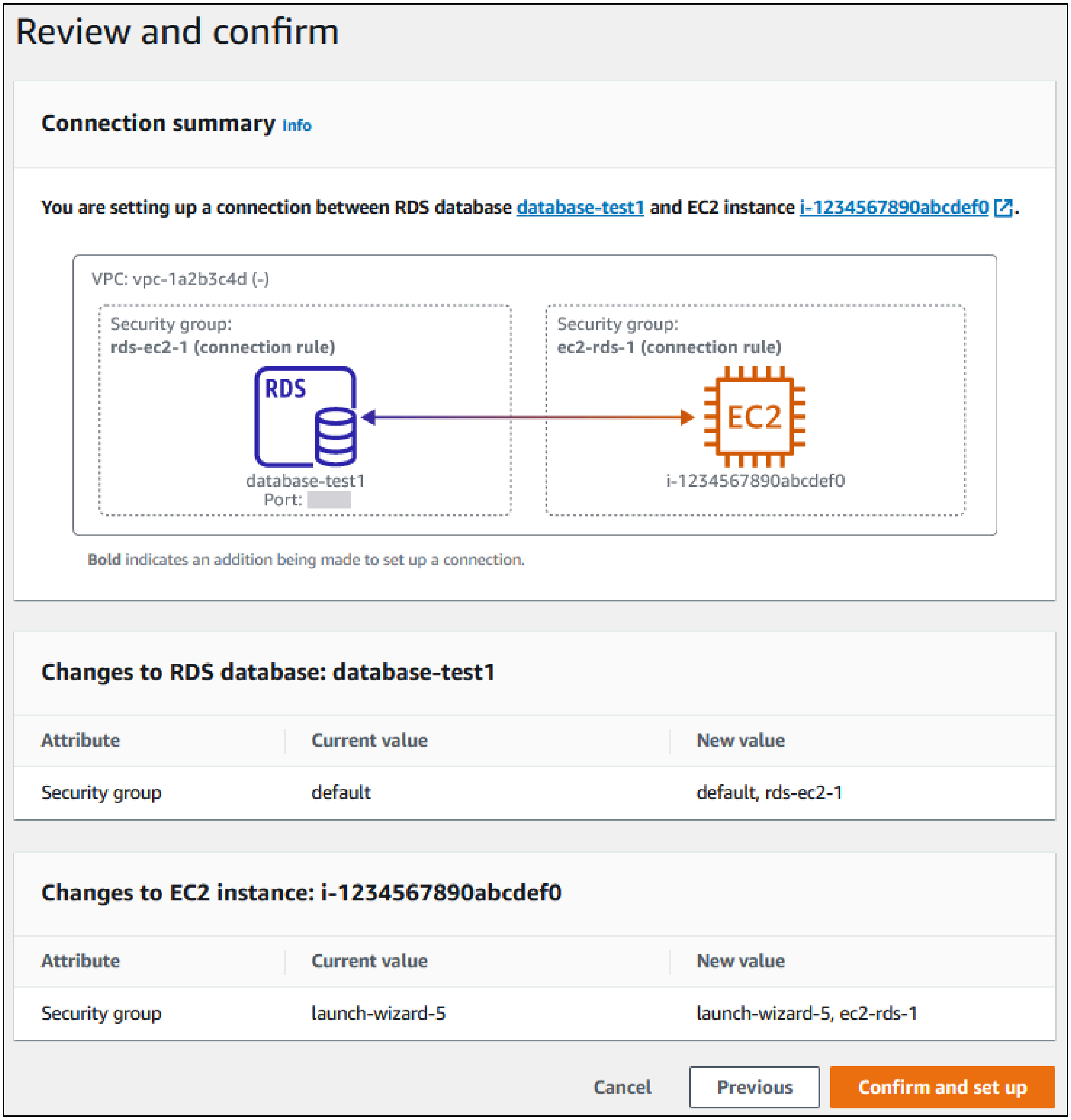
   将出现 Set up EC2 connection（设置 EC2 连接）页面。
4. 在 Set up EC2 connection（设置 EC2 连接）页上，选择 EC2 实例。



如果同一 VPC 中不存在任何 EC2 实例，请选择 Create EC2 instance（创建 EC2 实例）来创建一个此类实例。在这种情况下，请确保新的 EC2 实例与 RDS 数据库位于同一 VPC 中。

5. 选择继续。

   将出现 Review and confirm（检查并确认）页面。



6. 在 Review and confirm（检查并确认）页面上，检查 RDS 为设置与 EC2 实例的连接而将进行的更改。

   如果更改正确，请选择确认并设置。

   如果更改不正确，请选择 Previous（上一步）或 Cancel（取消）。

查看连接的计算资源

您可以使用 Amazon Web Services Management Console查看连接到 RDS 数据库的计算资源。显示的资源包括自动设置的计算资源连接。您可以通过以下方式自动设置与计算资源的连接：

- 您可以在创建数据库时选择计算资源。

  有关更多信息，请参阅创建 Amazon RDS 数据库实例 和创建 Amazon RDS 的多可用区数据库集群。
- 您可以在现有数据库和计算资源之间设置连接。

  有关更多信息，请参阅 自动连接 EC2 实例和 RDS 数据库。

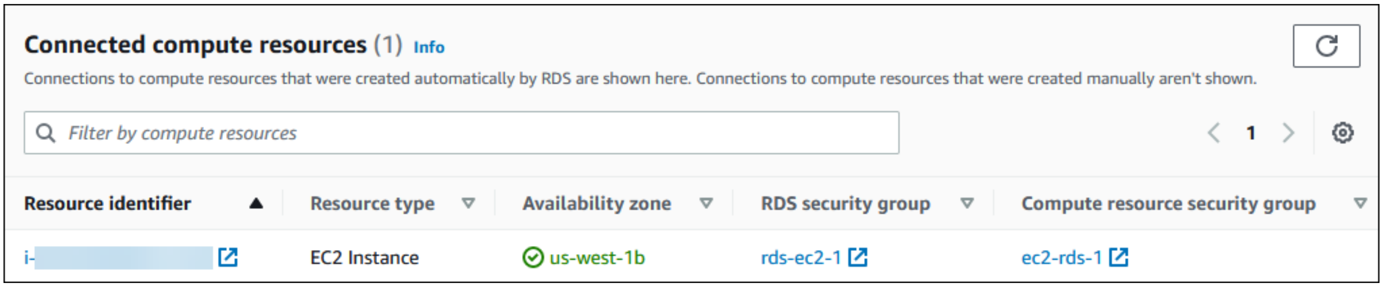
列出的计算资源不包括手动连接到数据库的计算资源。例如，您可以通过向与数据库关联的 VPC 安全组添加规则来允许计算资源手动访问数据库。

要列出计算资源，必须满足以下条件：

- 与计算资源关联的安全组的名称与模式 `ec2-rds-n` 相匹配（其中 *n* 是数字）。
- 与计算资源关联的安全组具有出站规则，其端口范围设置为 RDS 数据库使用的端口。
- 与计算资源关联的安全组具有出站规则，源设置为与 RDS 数据库关联的安全组。
- 与 RDS 数据库关联的安全组的名称与模式 `rds-ec2-n`（其中 *n* 是数字）相匹配。
- 与 RDS 数据库关联的安全组具有入站规则，其端口范围设置为 RDS 数据库使用的端口。
- 与 RDS 数据库关联的安全组有一条入站规则，其源设置为与计算资源关联的安全组。

查看连接到 RDS 数据库的计算资源

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases（数据库），然后选择 RDS 数据库的名称。
3. 在 Connectivity & security（连接和安全）选项卡上，在 Connected compute resources（连接的计算资源）中查看计算资源。



连接到运行特定数据库引擎的数据库实例

有关连接到运行特定数据库引擎的数据库实例的信息，请按照数据库引擎的说明操作：

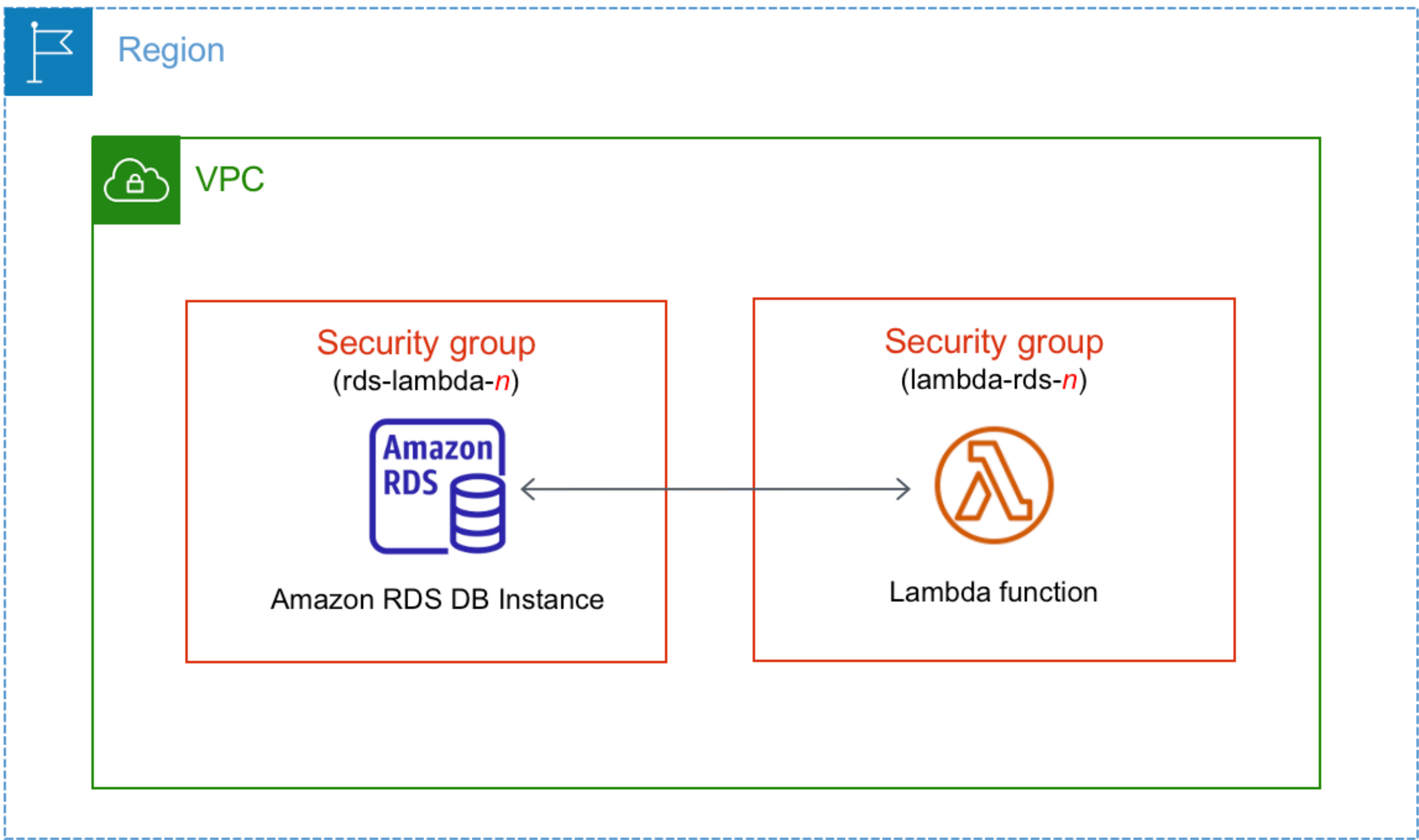
- 连接到 MariaDB 数据库实例
- 连接到 Microsoft SQL Server 数据库实例
- 连接到 MySQL 数据库实例
- 连接到 Oracle 数据库实例
- 连接到运行 PostgreSQL 数据库引擎的数据库实例

自动连接 Lambda 函数和数据库实例

您可以使用 Amazon RDS 控制台来简化在 Lambda 函数和数据库实例之间设置连接的过程。通常，您的数据库实例位于 VPC 内的私有子网中。应用程序可以使用 Lambda 函数访问您的私有数据库实例。

有关在 Lambda 函数和多可用区数据库集群之间设置连接的说明，请参阅the section called “连接 Lambda 函数和多可用区数据库集群”。

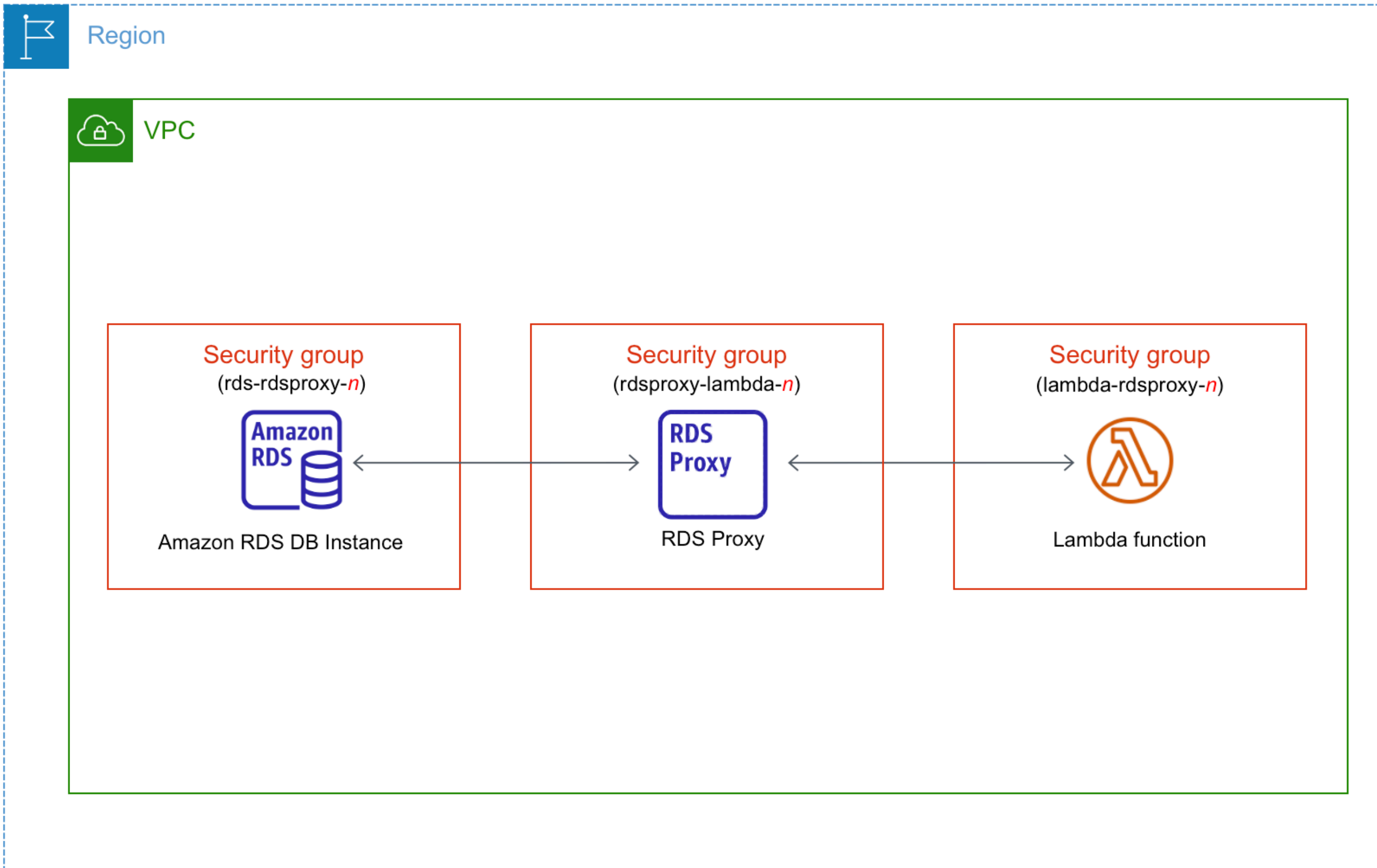
下图显示了数据库实例与您的 Lambda 函数之间的直接连接。



您可以通过 RDS 代理在 Lambda 函数和数据库实例之间设置连接，以提高数据库性能和弹性。通常，Lambda 函数会进行频繁的短期数据库连接，此类连接受益于 RDS 代理提供的连接池。您可以利用已为 Lambda 函数提供的任何 Amazon Identity and Access Management（IAM）身份验证，而不是在 Lambda 应用程序代码中管理数据库凭证。有关更多信息，请参阅 的 Amazon RDS 代理。

当您使用控制台通过现有代理进行连接时，Amazon RDS 会更新代理安全组，以允许来自数据库实例和 Lambda 函数的连接。

您也可以从同一个控制台页面创建新的代理。当您在控制台中创建代理时，要访问数据库实例，必须输入数据库凭证或选择 Amazon Secrets Manager 密钥。



> Tip
>
> 要将 Lambda 函数快速连接到数据库实例，还可以使用控制台内的引导式向导。要打开该向导，请执行以下操作：
>
> 1. 打开 Lamba 控制台的函数页面。
> 2. 选择要将数据库连接到其中的函数。
> 3. 在配置选项卡上，选择 RDS 数据库。
> 4. 选择连接到 RDS 数据库。
>
> 将函数连接到数据库后，您可以通过选择添加代理来创建代理。

主题

- 与 Lambda 函数的自动连接概述
- 自动连接 Lambda 函数和 RDS 数据库

- 查看连接的计算资源

与 Lambda 函数的自动连接概述

以下是将 Lambda 函数与 RDS 数据库实例连接的要求：

- Lambda 函数必须与数据库实例位于同一 VPC 中。
- 设置连接的用户必须具有执行以下 Amazon RDS、Amazon EC2、Lambda、Secrets Manager 和 IAM 操作的权限：
  - Amazon RDS
    - `rds:CreateDBProxies`
    - `rds:DescribeDBInstances`
    - `rds:DescribeDBProxies`
    - `rds:ModifyDBInstance`
    - `rds:ModifyDBProxy`
    - `rds:RegisterProxyTargets`
  - Amazon EC2
    - `ec2:AuthorizeSecurityGroupEgress`
    - `ec2:AuthorizeSecurityGroupIngress`
    - `ec2:CreateSecurityGroup`
    - `ec2:DeleteSecurityGroup`
    - `ec2:DescribeSecurityGroups`
    - `ec2:RevokeSecurityGroupEgress`
    - `ec2:RevokeSecurityGroupIngress`
  - Lambda
    - `lambda:CreateFunctions`
    - `lambda:ListFunctions`
    - `lambda:UpdateFunctionConfiguration`
  - Secrets Manager
    - `secretsmanager:CreateSecret`
    - `secretsmanager:DescribeSecret`
  - IAM

  - `iam:AttachPolicy`
  - `iam:CreateRole`
  - `iam:CreatePolicy`
- Amazon KMS
  - `kms:describeKey`

> **Note**
> 如果数据库实例与 Lambda 函数位于不同的可用区，您的账户可能会产生跨可用区成本。

当您在 Lambda 函数和 RDS 数据库之间设置连接时，Amazon RDS 会为您的函数和数据库实例配置 VPC 安全组。如果您使用 RDS 代理，则 Amazon RDS 还会为代理配置 VPC 安全组。Amazon RDS 根据与数据库实例、Lambda 函数和代理关联的安全组的当前配置执行操作，如下表所述。

当前 RDS 安全组配置	当前 Lambda 安全组配置	当前代理安全组配置	RDS 操作
有一个或多个安全组与数据库实例关联，其名称与模式 `rds-lambda-` *n* 匹配，或者，如果代理已连接到您的数据库实例，则 RDS 将检查关联代理的 `TargetHealth` 是否为 `AVAILABLE` 。 尚未修改与此模式匹配的安全组。该安全组只具有一条以 Lambda 函数或代理的 VPC 安全组作为源的入站规则。	有一个或多个安全组与 Lambda 函数关联，此函数的名称与模式 `lambda-rds-` *n* 或 `lambda-rdsproxy-` *n*（其中 *n* 是数字）相匹配。 尚未修改与此模式匹配的安全组。此安全组只有一条出站规则，此规则以数据库实例或代理的 VPC 安全组作为目标。	有一个或多个安全组与代理关联，此代理的名称与模式 `rdsproxy-lambda-` *n*（其中 *n* 是数字）相匹配。 尚未修改与此模式匹配的安全组。此安全组具有入站和出站规则，这些规则具有 Lambda 函数和数据库实例的 VPC 安全组。	Amazon RDS 不执行任何操作。 已在 Lambda 函数、代理（可选）和数据库实例之间自动配置了连接。由于函数、代理和数据库之间已经存在连接，因此不会修改安全组。

当前 RDS 安全组配置	当前 Lambda 安全组配置	当前代理安全组配置	RDS 操作
以下任一条件适用： • 没有与数据库实例（名称与模式 `rds-lambda-` *n* 匹配或关联代理的 `TargetHealth` 为 `AVAILABLE` ）相关联的安全组。 • 有一个或多个与数据库实例（名称与模式 `rds-lambda-` *n* 匹配或关联代理的 `TargetHealth` 为 `AVAILABLE` ）相关联的安全组。但是，这些安全组都不能用于连接 Lambda 函数。 如果安全组没有一条以 Lambda 函数或代理的 VPC 安全组作为源的入站规则，则 Amazon RDS 无法使用该安全组。Amazon RDS 也无法使用经过修改的安全组。修改示例包括添加规则或更改现有规则的端口。	以下任一条件适用： • 没有与 Lambda 函数（名称与模式 `lambda-rds-` *n* 或 `lambda-rdsproxy-` *n* 匹配）关联的安全组。 • 有一个或多个安全组与 Lambda 函数关联，此函数的名称与模式 `lambda-rds-` *n* 或 `lambda-rdsproxy-` *n* 相匹配。但是，Amazon RDS 不能使用这些安全组中的任何一个来连接数据库实例。 如果安全组没有一条以数据库实例或代理的 VPC 安全组作为目标的出站规则，则 Amazon RDS 无法使用该安全组。Amazon RDS 也无法使用经过修改的安全组。	以下任一条件适用： • 没有与代理（名称与模式 `rdsproxy-lambda-` *n* 匹配）关联的安全组。 • 有一个或多个安全组与代理关联，此代理的名称与 `rdsproxy-lambda-` *n* 相匹配。但是，Amazon RDS 不能使用这些安全组中的任何一个来连接数据库实例或 Lambda 函数。 如果安全组没有入站和出站规则（这些规则具有数据库实例和 Lambda 函数的 VPC 安全组），则 Amazon RDS 无法使用该安全组。Amazon RDS 也无法使用经过修改的安全组。	RDS action: create new security groups

当前 RDS 安全组配置	当前 Lambda 安全组配置	当前代理安全组配置	RDS 操作
有一个或多个与数据库实例（名称与模式 `rds-lambda-` *n* 匹配或关联代理的 `TargetHealth` 为 `AVAILABLE` ）相关联的安全组。 尚未修改与此模式匹配的安全组。该安全组只具有一条以 Lambda 函数或代理的 VPC 安全组作为源的入站规则。	有一个或多个安全组与 Lambda 函数关联，此函数的名称与模式 `lambda-rds-` *n* 或 `lambda-rdsproxy-` *n* 相匹配。 但是，Amazon RDS 不能使用这些安全组中的任何一个来连接数据库实例。如果安全组没有一条以数据库实例或代理的 VPC 安全组作为目标的出站规则，则 Amazon RDS 无法使用该安全组。Amazon RDS 也无法使用经过修改的安全组。	有一个或多个安全组与代理关联，此代理的名称与模式 `rdsproxy-lambda-` *n* 相匹配。 但是，Amazon RDS 不能使用这些安全组中的任何一个来连接数据库实例或 Lambda 函数。如果安全组没有入站和出站规则（这些规则具有数据库实例和 Lambda 函数的 VPC 安全组），则 Amazon RDS 无法使用该安全组。Amazon RDS 也无法使用经过修改的安全组。	RDS action: create new security groups
有一个或多个与数据库实例（名称与模式 `rds-lambda-` *n* 匹配或关联代理的 `TargetHealth` 为 `AVAILABLE` ）相关联的安全组。 尚未修改与此模式匹配的安全组。该安全组只具有一条以 Lambda 函数或代理的 VPC 安全组作为源的入站规则。	存在用于连接的有效 Lambda 安全组，但它与 Lambda 函数不关联。此安全组的名称与模式 `lambda-rds-` *n* 或 `lambda-rdsproxy-` *n* 相匹配。尚未修改它。它只有一条出站规则，此规则以数据库实例或代理的 VPC 安全组作为目标。	存在用于连接的有效代理安全组，但它与代理不关联。此安全组的名称与模式 `rdsproxy-lambda-` *n* 相匹配。尚未修改它。它具有入站和出站规则，这些规则具有数据库实例和 Lambda 函数的 VPC 安全组。	RDS action: associate Lambda security group

当前 RDS 安全组配置	当前 Lambda 安全组配置	当前代理安全组配置	RDS 操作
以下任一条件适用： • 没有与数据库实例（名称与模式 `rds-lambda-` *n* 匹配或关联代理的 `TargetHealth` 为 `AVAILABLE` ）相关联的安全组。 • 有一个或多个与数据库实例（名称与模式 `rds-lambda-` *n* 匹配或关联代理的 `TargetHealth` 为 `AVAILABLE` ）相关联的安全组。但是，Amazon RDS 不能使用这些安全组中的任何一个来连接 Lambda 函数或代理。 如果安全组没有一条以 Lambda 函数或代理的 VPC 安全组作为源的入站规则，则 Amazon RDS 无法使用该安全组。Amazon RDS 也无法使用经过修改的安全组。	有一个或多个安全组与 Lambda 函数关联，此函数的名称与模式 `lambda-rds-` *n* 或 `lambda-rdsproxy-` *n* 相匹配。 尚未修改与此模式匹配的安全组。此安全组只有一条出站规则，此规则以数据库实例或代理的 VPC 安全组作为目标。	有一个或多个安全组与代理关联，此代理的名称与模式 `rdsproxy-lambda-` *n* 相匹配。 尚未修改与此模式匹配的安全组。此安全组具有入站和出站规则，这些规则具有数据库实例和 Lambda 函数的 VPC 安全组。	[RDS action: create new security groups](#)

当前 RDS 安全组配置	当前 Lambda 安全组配置	当前代理安全组配置	RDS 操作
以下任一条件适用： • 没有与数据库实例（名称与模式 `rds-lambda- n` 匹配或关联代理的 `TargetHealth` 为 `AVAILABLE` ）相关联的安全组。 • 有一个或多个与数据库实例（名称与模式 `rds-lambda- n` 匹配或关联代理的 `TargetHealth` 为 `AVAILABLE` ）相关联的安全组。但是，Amazon RDS 不能使用这些安全组中的任何一个来连接 Lambda 函数或代理。 如果安全组没有一条以 Lambda 函数或代理的 VPC 安全组作为源的入站规则，则 Amazon RDS 无法使用该安全组。Amazon RDS 也无法使用经过修改的安全组。	以下任一条件适用： • 没有与 Lambda 函数（名称与模式 `lambda-rds- n` 或 `lambda-rdsproxy- n` 匹配）关联的安全组。 • 有一个或多个安全组与 Lambda 函数关联，此函数的名称与模式 `lambda-rds- n` 或 `lambda-rdsproxy- n` 相匹配。但是，Amazon RDS 不能使用这些安全组中的任何一个来连接数据库实例。 如果安全组没有一条以数据库实例或代理的 VPC 安全组作为源的出站规则，则 Amazon RDS 无法使用该安全组。Amazon RDS 也无法使用经过修改的安全组。	以下任一条件适用： • 没有与代理（名称与模式 `rdsproxy-lambda- n` 匹配）关联的安全组。 • 有一个或多个安全组与代理关联，此代理的名称与 `rdsproxy-lambda- n` 相匹配。但是，Amazon RDS 不能使用这些安全组中的任何一个来连接数据库实例或 Lambda 函数。 如果安全组没有入站和出站规则（这些规则具有数据库实例和 Lambda 函数的 VPC 安全组），则 Amazon RDS 无法使用该安全组。Amazon RDS 也无法使用经过修改的安全组。	[RDS action: create new security groups]()

RDS 操作：创建新的安全组

Amazon RDS 执行以下操作：

- 创建与模式 `rds-lambda-n` 或 `rds-rdsproxy-n` 匹配的新安全组（如果您选择使用 RDS 代理）。该安全组具有一条以 Lambda 函数或代理的 VPC 安全组作为源的入站规则。该安全组与数据库实例相关联，并允许函数或代理访问数据库实例。
- 创建与模式 `lambda-rds-n` 或 `lambda-rdsproxy-n` 匹配的新安全组。此安全组具有一条出站规则，此规则以数据库实例或代理的 VPC 安全组作为目标。此安全组与 Lambda 函数相关联，并允许该函数将流量发送到数据库实例或者通过代理发送流量。
- 创建与模式 `rdsproxy-lambda-n` 匹配的新安全组。此安全组具有入站和出站规则，这些规则具有数据库实例和 Lambda 函数的 VPC 安全组。

RDS 操作：关联 Lambda 安全组

Amazon RDS 将有效的现有的 Lambda 安全组与 Lambda 函数关联。此安全组允许该函数将流量发送到数据库实例或者通过代理发送流量。

自动连接 Lambda 函数和 RDS 数据库

您可以使用 Amazon RDS 控制台自动将 Lambda 函数连接到您的数据库实例。这简化了在这些资源之间设置连接的过程。

您也可以使用 RDS 代理在连接中包含代理。Lambda 函数会进行频繁的短期数据库连接，此类连接受益于 RDS 代理提供的连接池。您还可以使用已经为 Lambda 函数设置的任何 IAM 身份验证，而不是在 Lambda 应用程序代码中管理数据库凭证。

您可以使用设置 Lambda 连接页面将现有的数据库实例连接到新的和现有的 Lambda 函数。设置过程会自动为您设置所需的安全组。

在 Lambda 函数和数据库实例之间设置连接之前，请确保：

- 您的 Lambda 函数和数据库实例位于同一 VPC 中。
- 您拥有用户账户的相应权限。有关要求的更多信息，请参阅与 Lambda 函数的自动连接概述。

如果您在配置连接后更改安全组，则这些更改可能会影响 Lambda 函数与数据库实例之间的连接。

> **Note**
>
> 您只能在 Amazon Web Services Management Console中自动设置数据库实例和 Lambda 函数之间的连接。要连接 Lambda 函数，数据库实例必须处于可用状态。

自动连接 Lambda 函数和数据库实例

<result>

确认设置后，Amazon RDS 开始连接您的 Lambda 函数、RDS 代理（如果您使用了代理）和数据库实例的过程。控制台显示连接详细信息对话框，其中列出了允许在您的资源之间进行连接的安全组更改。

</result>

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择数据库，然后选择要连接到 Lambda 函数的数据库实例。
3. 对于操作，选择设置 Lambda 连接。
4. 在设置 Lambda 连接页面上，在选择 Lambda 函数下方，执行以下任一操作：
   - 如果您在与数据库实例相同的 VPC 中有一个现有的 Lambda 函数，请选择选择现有函数，然后选择该函数。
   - 如果您在同一 VPC 中没有 Lambda 函数，请选择创建新函数，然后输入函数名称。默认运行时系统设置为 Nodejs.18。完成连接设置后，您可以在 Lambda 控制台中修改新 Lambda 函数的设置。
5. （可选）在 RDS 代理下，选择使用 RDS 代理进行连接，然后执行以下任一操作：
   - 如果您有要使用的现有代理，请选择选择现有代理，然后选择此代理。
   - 如果您没有代理，并且希望 Amazon RDS 自动为您创建代理，请选择创建新代理。然后，对于数据库凭证，执行以下任一操作：

     a. 选择数据库用户名和密码，然后为您的数据库实例输入用户名和密码。

     b. 选择 Secrets Manager 密钥。然后，对于选择密钥，选择一个 Amazon Secrets Manager 密钥。如果您没有 Secrets Manager 密钥，请选择创建新的 Secrets Manager 密钥以创建新密钥。创建密钥后，对于选择密钥，选择新密钥。

     创建新代理后，选择选择现有代理，然后选择该代理。请注意，您的代理可能需要一些时间才能用于连接。

6. （可选）扩展连接摘要并验证您的资源的突出显示更新。
7. 选择 Set up (设置)。

查看连接的计算资源

您可以使用 Amazon Web Services Management Console查看连接到数据库实例的 Lambda 函数。显示的资源包括 Amazon RDS 自动设置的计算资源连接。

列出的计算资源不包括手动连接到数据库实例的计算资源。例如，您可以通过向与数据库关联的 VPC 安全组添加规则，来允许计算资源手动访问数据库实例。

要使控制台列出 Lambda 函数，必须满足以下条件：

- 与计算资源关联的安全组的名称与模式 `lambda-rds-n` 或 `lambda-rdsproxy-n`（其中 *n* 是数字）相匹配。
- 与计算资源关联的安全组具有出站规则，其端口范围设置为数据库实例或关联代理的端口。出站规则的目标必须设置为与数据库实例或关联代理关联的安全组。
- 如果配置包含代理，则连接到与您数据库关联的代理的安全组名称与模式 `rdsproxy-lambda-n`（其中 *n* 是一个数字）相匹配。
- 与函数关联的安全组具有出站规则，其端口设置为数据库实例或关联代理使用的端口。目标必须设置为与数据库实例或关联代理关联的安全组。

查看自动连接到数据库实例的计算资源

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择数据库，然后选择数据库集群。
3. 在连接和安全选项卡上，在连接的计算资源下查看计算资源。

修改 Amazon RDS 数据库实例

可以更改数据库实例的设置来完成添加额外存储或更改数据库实例类等任务。在本主题中，您可以了解如何修改 Amazon RDS 数据库实例并了解数据库实例的设置。

我们建议您在修改生产实例之前在测试实例上测试所有更改。这样做可以帮助您充分了解每项更改的影响。在升级数据库版本时，测试尤其重要。

对数据库实例的大多数修改都可以立即应用，也可以推迟到下一个维护时段。一些修改（例如，参数组更改）需要您手动重新启动数据库实例才能使更改生效。

> **Important**
>
> 一些修改可能会导致停机，因为 Amazon RDS 必须重新启动您的数据库实例才能使更改生效。在修改数据库实例设置之前，请先审核对数据库和应用程序的影响。

控制台

修改数据库实例

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择数据库，然后选择要修改的数据库实例。
3. 选择修改。将显示修改数据库实例页面。
4. 根据需要更改任意设置。有关每项设置的信息，请参阅 数据库实例的设置。
5. 当所有更改都达到您的要求时，选择继续并查看修改摘要。
6. （可选）选择立即应用以立即应用更改。选择此选项在某些情况下可能导致停机。有关更多信息，请参阅“使用计划修改设置”。
7. 在确认页面上，检查您的更改。如果更改正确无误，请选择 Modify DB Instance（修改数据库实例）保存更改。

   也可以选择 Back 编辑您的更改，或选择 Cancel 取消更改。

Amazon CLI

要使用 Amazon CLI 修改数据库实例，请调用 modify-db-instance 命令。指定数据库实例标识符以及要修改的选项值。有关各选项的信息，请参阅数据库实例的设置。

Example

以下代码修改 `mydbinstance`，将备份保留期设置为 1 周 (7 天)。该代码通过使用 `--deletion-protection` 启用删除保护。要禁用删除保护，请使用 `--no-deletion-protection`。将在下一维护时段使用 `--no-apply-immediately` 应用这些更改。使用 `--apply-immediately` 可立即应用更改。有关更多信息，请参阅 使用计划修改设置。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier mydbinstance \
    --backup-retention-period 7 \
    --deletion-protection \
    --no-apply-immediately
```

对于 Windows：

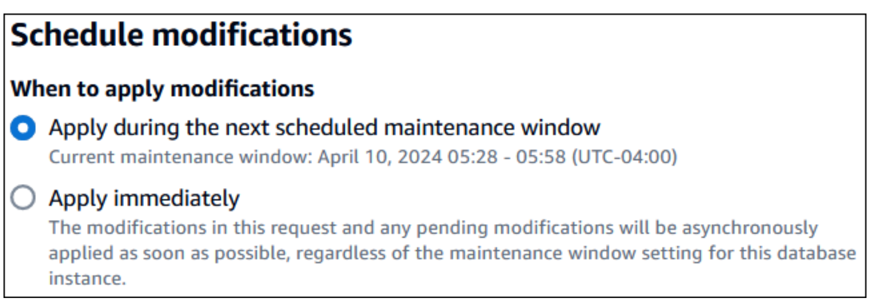
```
aws rds modify-db-instance ^
    --db-instance-identifier mydbinstance ^
    --backup-retention-period 7 ^
    --deletion-protection ^
    --no-apply-immediately
```

RDS API

要使用 Amazon RDS API 修改数据库实例，请调用 ModifyDBInstance 操作。指定数据库实例标识符，以及您要修改的设置参数。有关每个参数的信息，请参阅数据库实例的设置。

使用计划修改设置

修改数据库实例时，由您决定何时进行修改。



要立即应用更改，而不是在下一个维护时段应用更改，请选择 Amazon Web Services Management Console中的立即应用选项。或者，您可以在调用 Amazon CLI 参数时使用 `--apply-immediately` 参数，也可以在使用 Amazon RDS API 时将 `ApplyImmediately` 参数设置为 `true`。

如果您没有选择立即应用更改，RDS 将更改放入待定修改队列中。在下一维护时段内，RDS 将应用队列中的所有待定更改。如果您选择立即应用更改，将应用您的新更改以及等待修改队列中的所有更改。

要查看下一个维护时段待处理的修改，请使用 describe-db-instances Amazon CLI 命令并选中 `PendingModifiedValues` 字段。

> **Important**
>
> 如果任何待处理修改需要数据库实例暂时不可用（停机），则选择立即应用选项可能会导致意外停机。
>
> 当您选择立即应用更改时，任何待定修改也将立即应用，而不是在下一个维护时段内应用。
>
> 如果您不想将待定更改应用到下一个维护时段，您可以修改数据库实例来恢复更改。您可以通过使用 Amazon CLI 和指定 `--apply-immediately` 选项来执行此操作。

一些数据库设置的更改会立即应用，即使您选择延迟更改也是如此。要查看不同数据库的设置如何与“立即应用”设置进行交互，请参阅数据库实例的设置。

数据库实例的设置

在下表中，您可以找到有关可以修改和不能修改哪些设置的详细信息。还可以了解何时可以应用更改以及这些更改是否导致数据库实例停机。通过使用多可用区等 Amazon RDS 功能，您可以在稍后修改数据库实例时最大限度地减少停机时间。有关更多信息，请参阅 配置和管理 Amazon RDS 的多可用区部署。

您可以使用控制台、`modify-db-instance` CLI 命令或 `ModifyDBInstance` RDS API 操作修改数据库实例。

控制台设置和说明	CLI 选项和 RDS API 参数	发生更改时	停机说明	支持的数据库引擎
分配的存储空间	CLI 选项：	如果您选择立即应用更改，	在此更改期间，不会出现	所有数据

控制台设置和说明	CLI 选项和 RDS API 参数	发生更改时	停机说明	支持的数据库引擎

要分配给数据库实例的存储空间大小，单位为 GiB。您只能增加分配的存储空间。您无法减少分配的存储空间。
 	
您无法修改某些旧数据库实例或从旧数据库快照还原的数据库实例的存储。如果数据库实例不符合条件，则会在控制台中禁用分配的存储空间设置。您可以使用 CLI 命令 describe-valid-db-instance-modifications 检查是否可以分配更多的存储空间。此命令返回数据库实例的有效存储选项。
 	
如果数据库实例状态为 storage-optimization（存储-优化），则无法修改分配的存储。如果在过去 6 小时内修改了数据库实例，则也无法修改为此实例分配的存储。
 	
允许的最大存储空间取决于您的数据库引擎和存储类型。有关更多信息，请参阅“Amazon RDS 数据库实例存储”。
	`--allocated-storage`
	
	RDS API 参数：
	
	`AllocatedStorage`
		则更改立即生效。
		
		如果您不选择立即应用更改，则更改会在下一个维护时段中生效。
			停机。在更改期间，性能可能降低。	库引擎

控制台设置和说明	CLI 选项和 RDS API 参数	发生更改时	停机说明	支持的数据库引擎
架构配置 该配置允许在您的数据库实例中驻留多个租户数据库。目前，只有 RDS for Oracle 容器数据库（CDB）支持此设置。 如果您的 CDB 采用单租户配置，则可以将其修改为使用多租户配置。在此配置中，您可以使用 RDS API 创建 1-30 个租户数据库，具体取决于数据库版本和任何所需的选项许可证。不支持应用程序 PDB 和代理 PDB。多租户配置是永久性的，这意味着您以后无法将 CDB 恢复回单租户配置。 Note Amazon RDS 配置之所以称为“多-租户”而不是“多租户”，是因为它是 RDS 平台的一项功能，而不仅仅是 Oracle 数据库引擎的功能。同样，RDS 术语“租户”是指 RDS 配置中的任何租户，而不仅仅是 Oracle PDB。在 RDS 文档中，未加连字号的术语“Oracle 多租户”仅指 Oracle 数据库 CDB 架	CLI 选项： `--multi-tenant` （CDB 架构的多租户配置） `--no-multi-tenant`（CDB 架构的单租户配置） API 参数： `MultiTenant`	更改立即发生。	在此更改期间，不会停机。	Oracle

控制台设置和说明	CLI 选项和 RDS API 参数	发生更改时	停机说明	支持的数据库引擎
构，该架构与本地部署和 RDS 部署均兼容。 有关更多信息，请参阅 [RDS for Oracle CDB 概述]()。				

控制台设置和说明	CLI 选项和 RDS API 参数	发生更改时	停机说明	支持的数据库引擎
架构设置 Oracle 数据库架构：CDB 或非 CDB。如果您选择 Oracle 多租户架构，则 RDS for Oracle 会将您的非 CDB 转换为使用单租户配置的 CDB。 仅当您的数据库是运行 Oracle Database 19c（具有 2021 年 4 月或更高版本的 RU）的非 CDB 时，才支持此设置。转换后，您的 CDB 包含一个初始可插拔数据库（PDB）。架构更改是永久性的，这意味着您无法将 CDB 转换回非 CDB。 **Note** 要将采用单租户配置的 CDB 转换为采用多租户配置，请再次修改您的 CDB 实例，然后为您的架构配置选择多租户配置。 有关更多信息，请参阅 [CDB 架构的单租户配置]()。	CLI 选项： `--engine oracle-ee-cdb`（Oracle 多租户） `--engine oracle-se2-cdb`（Oracle 多租户） API 参数： `Engine`	如果您选择立即应用更改，则更改立即生效。 如果您不选择立即应用更改，则更改会在下一个维护时段中生效。	在此更改期间发生停机。	Oracle

控制台设置和说明	CLI 选项和 RDS API 参数	发生更改时	停机说明	支持的数据库引擎

自动次要版本升级
 	
选择启用自动次要版本升级，以使数据库实例能够在首选次要数据库引擎版本升级可用时自动接收这些升级。这是默认行为。Amazon RDS 在维护时段内执行自动次要版本升级。如果您未选择启用自动次要版本升级，则当新的次要版本可用时，您的数据库实例不会自动升级。
 	
有关更多信息，请参阅 自动升级次要引擎版本。
	CLI 选项：
	
	`--auto-minor-version-upgrade` | `--no-auto-minor-version-upgrade`
	
	RDS API 参数：
	
	`AutoMinorVersionUpgrade`
		更改立即发生。此设置忽略“立即应用”设置。	在此更改期间，不会出现停机。	所有数据库引擎

备份复制 选择 Enable replication to another Amazon Region (启用复制到另一个 Amazon 区域)，以在其他区域中创建备份用于灾难恢复。 然后为其他备份选择目标区域。	修改数据库实例时不可用。有关使用 Amazon CLI 或 RDS API 启用跨区域备份的信息，请参阅 启用 Amazon RDS 的跨区域自动备份。	此更改会尽快异步应用。	在此更改期间，不会停机。	Oracle、PostgreSQL、SQL Server

控制台设置和说明	CLI 选项和 RDS API 参数	发生更改时	停机说明	支持的数据库引擎

备份保留期
 	
自动备份的保留天数。要禁用自动备份，请将备份保留期设置为 0。
 	
有关更多信息，请参阅 备份简介。
 	
Note
如果您使用 Amazon Backup 管理备份，则此选项不适用。有关 Amazon Backup 的信息，请参阅 Amazon 开发人员指南。
	CLI 选项：
	
	`--backup-retention-period`
	
	RDS API 参数：
	
	`BackupRetentionPeriod`
		如果您选择立即应用更改，则更改立即生效。
		
		如果不选择立即应用更改，并且您将该设置从一个非零值更改为另一个非零值，则会尽快异步应用更改。否则，更改在下一维护时段内生效。
			如果从 0 改为非零值或从非零值改为 0，则会发生停机。
			
			这适用于单可用区和多可用区数据库实例。
				所有数据库引擎

控制台设置和说明	CLI 选项和 RDS API 参数	发生更改时	停机说明	支持的数据库引擎

备份时段
 	
数据库的自动备份时间范围。备份时段是开始时间 (采用通用协调时间 (UTC)) 和持续时间 (以小时为单位)。
 	
有关更多信息，请参阅“备份简介”。
 	
Note
如果您使用 Amazon Backup 管理备份，则不会显示此选项。有关 Amazon Backup 的更多信息，请参阅 Amazon Backup 开发人员指南。
	CLI 选项：
	
	`--preferred-backup-window`
	
	RDS API 参数：
	
	`PreferredBackupWindow`
		更改会尽快异步应用。	在此更改期间，不会出现停机。	所有数据库引擎

控制台设置和说明	CLI 选项和 RDS API 参数	发生更改时	停机说明	支持的数据库引擎

证书颁发机构
 	
数据库实例使用的服务器证书的证书颁发机构（CA）。
 	
有关更多信息，请参阅 使用 SSL/TLS 加密与数据库实例或集群的连接。
	CLI 选项：
	
	`--ca-certificate-identifier`
	
	RDS API 参数：
	
	`CACertificateIdentifier`
		如果您选择立即应用更改，则更改立即生效。
		
		如果您不选择立即应用更改，则更改会在下一个维护时段中生效。
			仅当数据库引擎不支持在不重新启动的情况下轮换时，才会发生停机。您可以使用 describe-db-engine-versions Amazon CLI 命令来确定数据库引擎是否支持在不重新启动的情况下轮换。	所有数据库引擎

将标签复制到快照
 	
如果您有任何数据库实例标签，请启用此选项，以在您创建数据库快照时复制这些标签。
 	
有关更多信息，请参阅“为 Amazon RDS 资源添加标签”。
	CLI 选项：
	
	`--copy-tags-to-snapshot` 或者 `--no-copy-tags-to-snapshot`
	
	RDS API 参数：
	
	`CopyTagsToSnapshot`
		更改立即发生。此设置忽略“立即应用”设置。	在此更改期间，不会出现停机。	所有数据库引擎

控制台设置和说明	CLI 选项和 RDS API 参数	发生更改时	停机说明	支持的数据库引擎

数据库端口
 	
要用于访问数据库实例的端口。
 	
端口值不能与数据库实例所关联选项组中的选项指定的任何端口值相同。
 	
有关更多信息，请参阅“连接到 Amazon RDS 数据库实例”。
	CLI 选项：
	
	`--db-port-number`
	
	RDS API 参数：
	
	`DBPortNumber`
		更改立即发生。此设置忽略“立即应用”设置。	数据库实例立即重新启动。	所有数据库引擎

数据库引擎版本
 	
要使用的数据库引擎的版本。在您升级生产数据库实例之前，我们建议您在测试数据库实例上测试升级过程。这样做有助于验证其持续时间并验证您的应用程序。
 	
有关更多信息，请参阅 升级数据库实例引擎版本。
	CLI 选项：
	
	`--engine-version`
	
	RDS API 参数：
	
	`EngineVersion`
		如果您选择立即应用更改，则更改立即生效。
		
		如果您不选择立即应用更改，则更改会在下一个维护时段中生效。
			在此更改期间发生停机。	所有数据库引擎

数据库实例类
 	
您要使用的数据库实例类。
 	
有关更多信息，请参阅“数据库实例类”。
	CLI 选项：
	
	`--db-instance-class`
	
	RDS API 参数：
	
	`DBInstanceClass`
		如果您选择立即应用更改，则更改立即生效。
		
		如果您不选择立即应用更改，则更改会在下一个维护时段中生效。
			在此更改期间发生停机。	所有数据库引擎

控制台设置和说明	CLI 选项和 RDS API 参数	发生更改时	停机说明	支持的数据库引擎

数据库实例标识符
 	
新的数据库实例标识符。此值以一个小写字符串存储。
 	
有关数据库实例重命名效果的更多信息，请参阅[重命名数据库实例](#)。
	CLI 选项：
	
	`--new-db-instance-identifier`
	
	RDS API 参数：
	
	`NewDBInstanceIdentifier`
		如果您选择立即应用更改，则更改立即生效。
		
		如果您不选择立即应用更改，则更改会在下一个维护时段中生效。
			除非您的数据库引擎版本支持动态 SSL 上传，否则在此更改期间会出现停机。要确定您的版本是否需要重启，请运行以下 Amazon CLI 命令：
			
			`aws rds describe-db-engine-versions \ --default-only \ --engine your-db-engine \ --query 'DBEngineVersions[*].SupportsCertificateRotationWithoutRestart'`
				所有数据库引擎

控制台设置和说明	CLI 选项和 RDS API 参数	发生更改时	停机说明	支持的数据库引擎

数据库参数组
 	
要与数据库实例关联的数据库参数组。
 	
有关更多信息，请参阅“Amazon RDS 的参数组”。
	CLI 选项：
	
	`--db-parameter-group-name`
	
	RDS API 参数：
	
	`DBParameterGroupName`
		新数据库参数组将立即与数据库实例发生关联。	将新的数据库参数组与数据库实例关联时，不会出现停机。
			
			关联数据库参数组不同于在参数组中应用参数更改。只有在您手动重启数据库实例后，RDS 才在新关联的组中应用修改后的静态和动态参数设置。但是，如果在将数据库参数组与数据库实例关联之后修改该组中的动态参数，将立即应用这些参数设置，无需重启。
			
			有关更多信息，请参阅Amazon RDS 的参数组
				所有数据库引擎

控制台设置和说明	CLI 选项和 RDS API 参数	发生更改时	停机说明	支持的数据库引擎
			和重启中的数据库实例。	

专用日志卷
 	
使用专用日志卷（DLV）将数据库事务日志存储在与包含数据库表的卷不同的存储卷上。
 	
有关更多信息，请参阅 使用专用日志卷（DLV）。
	CLI 选项：
	
	`-dedicated-log-volume`
	
	RDS API 参数：
	
	`DedicatedLogVolume`
		此更改在重启数据库实例之后即会应用。	数据库实例重启时发生停机。	MariaDB、MySQL、PostgreSQL

删除保护
 	
启用删除保护以禁止删除数据库实例。
 	
有关更多信息，请参阅“删除数据库实例”。
	CLI 选项：
	
	`--deletion-protection|--no-deletion-protection`
	
	RDS API 参数：
	
	`DeletionProtection`
		更改立即发生。此设置忽略“立即应用”设置。	在此更改期间，不会出现停机。	所有数据库引擎

控制台设置和说明	CLI 选项和 RDS API 参数	发生更改时	停机说明	支持的数据库引擎

增强监控
 	
启用增强监测，以允许为运行数据库实例的操作系统实时收集指标。
 	
有关更多信息，请参阅"使用增强监控来监控操作系统指标"。
	CLI 选项：
	
	`--monitoring-interval` 和 `--monitoring-role-arn`
	
	RDS API 参数：
	
	`MonitoringInterval` 和 `MonitoringRoleArn`
		更改立即发生。此设置忽略"立即应用"设置。	在此更改期间，不会出现停机。	所有数据库引擎

控制台设置和说明	CLI 选项和 RDS API 参数	发生更改时	停机说明	支持的数据库引擎

IAM 数据库身份验证
 	
Enable IAM DB authentication（启用 IAM 数据库身份验证），通过用户和角色对数据库用户进行身份验证。
 	
有关更多信息，请参阅 适用于 MariaDB、MySQL 和 PostgreSQL 的 IAM 数据库身份验证。
	CLI 选项：
	
	`--enable-iam-database-authentication` | `--no-enable-iam-database-authentication`
	
	RDS API 参数：
	
	`EnableIAMDatabaseAuthentication`
		如果您选择立即应用更改，则更改立即生效。
		
		如果您不选择立即应用更改，则更改会在下一个维护时段中生效。
			在此更改期间，不会出现停机。	仅限 MariaDB、MySQL 和 PostgreSQL

Kerberos 身份验证
 	
选择要将数据库实例移动到的 Active Directory。在执行此操作之前此目录必须存在。如果已选择目录，则可以指定 None (无) 以从其当前目录中删除数据库实例。
 	
有关更多信息，请参阅"Kerberos 身份验证"。
	CLI 选项：
	
	`--domain` 和 `--domain-iam-role-name`
	
	RDS API 参数：
	
	`Domain` 和 `DomainIAMRoleName`
		如果您选择立即应用更改，则更改立即生效。
		
		如果您不选择立即应用更改，则更改会在下一个维护时段中生效。
			在此更改期间会出现短暂停机。	仅 Microsoft SQL Server、MySQL、Oracle 和 PostgreSQL

控制台设置和说明	CLI 选项和 RDS API 参数	发生更改时	停机说明	支持的数据库引擎

许可模式
 	
选择自带许可以使用 Db2 和 Oracle 的许可证。
 	
选择 license-included (附带许可) 可使用 Microsoft SQL Server 或 Oracle 的一般许可协议。
 	
有关更多信息，请参阅Amazon RDS for Db2 许可选项、授予 Amazon RDS 上 Microsoft SQL Server 的许可和RDS for Oracle 许可选项。
	CLI 选项：
	
	`--license-model`
	
	RDS API 参数：
	
	`LicenseModel`
		如果您选择立即应用更改，则更改立即生效。
		
		如果您不选择立即应用更改，则更改会在下一个维护时段中生效。
			在此更改期间发生停机。	仅 Microsoft SQL Server 和 Oracle

日志导出
 	
要发布到 Amazon CloudWatch Logs 的数据库日志文件的类型。
 	
有关更多信息，请参阅“将数据库日志发布到 Amazon CloudWatch Logs”。
	CLI 选项：
	
	`--cloudwatch-logs-export-configuration`
	
	RDS API 参数：
	
	`CloudwatchLogsExportConfiguration`
		更改立即发生。此设置忽略“立即应用”设置。	在此更改期间，不会出现停机。	所有数据库引擎

控制台设置和说明	CLI 选项和 RDS API 参数	发生更改时	停机说明	支持的数据库引擎

维护窗口
 	
进行系统维护的时间范围。维护系统包括升级 (如果适用)。维护时段是开始时间 (采用通用协调时间 (UTC)) 和持续时间 (以小时为单位)。
 	
如果将时段设置为当前时间，则当前时间与该时段结束之间必须相隔至少 30 分钟。此时间安排有助于确保所有待处理的更改得到应用。
 	
有关更多信息，请参阅 [Amazon RDS 维护时段]()。
	CLI 选项：
	
	`--preferred-maintenance-window`
	
	RDS API 参数：
	
	`PreferredMaintenanceWindow`
		更改立即发生。此设置忽略“立即应用”设置。	如果有一个或多个待处理的操作导致停机，并且维护时段经过更改，加入了当前时间，则立即应用这些待处理的操作并会出现停机。	所有数据库引擎

控制台设置和说明	CLI 选项和 RDS API 参数	发生更改时	停机说明	支持的数据库引擎

在 Amazon Secrets Manager 中管理主凭证
 	
选择在 Amazon Secrets Manager 中管理主凭证，以在 Secrets Manager 的密钥中管理主用户密码。
 	
（可选）选择用于保护密钥的 KMS 密钥。请从您的账户的 KMS 密钥中进行选择，或输入来自其他账户的密钥。
 	
如果 RDS 已在管理数据库实例的主用户密码，则可以通过选择 Rotate secret immediately（立即轮换密钥）来轮换主用户密码。
 	
有关更多信息，请参阅 使用 Amazon RDS 和 Amazon Secrets Manager 管理密码。
	CLI 选项：
	
	`--manage-master-user-password` | `--no-manage-master-user-password`
	
	`--master-user-secret-kms-key-id`
	
	`--rotate-master-user-password` | `--no-rotate-master-user-password`
	
	RDS API 参数：
	
	`ManageMasterUserPassword`
		如果您开启或关闭自动主用户密码管理，则更改会立即发生。此更改忽略立即应用设置。
		
		如果要轮换主用户密码，则必须指定立即应用更改。
			在此更改期间，不会停机。	所有数据库引擎

控制台设置和说明	CLI 选项和 RDS API 参数	发生更改时	停机说明	支持的数据库引擎

	`MasterUserSecretKmsKeyId`
	
	`RotateMasterUserPassword`			

多可用区部署
 	
选择是以在多个可用区中部署数据库实例。否则，选择否。
 	
有关更多信息，请参阅“配置和管理 Amazon RDS 的多可用区部署”。
	CLI 选项：
	
	`--multi-az|--no-multi-az`
	
	RDS API 参数：
	
	`MultiAZ`
		如果您选择立即应用更改，则更改立即生效。
		
		如果您不选择立即应用更改，则更改会在下一个维护时段中生效。
			在此更改期间，不会出现停机。不过，可能会对性能产生影响。有关更多信息，请参阅 将 Amazon RDS 的数据库实例转换为多可用区部署。	所有数据库引擎

控制台设置和说明	CLI 选项和 RDS API 参数	发生更改时	停机说明	支持的数据库引擎

网络类型
 	
数据库实例支持的 IP 寻址协议。
 	
IPv4，规定资源只能通过 Internet 协议版本 4 (IPv4) 寻址协议与数据库实例通信。
 	
双堆栈模式，规定资源可通过 IPv4 和/或 Internet 协议版本 6 (IPv6) 与数据库实例通信。如果您有任何必须通过 IPv6 寻址协议与数据库实例通信的资源，请使用双堆栈模式。此外，请确保将 IPv6 CIDR 块与指定的数据库子网组中的所有子网进行关联。
 	
有关更多信息，请参阅 [Amazon RDS IP 寻址](#)。
	CLI 选项：
	
	`--network-type`
	
	RDS API 参数：
	
	`NetworkType`
		如果您选择立即应用更改，则更改立即生效。
		
		如果您不选择立即应用更改，则更改会在下一个维护时段中生效。
			在此更改期间，可能会出现停机。	所有数据库引擎

新建主密码
 	
您的主用户密码。密码必须包含 8–41 个字母数字字符。
	CLI 选项：
	
	`--master-user-password`
	
	RDS API 参数：
	
	`MasterUserPassword`
		更改会尽快异步应用。此设置忽略“立即应用”设置。	在此更改期间，不会出现停机。	所有数据库引擎

控制台设置和说明	CLI 选项和 RDS API 参数	发生更改时	停机说明	支持的数据库引擎

选项组
 	
要与数据库实例关联的选项组。
 	
有关更多信息，请参阅“使用选项组”。
	CLI 选项：
	
	`--option-group-name`
	
	RDS API 参数：
	
	`OptionGroupName`
		如果您选择立即应用更改，则更改立即生效。
		
		如果您不选择立即应用更改，则更改会在下一个维护时段中生效。
			在此更改期间，不会出现停机。一个例外是将 MariaDB 审计插件添加到 RDS for MariaDB 或 RDS for MySQL 数据库实例中，这可能会导致中断。	所有数据库引擎

Performance Insights
 	
启用 Performance Insights 以监控数据库实例负载，以便您可以分析数据库性能和解决数据库性能问题。
 	
Performance Insights 不适用于某些数据库引擎版本和数据库实例类。如果 Performance Insights 部分不适用于您的数据库实例，则控制台中不会显示该部分。
 	
有关更多信息，请参阅 在 Amazon RDS 上使用性能详情监控数据库负载 和 支持性能详情的 Amazon RDS 数据库引擎、区域和实例类。
	CLI 选项：
	
	`--enable-performance-insights` | `--no-enable-performance-insights`
	
	RDS API 参数：
	
	`EnablePerformanceInsights`
		更改立即发生。此设置忽略“立即应用”设置。	在此更改期间，不会停机。	全部，但 Db2 除外

控制台设置和说明	CLI 选项和 RDS API 参数	发生更改时	停机说明	支持的数据库引擎

Performance InsightsAmazon KMS key
 	
用于加密 Performance Insights 数据的 Amazon KMS 的 Amazon KMS key 密钥标识符。密钥标识符是 Amazon Resource Name (ARN)、Amazon KMS 密钥标识符或 KMS 的密钥别名。
 	
有关更多信息，请参阅 为 Amazon RDS 打开和关闭性能详情。
	CLI 选项：
	
	`--performance-insights-kms-key-id`
	
	RDS API 参数：
	
	`PerformanceInsightsKMSKeyId`
		更改立即发生。此设置忽略“立即应用”设置。	在此更改期间，不会停机。	全部，但 Db2 除外

Performance Insights 保留期
 	
保留 Performance Insights 数据的时间（天）。免费套餐中的保留设置为 Default (7 days) [原定设置（7 天）]。要将性能数据保留更长时间，请指定 1–24 个月。有关保留期的更多信息，请参阅性能详情的定价和数据留存。
 	
有关更多信息，请参阅 为 Amazon RDS 打开和关闭性能详情。
	CLI 选项：
	
	`--performance-insights-retention-period`
	
	RDS API 参数：
	
	`PerformanceInsightsRetentionPeriod`
		更改立即发生。此设置忽略“立即应用”设置。	在此更改期间，不会停机。	全部，但 Db2 除外

控制台设置和说明	CLI 选项和 RDS API 参数	发生更改时	停机说明	支持的数据库引擎

处理器功能
 	
数据库实例的数据库实例类的 CPU 内核数和每个内核的线程数。
 	
有关更多信息，请参阅“在 RDS for Oracle 中为数据库实例类配置处理器”。
	CLI 选项：
	
	`--processor-features` 和 `--use-default-processor-features` | `--no-use-default-processor-features`
	
	RDS API 参数：
	
	`ProcessorFeatures` 和 `UseDefaultProcessorFeatures`
		如果您选择立即应用更改，则更改立即生效。
		
		如果您不选择立即应用更改，则更改会在下一个维护时段中生效。
			在此更改期间发生停机。	仅 Oracle

控制台设置和说明	CLI 选项和 RDS API 参数	发生更改时	停机说明	支持的数据库引擎

预置 IOPS
 	
数据库实例的预调配 IOPS（每秒 I/O 操作数）值。仅当您为 Storage type（存储类型）选择以下类型之一时，此设置才可用：
 	
• General purpose SSD (gp3) [通用型 SSD（gp3）]
• Provisioned IOPS SSD (io1) [预调配 IOPS（io1）]
• 预调配 IOPS SSD（io2）
 	
有关更多信息，请参阅the section called “预置 IOPS 存储” 和the section called “gp3 存储（推荐）”。
	CLI 选项：
	
	`--iops`
	
	RDS API 参数：
	
	`Iops`
		如果您选择立即应用更改，则更改立即生效。
		
		如果您不选择立即应用更改，则更改会在下一个维护时段中生效。
			在此更改期间，不会出现停机。	所有数据库引擎

控制台设置和说明	CLI 选项和 RDS API 参数	发生更改时	停机说明	支持的数据库引擎

公有访问权限
 	
可公开访问：为数据库实例提供公有 IP 地址，这表示可以在 VPC 外部访问该实例。要可供公开访问，数据库实例还必须在 VPC 的公有子网中。
 	
不可公开访问：只能从 VPC 内部访问数据库实例。
 	
有关更多信息，请参阅 对互联网隐藏 VPC 中的数据库实例。
 	
要从 VPC 外部连接到数据库实例，该数据库实例必须可供公开访问。此外，必须使用数据库实例安全组的入站规则授予访问权限。此外，还必须满足其他要求。有关更多信息，请参阅 无法连接到 Amazon RDS 数据库实例。
 	
如果您的数据库实例不可供公开访问，还可以使用 Amazon Site-to-Site VPN 连接或 Amazon Direct Connect 连接从私有网络访问该实例。有关更多信息，请参阅 互联网络流量隐私。
	CLI 选项：
	
	`--publicly-accessible | --no-publicly-accessible`
	
	RDS API 参数：
	
	`PubliclyAccessible`
		更改立即发生。此设置忽略“立即应用”设置。	在此更改期间，不会出现停机。	所有数据库引擎

控制台设置和说明	CLI 选项和 RDS API 参数	发生更改时	停机说明	支持的数据库引擎

安全组
 	
要与数据库实例关联的 VPC 安全组。
 	
有关更多信息，请参阅“使用安全组控制访问权限”。
	CLI 选项：
	
	`--vpc-security-group-ids`
	
	RDS API 参数：
	
	`VpcSecurityGroupIds`
		更改会尽快异步应用。此设置忽略“立即应用”设置。	在此更改期间，不会出现停机。	所有数据库引擎

Storage autoscaling (存储自动扩展)
 	
Enable storage autoscaling (启用存储自动扩展)，使 Amazon RDS 能够在需要时自动增加存储空间，以避免数据库实例耗尽存储空间。
 	
使用最大存储阈值设置 Amazon RDS 自动将数据库实例存储增加到的上限。默认值为 1,000 GiB。
 	
有关更多信息，请参阅“使用 Amazon RDS 存储自动扩展功能自动管理容量”。
	CLI 选项：
	
	`--max-allocated-storage`
	
	RDS API 参数：
	
	`MaxAllocatedStorage`
		更改立即发生。此设置忽略“立即应用”设置。	在此更改期间，不会出现停机。	所有数据库引擎

控制台设置和说明	CLI 选项和 RDS API 参数	发生更改时	停机说明	支持的数据库引擎

Storage throughput（存储吞吐量）
 	
数据库实例的新存储吞吐量值。仅当您为 Storage type（存储类型）选择 General purpose SSD (gp3) [通用型 SSD（gp3）]时，此设置才可用：
 	
有关更多信息，请参阅 the section called “gp3 存储（推荐）”。
	CLI 选项：
	
	`--storage-throughput`
	
	RDS API 参数：
	
	`StorageThroughput`
		如果您选择立即应用更改，则更改立即生效。
		
		如果您不选择立即应用更改，则更改会在下一个维护时段中生效。
			在此更改期间，不会出现停机。	所有数据库引擎

控制台设置和说明	CLI 选项和 RDS API 参数	发生更改时	停机说明	支持的数据库引擎

存储类型
 	
要使用的存储类型。
 	
如果您选择 General Purpose SSD (gp3) [通用型 SSD（gp3）]，则可以在 Advanced settings（高级设置）下预调配额外的 Provisioned IOPS（预调配 IOPS）和 Storage throughput（存储吞吐量）。
 	
如果您选择预调配 IOPS SSD（io1）或预调配 IOPS SSD（io2），请输入预调配 IOPS 值。
 	
在 Amazon RDS 开始修改数据库实例以更改存储大小或类型后，您无法在 6 小时内提交另一个请求以更改存储大小、性能或类型。
 	
有关更多信息，请参阅 [Amazon RDS 存储类型](#)。
	CLI 选项：
	
	`--storage-type`
	
	RDS API 参数：
	
	`StorageType`
		如果您选择立即应用更改，则更改立即生效。
		
		如果您不选择立即应用更改，则更改会在下一个维护时段中生效。
			进程开始后，以下更改均会导致短暂停机。此后，在更改发生时，您可以正常使用数据库。
			
			• 从 General Purpose (SSD) [通用型 (SSD)] 或 Provisioned IOPS (SSD) [预调配 IOPS (SSD)] 到 Magnetic（磁介质）。
			• 从 Magnetic（磁介质）到 General Purpose (SSD) [通用型 (SSD)] 或 Provisioned IOPS (SSD) [预调配 IOPS (SSD)]。
				所有数据库引擎

控制台设置和说明	CLI 选项和 RDS API 参数	发生更改时	停机说明	支持的数据库引擎

DB subnet group（数据库子网组）
 	
数据库实例的数据库子网组。您可以使用此设置将数据库实例移动到其他 VPC。
 	
有关更多信息，请参阅 Amazon VPC 和 Amazon RDS。
	CLI 选项：
	
	`--db-subnet-group-name`
	
	RDS API 参数：
	
	`DBSubnetGroupName`
		如果您选择立即应用更改，则更改立即生效。
		
		如果您不选择立即应用更改，则更改会在下一个维护时段中生效。
			在此更改期间发生停机。	所有数据库引擎

维护数据库实例

Amazon RDS 会定期对 Amazon RDS 资源执行维护。以下主题介绍了这些维护操作以及如何应用它们。

数据库实例维护更新概述

维护通常涉及对数据库实例中以下资源的更新：

- 底层硬件
- 底层操作系统（OS）
- 数据库引擎版本

针对操作系统的更新最常见的原因是安全问题。建议您尽快进行更新。有关操作系统更新的更多信息，请参阅the section called “应用更新”。

主题

- 维护更新期间的离线资源
- 推迟的数据库实例修改
- DescribePendingMaintenanceActions API 的最终一致性

维护更新期间的离线资源

一些维护项目要求 Amazon RDS 使您的数据库实例脱机一小段时间。要求资源脱机的维护项目包括必需的操作系统或数据库修补。仅对与安全性和实例可靠性相关的修补程序自动安排必需的修补。此类补丁很少发生，通常每隔几个月发生一次。它所需要的维护时间很少超过维护窗口的一小部分。

推迟的数据库实例修改

您已选择不立即应用的延迟的数据库实例修改会在维护时段内应用。例如，您可以选择在维护时段内更改数据库实例类或参数组。您使用等待重启设置指定的此类修改不会显示在等待维护列表中。有关修改数据库实例的信息，请参阅修改 Amazon RDS 数据库实例。

要查看下一个维护时段待处理的修改，请使用 describe-db-instances Amazon CLI 命令并选中 `PendingModifiedValues` 字段。

DescribePendingMaintenanceActions API 的最终一致性

Amazon RDS `DescribePendingMaintenanceActions` API 采用最终一致性模型。这意味着，`DescribePendingMaintenanceActions` 命令的结果可能不会立即对所有后续 RDS 命令可见。当您使用之前的 API 命令后立即使用 `DescribePendingMaintenanceActions` 时，请记住这一点。

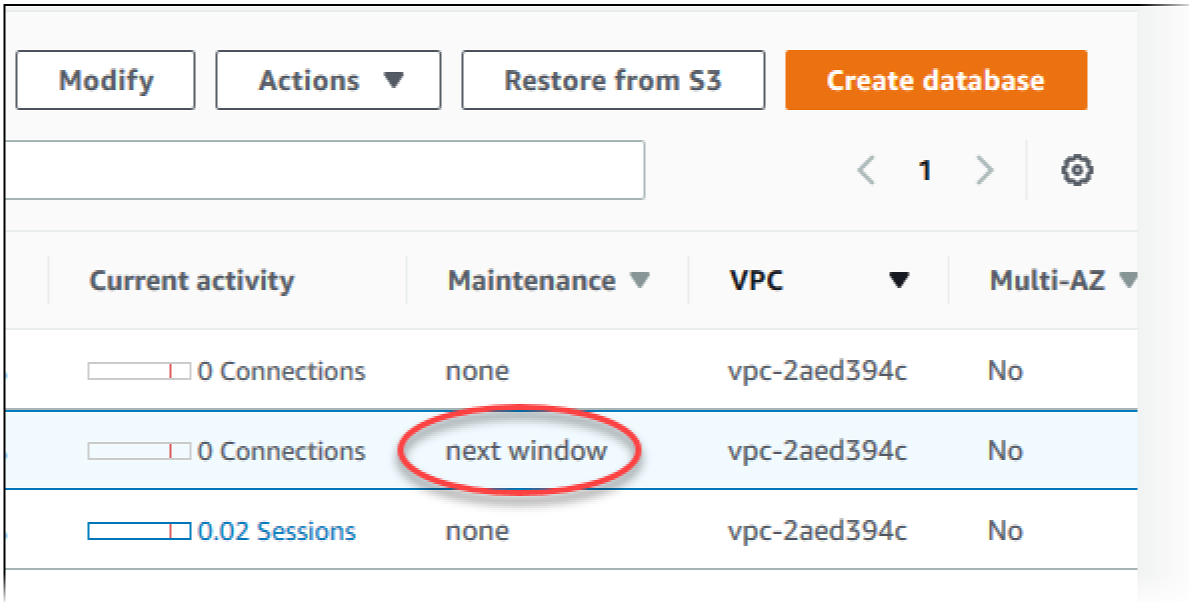
最终一致性可能会影响您管理维护更新的方式。例如，如果您运行 `ApplyPendingMaintenanceActions` 命令来更新数据库实例的数据库引擎版本，则该版本最终将对 `DescribePendingMaintenanceActions` 可见。在这种情况下，`DescribePendingMaintenanceActions` 可能表明维护操作未被应用，即使已应用也是如此。

要管理最终一致性，您可以执行以下操作：

- 请先确认数据库实例的状态，然后运行命令来对其进行修改。使用指数回退算法运行相应的 `DescribePendingMaintenanceActions` 命令，来确保有足够的时间让前一个命令传播遍整个系统。为此，请重复运行 `DescribePendingMaintenanceActions` 命令，以几秒钟的等待时间开始，然后逐渐增加达到五分钟的等待时间。
- 增加后续命令之间的等待时间，即使 `DescribePendingMaintenanceActions` 命令返回准确的响应，也是如此。应用指数回退算法，以几秒钟的等待时间开始，然后逐渐增加达到大约五分钟的等待时间。

查看待处理维护更新

通过使用 RDS 控制台、Amazon CLI 或 RDS API 来查看维护更新是否可用于数据库实例。如果有可用更新，将在 Amazon RDS 控制台上数据库实例的维护列中显示，如下图中所示。



如果没有维护更新可用于数据库实例，则它的列值为无。

如果有维护更新可用于数据库实例，则可能为以下列值：

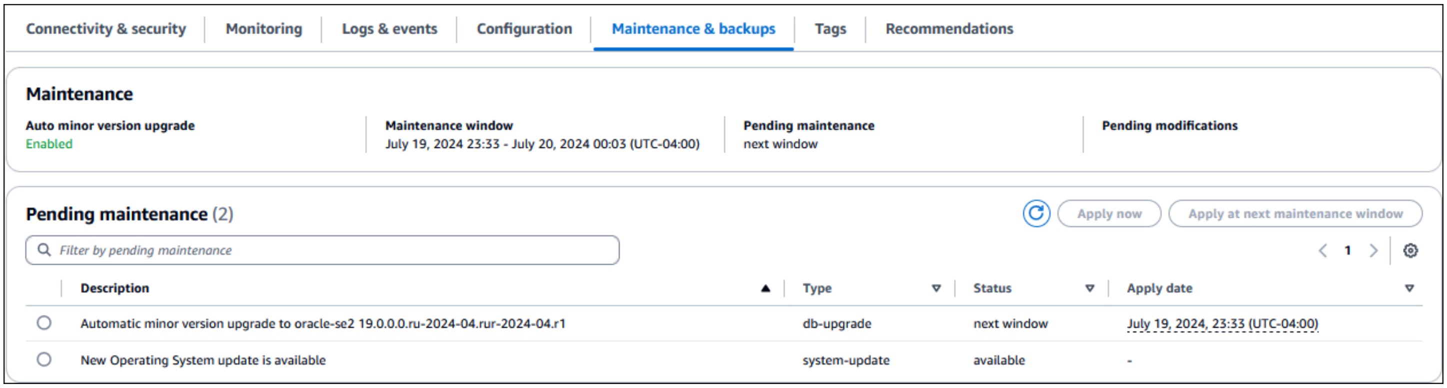
- 必需 – 维护操作将应用于资源且不能无限期推迟。
- 可用 – 有可执行的维护操作，但不会自动将其应用于资源。您可以手动应用它。
- 下一个时段 – 维护操作将在下一个维护时段应用于资源。
- 正在进行 – 维护操作正在应用于资源。

如果有可用更新，您可以执行以下操作之一：

- 如果维护值为下一个时段，可通过从操作中选择推迟升级来推迟启动维护操作。无法推迟已经启动的维护操作。
- 立即应用维护操作。
- 在下一个维护时段内应用维护操作。
- 不执行任何操作。

要执行操作，请使用 Amazon Web Services Management Console

1. 选择数据库实例以显示其详细信息。
2. 选择维护和备份。将显示待执行的维护操作。
3. 选择要执行的操作，然后选择何时应用该操作。



维护时段确定待执行的操作何时开始，但不限制这些操作的总运行时间。维护操作不保证在维护时段结束前完成，可以在超出指定的结束时间后继续。有关更多信息，请参阅 Amazon RDS 维护时段。

可以通过运行 describe-pending-maintenance-actions Amazon CLI 命令，来查看维护更新是否可用于您的数据库实例。

有关应用维护更新的信息，请参阅对数据库实例应用更新。

适用于 Amazon RDS 的维护操作

以下维护操作适用于 RDS 数据库实例：

- `ca-certificate-rotation` – 更新数据库实例的 Amazon RDS 证书机构证书。
- `db-upgrade` – 升级数据库实例的数据库引擎版本。
- `hardware-maintenance` – 对数据库实例的底层硬件进行维护。
- `system-update` – 更新数据库实例的操作系统。

多可用区部署的维护

采用多可用区部署的方式运行数据库实例可进一步降低维护事件的影响。此结果是因为 Amazon RDS 通过执行以下步骤来应用操作系统更新：

1. 执行备用实例的维护。
2. 将备用实例提升为主实例。
3. 执行原主实例的维护，该实例将变成新的备用实例。

如果您在多可用区部署中为数据库实例升级数据库引擎，Amazon RDS 会同时修改主数据库实例和辅助数据库实例。在这种情况下，多可用区部署中的主数据库实例和辅助数据库实例在升级过程中都不可用。此操作会导致停机，直至升级完成。停机持续时间因数据库实例的大小而异。

如果需要应用底层操作系统补丁，则需要短暂的多可用区失效转移，才能将补丁应用于主数据库实例。这种失效转移通常持续不到一分钟。

如果您的数据库实例运行 RDS for MySQL、RDS for PostgreSQL 或 RDS for MariaDB，您可以使用蓝绿部署，显著减少升级所需的停机时间。有关更多信息，请参阅 使用 Amazon RDS 蓝绿部署进行数据库更新。如果您在多可用区部署中升级 RDS for SQL Server 或 RDS Custom for SQL Server 数据库实例，则 Amazon RDS 会执行滚动升级，因此只会在失效转移期间出现中断。有关更多信息，请参阅 多可用区注意事项。

有关多可用区部署的更多信息，请参阅配置和管理 Amazon RDS 的多可用区部署。

Amazon RDS 维护时段

维护时段是每周时间间隔，在此期间会应用任何系统更改。每个数据库实例都具有每周维护时段。可以利用维护时段来控制何时进行修改和软件修补。有关调整维护时段的更多信息，请参阅调整首选数据库实例维护时段。

在应用维护时，RDS 会使用您的数据库实例上的一些资源。您可观察到对性能的影响甚微。对于数据库实例来说，在极少数情况下，可能需要多可用区故障转移才能完成维护更新。

如果在给定的周内安排了维护事件，则将在您确定的 30 分钟维护时段内启动维护。大部分维护事件也将在 30 分钟的维护时段内完成，但较大的维护事件可能需要 30 分钟以上的时间才能完成。数据库实例停止后，维护时段将暂停。

这个 30 分钟维护时段是随机从每个地区的 8 小时时间段中选择出来的。如果在创建数据库实例时未指定维护时段，则 RDS 在该星期内随机选择的某一天中分配 30 分钟的维护时段。

下表显示的是分配了默认维护窗口的各个 Amazon Web Services 区域的时间段。

区域名称	区域	时间段
美国东部（弗吉尼亚州北部）	us-east-1	03:00–11:00 UTC
美国东部（俄亥俄州）	us-east-2	03:00–11:00 UTC

区域名称	区域	时间段
美国西部（加利福尼亚北部）	us-west-1	06:00–14:00 UTC
美国西部（俄勒冈州）	us-west-2	06:00–14:00 UTC
非洲（开普敦）	af-south-1	03:00–11:00 UTC
亚太地区（香港）	ap-east-1	06:00–14:00 UTC
亚太地区（海得拉巴）	ap-south-2	06:30–14:30 UTC
亚太地区（雅加达）	ap-southeast-3	08:00–16:00 UTC
亚太地区（马来西亚）	ap-southeast-5	09:00–17:00 UTC
亚太地区（墨尔本）	ap-southeast-4	11:00–19:00 UTC
亚太地区（孟买）	ap-south-1	06:00–14:00 UTC
Asia Pacific (Osaka)	ap-northeast-3	22:00–23:59 UTC
Asia Pacific (Seoul)	ap-northeast-2	13:00–21:00 UTC
亚太地区（新加坡）	ap-southeast-1	14:00–22:00 UTC
亚太地区（悉尼）	ap-southeast-2	12:00–20:00 UTC
亚太地区（东京)	ap-northeast-1	13:00–21:00 UTC
加拿大（中部）	ca-central-1	03:00–11:00 UTC
加拿大西部（卡尔加里）	ca-west-1	18:00–02:00 UTC
中国（北京）	cn-north-1	06:00–14:00 UTC

区域名称	区域	时间段
中国（宁夏）	cn-northwest-1	06:00–14:00 UTC
Europe (Frankfurt)	eu-central-1	21:00–05:00 UTC
欧洲地区（爱尔兰）	eu-west-1	22:00–06:00 UTC
欧洲地区（伦敦）	eu-west-2	22:00–06:00 UTC
欧洲（米兰）	eu-south-1	02:00–10:00 UTC
欧洲（巴黎）	eu-west-3	23:59–07:29 UTC
欧洲（西班牙）	eu-south-2	02:00–10:00 UTC
Europe (Stockholm)	eu-north-1	23:00–07:00 UTC
欧洲（苏黎世）	eu-central-2	02:00–10:00 UTC
以色列（特拉维夫）	il-central-1	03:00–11:00 UTC
中东（巴林）	me-south-1	06:00–14:00 UTC
中东（阿联酋）	me-central-1	05:00–13:00 UTC
南美洲（圣保罗）	sa-east-1	00:00–08:00 UTC
Amazon GovCloud（美国东部）	us-gov-east-1	17:00–01:00 UTC
Amazon GovCloud（美国西部）	us-gov-west-1	06:00–14:00 UTC

主题

- 调整首选数据库实例维护时段

调整首选数据库实例维护时段

维护时段应当选在使用量最小的时段上，因而可能必须不时予以修改。数据库实例只会在应用系统更改（例如，数据库实例类中的更改）而需要中断的期间内不可用。数据库实例仅在进行必要更改所需的最短时间内不可用。

在以下示例中，将调整数据库实例的首选维护时段。

对于此示例，我们假设存在名为 mydbinstance 的数据库实例，它首选的维护时段为 UTC 时间“Sun:05:00–Sun:06:00”。

控制台

调整首选维护时段

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择数据库，然后选择您希望修改的数据库实例。
3. 选择修改。将显示修改数据库实例页面。
4. 在维护部分中，更新维护时段。

> **Note**
> 数据库实例的维护时段不能与备份时段重叠。如果向与备份时段重叠的维护时段内输入值，会显示一条错误消息。

5. 选择 Continue (继续)。

   在确认页面上，检查您的更改。
6. 要立即应用对维护时段的更改，请选择立即应用。
7. 选择修改数据库实例以保存更改。

   或者，选择 Back 编辑您的更改，或者选择 Cancel 取消更改。

Amazon CLI

要调整首选维护时段，请使用带下列参数的 Amazon CLI `modify-db-instance` 命令：

- `--db-instance-identifier`

- `--preferred-maintenance-window`

Example

以下代码示例将维护时段设置为周二的凌晨 4:00-4:30 (UTC)。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
--db-instance-identifier mydbinstance \
--preferred-maintenance-window Tue:04:00-Tue:04:30
```

对于 Windows：

```
aws rds modify-db-instance ^
--db-instance-identifier mydbinstance ^
--preferred-maintenance-window Tue:04:00-Tue:04:30
```

RDS API

要调整首选维护时段，请使用具有以下参数的 Amazon RDS API `ModifyDBInstance` 操作：

- `DBInstanceIdentifier`
- `PreferredMaintenanceWindow`

对数据库实例应用更新

通过 Amazon RDS，您可以选择何时应用维护操作。您可通过使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 来决定 Amazon RDS 何时应用更新。

控制台

管理数据库实例的更新

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择数据库。
3. 选择具有必需更新的数据库实例。
4. 对于操作，请选择下列选项之一：
   - 立即修补
   - 在下一个时段修补

> **Note**
>
> 如果您选择在下一个时段修补，并且以后希望延迟更新，可以选择推迟升级。如果维护操作已经启动，则无法推迟该操作。
> 要取消维护操作，请修改数据库实例并禁用 Auto minor version upgrade (自动次要版本升级)。

Amazon CLI

要将待处理的更新应用于数据库实例，请使用 apply-pending-maintenance-action Amazon CLI 命令。

Example

对于 Linux、macOS 或 Unix：

```
aws rds apply-pending-maintenance-action \
    --resource-identifier arn:aws:rds:us-west-2:001234567890:db:mysql-db \
    --apply-action system-update \
    --opt-in-type immediate
```

对于 Windows：

```
aws rds apply-pending-maintenance-action ^
    --resource-identifier arn:aws:rds:us-west-2:001234567890:db:mysql-db ^
    --apply-action system-update ^
    --opt-in-type immediate
```

> **Note**
>
> 要推迟维护操作，请为 undo-opt-in 指定 --opt-in-type。如果维护操作已启动，则无法为 undo-opt-in 指定 --opt-in-type。
>
> 要取消维护操作，请运行 [modify-db-instance](https://docs.aws.amazon.com/cli/latest/reference/rds/modify-db-instance.html) Amazon CLI 命令并指定 --no-auto-minor-version-upgrade。

要返回具有至少一个待处理更新的资源的列表，请使用 [describe-pending-maintenance-actions](https://docs.aws.amazon.com/cli/latest/reference/rds/describe-pending-maintenance-actions.html) Amazon CLI 命令。

Example

对于 Linux、macOS 或 Unix：

```
aws rds describe-pending-maintenance-actions \
    --resource-identifier arn:aws:rds:us-west-2:001234567890:db:mysql-db
```

对于 Windows：

```
aws rds describe-pending-maintenance-actions ^
    --resource-identifier arn:aws:rds:us-west-2:001234567890:db:mysql-db
```

您还可以通过指定 `describe-pending-maintenance-actions` Amazon CLI 命令的 `--filters` 参数返回数据库实例的资源列表。`--filters` 命令的格式是 `Name=filter-name,Value=resource-id,...`。

下面是筛选条件的 `Name` 参数中接受的值：

- `db-instance-id` – 接受数据库实例标识符或 Amazon Resource Names (ARN) 的列表。返回的列表只包括这些标识符或 ARN 标识的数据库实例的待处理维护操作。
- `db-cluster-id` – 接受 Amazon Aurora 的数据库集群标识符或 ARN 的列表。返回的列表只包括这些标识符或 ARN 标识的数据库集群的待处理维护操作。

例如，以下示例返回sample-instance1 和 sample-instance2 数据库实例的待处理维护操作。

Example

对于 Linux、macOS 或 Unix：

```
aws rds describe-pending-maintenance-actions \
 --filters Name=db-instance-id,Values=sample-instance1,sample-instance2
```

对于 Windows：

```
aws rds describe-pending-maintenance-actions ^
 --filters Name=db-instance-id,Values=sample-instance1,sample-instance2
```

RDS API

要将更新应用于数据库实例，请调用 Amazon RDS API ApplyPendingMaintenanceAction 操作。

要返回具有至少一个待处理更新的资源的列表，请调用 Amazon RDS API DescribePendingMaintenanceActions 操作。

为 RDS 数据库实例更新操作系统

RDS for Db2、RDS for MariaDB、RDS for MySQL、RDS for PostgreSQL、RDS for SQL Server 和 RDS for Oracle 数据库实例有时需要更新操作系统。Amazon RDS 将操作系统升级到更新的版本，以提高数据库性能和客户的整体安保状况。通常而言，更新大约需要花费 10 分钟。操作系统更新不会更改数据库实例的数据库引擎版本或数据库实例类。

操作系统更新可能是可选的，也可能是强制性的：

- 可以随时应用可选更新。虽然这些更新是可选的，但建议您定期应用它们，以使 RDS 实例集保持最新状态。RDS 不自动应用这些更新。

  要在新的可选操作系统补丁变为可用时收到通知，您可以订阅安全修补事件类别中的 RDS-EVENT-0230。有关订阅 RDS 事件的信息，请参阅 订阅 Amazon RDS 事件通知。

  > Note
  >
  > RDS-EVENT-0230 不适用于操作系统发行版升级。

- 强制更新为必需的更新且有应用日期。请制定计划以安排在此应用日期之前更新。在指定的应用日期之后，Amazon RDS 会在指定的维护时段之一内，自动将数据库实例的操作系统升级到最新版本。

> **Note**
>
> 为了履行各种合规性义务，可能需要及时了解所有可选和强制性更新。建议您在维护时段内定期应用 RDS 提供的所有更新。

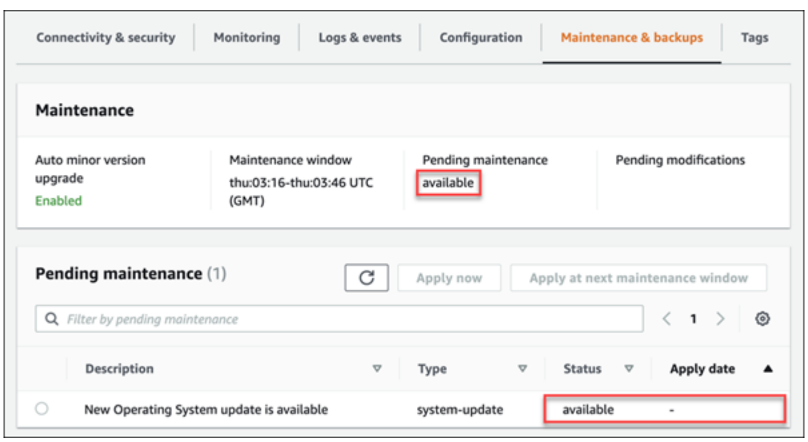
您可以使用 Amazon Web Services Management Console或 Amazon CLI 来获取有关操作系统升级类型的信息。

控制台

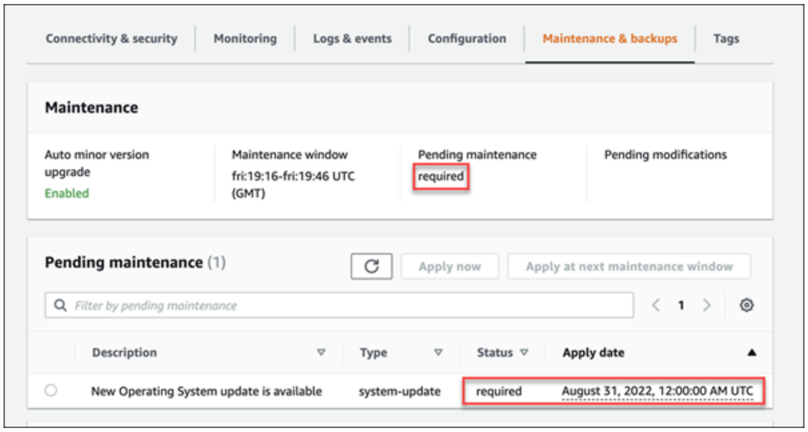
使用 Amazon Web Services Management Console获取更新信息

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases（数据库），然后选择数据库实例。
3. 选择维护和备份。
4. 在 Pending maintenance（等待维护）部分中，找到操作系统更新，然后检查 Status（状态）值。

在 Amazon Web Services Management Console中，可选的更新将其维护 Status（状态）设置为 available（可用）并且没有 Apply date（应用日期），如下图所示。



强制性更新将其维护 Status（状态）设置为 required（必需）并且有 Apply date（应用日期），如下图所示。



Amazon CLI

要从 Amazon CLI 中获取更新信息，请使用 describe-pending-maintenance-actions 命令。

```
aws rds describe-pending-maintenance-actions
```

强制性操作系统更新包括 `AutoAppliedAfterDate` 值和 `CurrentApplyDate` 值。可选的操作系统更新不包含这些值。

以下输出显示强制性操作系统更新。

```
{
  "ResourceIdentifier": "arn:aws:rds:us-east-1:123456789012:db:mydb1",
  "PendingMaintenanceActionDetails": [
    {
      "Action": "system-update",
      "AutoAppliedAfterDate": "2022-08-31T00:00:00+00:00",
      "CurrentApplyDate": "2022-08-31T00:00:00+00:00",
      "Description": "New Operating System update is available"
    }
  ]
}
```

以下输出显示可选的操作系统更新。

```
{
  "ResourceIdentifier": "arn:aws:rds:us-east-1:123456789012:db:mydb2",
  "PendingMaintenanceActionDetails": [
    {
      "Action": "system-update",
```

```
      "Description": "New Operating System update is available"
    }
  ]
}
```

操作系统更新的可用性

操作系统更新特定于数据库引擎版本和数据库实例类。因此，数据库实例在不同的时间接收或要求更新。当根据数据库实例的引擎版本和实例类，数据库实例有可用的操作系统更新时，更新将显示在控制台中。也可以通过运行 describe-pending-maintenance-actions Amazon CLI 命令或通过调用 DescribePendingMaintenanceActions RDS API 操作来查看更新。如果您的实例有可用更新，则可以按照对数据库实例应用更新中的说明更新操作系统。

升级数据库实例引擎版本

Amazon RDS 提供每个支持的数据库引擎的更高版本，因此您可以保持数据库实例最新。更新的版本可能包含对于数据库引擎的错误修复、安全增强功能和其他改进。当 Amazon RDS 支持数据库引擎的新版本时，您可以选择如何以及何时升级您的数据库实例。

有两种升级方式：主要版本升级和次要版本升级。一般而言，主引擎版本升级 可能引入与现有应用程序不兼容的更改。相比之下，次要版本升级 仅包含与现有应用程序向后兼容的更改。

对于多可用区数据库集群，只有 RDS for PostgreSQL 支持主要版本升级。支持多可用区数据库集群的所有引擎均支持次要版本升级。有关更多信息，请参阅 the section called “升级多可用区数据库集群”。

版本编号序列特定于每个数据库引擎。例如，RDS for MySQL 5.7 和 8.0 是主引擎版本，从任何 5.7 版本升级到任何 8.0 版本是主要版本升级。RDS for MySQL 版本 5.7.22 和 5.7.23 是次要版本，从 5.7.22 升级到 5.7.23 是次要版本升级。

> **Important**
>
> 升级数据库实例时无法修改该实例。升级期间，数据库实例的状态为 `upgrading`。

有关特定数据库引擎的主要版本升级和次要版本升级的更多信息，请参阅适用于您的数据库引擎的以下文档：

- 升级 MariaDB 数据库引擎
- 升级 Microsoft SQL Server 数据库引擎
- 升级 RDS for MySQL 数据库引擎
- 升级 RDS for Oracle 数据库引擎
- 升级 RDS for PostgreSQL 数据库引擎

对于主要版本升级，您必须通过 Amazon Web Services Management Console、Amazon CLI 或 RDS API 手动修改数据库引擎版本。对于次要版本升级，您可以手动修改引擎版本，也可以选择启用自动次要版本升级选项。

> **Note**
>
> 数据库引擎升级需要停机。您可以使用蓝绿部署，最大限度地减少数据库实例升级所需的停机时间。有关更多信息，请参阅 使用 Amazon RDS 蓝绿部署进行数据库更新。

主题

手动升级引擎版本

要手动升级数据库实例的引擎版本，可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API。

控制台

使用控制台升级数据库实例的引擎版本

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)，然后选择要升级的数据库实例。
3. 选择修改。将显示修改数据库实例页面。
4. 对于数据库引擎版本，选择新版本。
5. 选择继续，查看修改摘要。
6. 决定何时调度升级：
   - 要将更改放到待处理的修改队列中，请选择在下一个计划的维护时段内应用。在下一维护时段内，RDS 将应用队列中的所有待定更改。
   - 要立即应用更改，请选择立即应用。选择此选项在某些情况下可能导致中断。有关更多信息，请参阅“使用计划修改设置”。
7. 在确认页面上，检查您的更改。如果更改正确无误，请选择 Modify DB Instance（修改数据库实例）保存更改。

   或者，选择 Back 编辑您的更改，或者选择 Cancel 取消更改。

Amazon CLI

要升级数据库实例的引擎版本，请使用 CLI modify-db-instance 命令。指定以下参数：

- `--db-instance-identifier` – 数据库实例的名称。
- `--engine-version` – 数据库引擎要升级到的版本号。

  有关有效的引擎版本的信息，请使用 Amazon CLI describe-db-engine-versions 命令。
- `--allow-major-version-upgrade` – 升级主要版本。
- `--no-apply-immediately` – 在下一维护时段内应用更改。要立即应用更改，请使用 `--apply-immediately`。

Example

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier mydbinstance \
    --engine-version new_version \
    --allow-major-version-upgrade \
    --no-apply-immediately
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier mydbinstance ^
    --engine-version new_version ^
    --allow-major-version-upgrade ^
    --no-apply-immediately
```

RDS API

要升级数据库实例的引擎版本，请使用 ModifyDBInstance 操作。指定以下参数：

- `DBInstanceIdentifier` – 数据库实例的名称，例如 *mydbinstance*。
- `EngineVersion` – 数据库引擎要升级到的版本号。有关有效的引擎版本的信息，请使用 DescribeDBEngineVersions 操作。
- `AllowMajorVersionUpgrade` – 是否允许主版本升级。为此，请将该值设置为 `true`。

- `ApplyImmediately` – 是立即应用更改还是在下一个维护时段内应用更改。要立即应用更改，请将该值设置为 `true`。要在下一个维护时段内应用更改，请将该值设置为 `false`。

自动升级次要引擎版本

自动次要版本升级会定期将数据库更新到新近的数据库引擎版本。但是，升级可能并不始终包括最新的数据库引擎版本。如果您需要在特定时间将数据库保持在特定版本上，我们建议您根据所需的时间表，手动升级到您需要的数据库版本。在出现严重安全问题的情况下或当版本达到其终止支持日期时，Amazon RDS 可能会应用次要版本升级，即使您尚未启用自动次要版本升级选项也是如此。有关更多信息，请参阅特定数据库引擎的升级文档。

- RDS for PostgreSQL 的自动次要版本升级
- RDS for MySQL 的自动次要版本升级
- RDS for MariaDB 的自动次要版本升级
- Oracle 次要版本升级
- 升级 Microsoft SQL Server 数据库引擎
- Amazon RDS 上的 Db2 版本

主题

自动次要版本升级的工作原理

升级目标 是 Amazon RDS 将您的数据库升级到的数据库引擎版本。当满足以下条件时，将次要引擎版本指定为升级目标：

- 数据库正在运行的数据库引擎次要版本低于目标次要引擎版本。

  通过查看数据库详细信息页面的配置选项卡或运行 CLI 命令 `describe-db-instances`，可以找到数据库实例的当前引擎版本。

- 数据库启用了自动次要版本升级。

RDS 安排升级在维护时段中自动运行。在升级过程中，RDS 将执行以下操作：

1. 运行系统预检查，以确保数据库运行状况良好并已准备好进行升级
2. 将数据库引擎升级到目标次要引擎版本
3. 运行升级后检查
4. 将数据库升级标记为完成

自动升级会导致停机。停机时间长短取决于各种因素，包括数据库引擎类型和数据库的大小。

启用自动次要版本升级

当您执行以下任务时，您可以控制是否为数据库实例启用自动次要版本升级。

- 创建数据库实例
- 修改数据库实例
- 创建只读副本
- 从快照还原数据库实例
- 将数据库实例还原到特定时间
- 从 Amazon S3 导入数据库实例（适用于 Amazon S3 上的 MySQL 备份）

当您执行这些任务时，您可以通过以下方式控制是否为数据库实例启用自动次要版本升级：

- 使用控制台，设置自动次要版本升级选项。
- 使用 Amazon CLI，设置 `--auto-minor-version-upgrade|--no-auto-minor-version-upgrade` 选项。
- 使用 RDS API，设置 `AutoMinorVersionUpgrade` 参数。

确定维护更新的可用性

要确定维护更新（如数据库引擎版本升级）是否可用于您的数据库实例，您可以使用控制台、Amazon CLI 或 RDS API。您还可以手动升级数据库引擎实例版本并调整维护时段。有关更多信息，请参阅 维护数据库实例。

查找自动次要版本升级目标

您可以使用以下 Amazon CLI 命令，以确定特定 Amazon Web Services 区域中指定数据库引擎次要版本的当前自动次要升级目标版本。您可以在 CreateDBInstance 中的 `Engine` 参数描述中找到此命令可能的 `--engine` 值。

对于 Linux、macOS 或 Unix：

```
aws rds describe-db-engine-versions \
--engine engine \
--engine-version minor-version \
--region region \
--query "DBEngineVersions[*].ValidUpgradeTarget[*].
{AutoUpgrade:AutoUpgrade,EngineVersion:EngineVersion}" \
--output text
```

对于 Windows：

```
aws rds describe-db-engine-versions ^
--engine engine ^
--engine-version minor-version ^
--region region ^
--query "DBEngineVersions[*].ValidUpgradeTarget[*].
{AutoUpgrade:AutoUpgrade,EngineVersion:EngineVersion}" ^
--output text
```

例如，以下 Amazon CLI 命令可确定美国东部（俄亥俄）Amazon 区域 (us-east-2) 中 MySQL 次要版本 8.0.11 的自动次要升级目标。

对于 Linux、macOS 或 Unix：

```
aws rds describe-db-engine-versions \
--engine mysql \
--engine-version 8.0.11 \
--region us-east-2 \
--query "DBEngineVersions[*].ValidUpgradeTarget[*].
{AutoUpgrade:AutoUpgrade,EngineVersion:EngineVersion}" \
--output table
```

对于 Windows：

```
aws rds describe-db-engine-versions ^
--engine mysql ^
--engine-version 8.0.11 ^
--region us-east-2 ^
--query "DBEngineVersions[*].ValidUpgradeTarget[*].
{AutoUpgrade:AutoUpgrade,EngineVersion:EngineVersion}" ^
--output table
```

输出类似于以下内容。

```
----------------------------------
|    DescribeDBEngineVersions    |
+--------------+-----------------+
|  AutoUpgrade |  EngineVersion  |
+--------------+-----------------+
|  False       |  8.0.15         |
|  False       |  8.0.16         |
|  False       |  8.0.17         |
|  False       |  8.0.19         |
|  False       |  8.0.20         |
|  False       |  8.0.21         |
|  True        |  8.0.23         |
|  False       |  8.0.25         |
+--------------+-----------------+
```

在此示例中，`AutoUpgrade` 值对于 MySQL 版本 8.0.23 为 `True`。因此，自动次要升级目标是 MySQL 版本 8.0.23，它在输出中突出显示。

> **Important**
>
> 如果您计划在不久后将 RDS for PostgreSQL 数据库实例迁移到 Aurora PostgreSQL 数据库集群，我们强烈建议您在规划期间的早期对数据库实例禁用自动次要版本升级。如果 Aurora PostgreSQL 尚不支持 RDS for PostgreSQL 版本，迁移到 Aurora PostgreSQL 可能会延迟。有关 Aurora PostgreSQL 版本的信息，请参阅 [Amazon Aurora PostgreSQL 的引擎版本]。

重命名数据库实例

您可以使用Amazon Web Services Management Console、Amazon CLI `modify-db-instance` 命令或 Amazon RDS API `ModifyDBInstance` 操作重命名数据库实例。重命名数据库实例会产生深远的影响。以下是您在重命名数据库实例之前需了解的注意事项列表。

- 当您重命名数据库实例时，数据库实例的终端节点会发生更改，因为 URL 包含了您分配给数据库实例的名称。您应当始终将流量从旧的 URL 重定向到新的。
- 当您重命名数据库实例时，数据库实例使用的旧的 DNS 名称会立刻被删除，尽管它可能会在缓存中保留几分钟。重命名数据库实例的新 DNS 名称大约在 10 分钟之内生效。重命名的数据库实例在新名称生效前不可用。
- 重命名实例时，不能使用现有的数据库实例名称。
- 重命名后，所有与数据库实例关联的只读副本会保持与该实例的关联。例如，假设您有一个为生产数据库服务的数据库实例，而该实例有多个关联的只读副本。如果您重命名该数据库实例，然后在生产环境中将其替换为数据库快照，则您重命名的数据库实例仍会拥有与它关联的只读副本。
- 如果您重新使用数据库实例的名称，则与数据库实例的名称关联的指标和事件保持不变。例如，如果提升一个只读副本，然后将其重命名为以前主数据库实例的名称，则与该主数据库实例关联的事件和指标与重命名的实例关联。
- 无论是否重命名，数据库实例标签会始终与数据库实例关联在一起。
- 对于重命名的数据库实例，数据库快照也会保留下来。

> **Note**
>
> 数据库实例是在云中运行的独立数据库环境。一个数据库实例可承载多个数据库，或一个带有多个架构的 Oracle 数据库。有关更改数据库名称的信息，请参阅数据库引擎文档。

重命名以替换现有的数据库实例

重命名数据库实例最常见的原因是要提升只读副本或从数据库快照或时间点恢复 (PITR) 还原数据。通过重命名数据库，您无需更改引用该数据库实例的任何应用程序代码，就可以替换数据库实例。在这些情况下，请执行以下操作：

1. 停止传向主数据库实例的所有流量。这可能包括重定向访问数据库实例中数据库的流量，或采取阻止流量访问数据库实例中数据库的其他方法。

2. 在重命名主数据库实例时，请使用一个表示它不再是本主题下文所述的主数据库实例的名称。
3. 通过从数据库快照还原或提升只读副本而创建一个新的主数据库实例，然后将新实例命名为此前主数据库实例的名称。
4. 将所有只读副本与新的主数据库实例关联。

如果您删除旧的主数据库实例，就必须删除旧的主数据库实例中所有不再需要的数据库快照。

有关提升只读副本的信息，请参阅 将只读副本提升为独立的数据库实例。

> **Important**
>
> 重命名之后，数据库实例重启。对于启用了 AlwaysOn 或镜像选项的 RDS for SQL Server 多可用区实例，在重命名操作后重启实例时，预计会进行失效转移。

控制台

要重命名数据库实例，请执行以下操作：

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。
3. 选择要重命名的数据库实例。
4. 选择修改。
5. 在设置中，为数据库实例标识符输入一个新名称。
6. 选择 Continue (继续)。
7. 要立即应用更改，请选择立即应用。选择此选项在某些情况下可能导致中断。有关更多信息，请参阅“修改 Amazon RDS 数据库实例”。
8. 在确认页面上，检查您的更改。如果更改正确无误，请选择 Modify DB Instance (修改数据库实例) 保存更改。

   或者，选择 Back (返回) 编辑您的更改，或者选择 Cancel (取消) 取消更改。

Amazon CLI

要重命名数据库实例，请使用 Amazon CLI 命令 [modify-db-instance]。提供当前 `--db-instance-identifier` 值并为 `--new-db-instance-identifier` 参数提供数据库实例的新名称。

Example

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier DBInstanceIdentifier \
    --new-db-instance-identifier NewDBInstanceIdentifier
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier DBInstanceIdentifier ^
    --new-db-instance-identifier NewDBInstanceIdentifier
```

RDS API

要重命名数据库实例，请结合以下参数调用 Amazon RDS API 操作 [ModifyDBInstance]：

- `DBInstanceIdentifier` – 实例的现有名称
- `NewDBInstanceIdentifier` – 实例的新名称

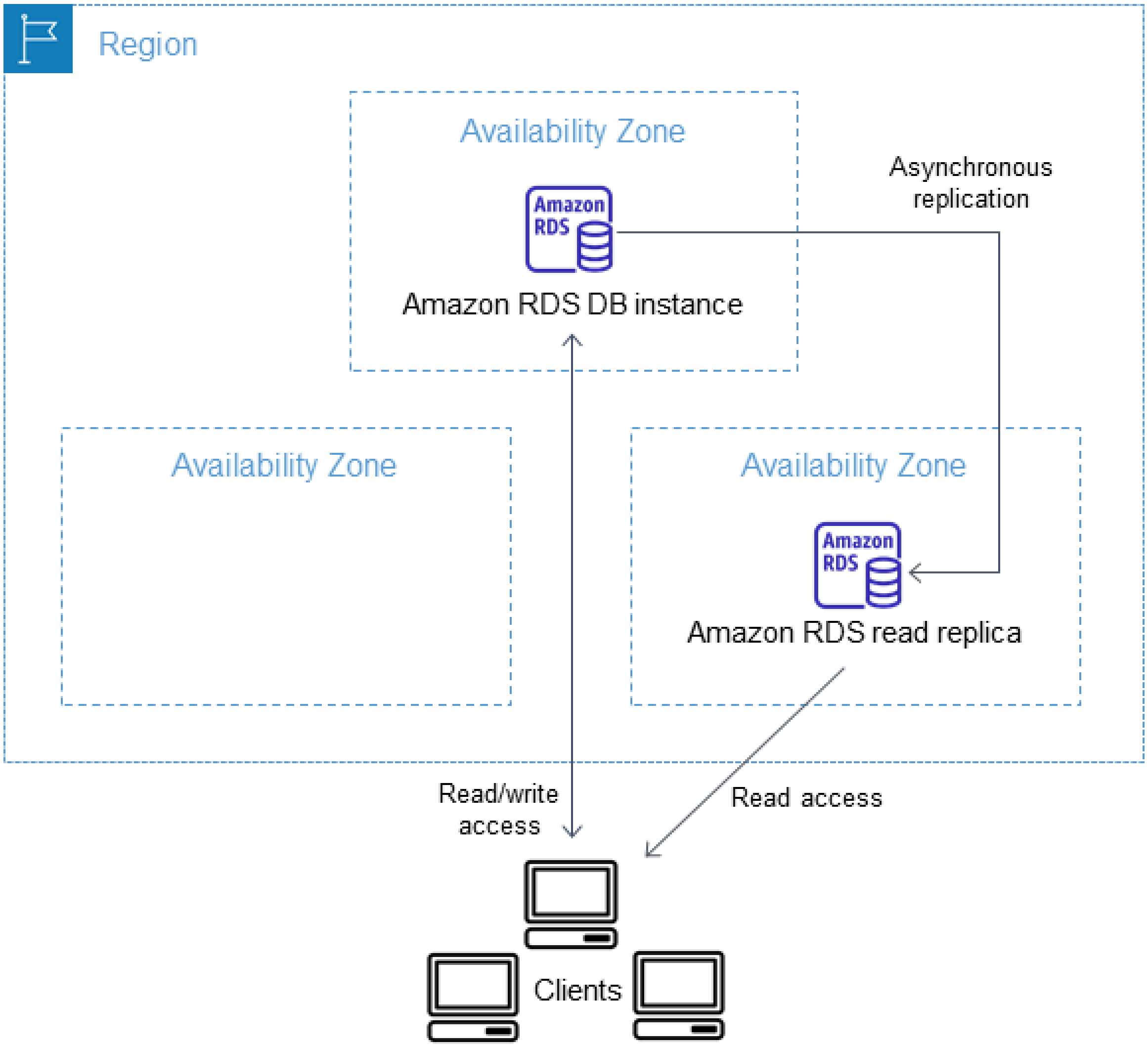
使用数据库实例只读副本

只读副本是数据库实例的只读副本。您可以将从应用程序发出的查询路由到只读副本，以减轻主数据库实例上的负载。通过这种方法，可以实现弹性横向扩展并超越单个数据库实例的容量限制，以处理读取密集型数据库工作负载。

要从源数据库实例创建只读副本，Amazon RDS 使用数据库引擎的内置复制特征。有关在特定引擎上使用只读副本的信息，请参阅以下部分：

- 使用适用于 Amazon RDS for Db2 的副本
- 使用 MariaDB 只读副本
- 在 Amazon RDS 中使用 Microsoft SQL Server 的只读副本
- 使用 MySQL 只读副本
- 使用 Amazon RDS for Oracle 的只读副本
- 使用 Amazon RDS for PostgreSQL 只读副本

在从源数据库实例创建只读副本后，源将成为主数据库实例。当您更新主数据库实例时，Amazon RDS 将它们异步复制到只读副本。下图显示了源数据库实例正在复制到不同可用区（AZ）中的只读副本。客户端拥有对主数据库实例的读/写访问权限以及对副本的只读访问权限。



只读副本按标准数据库实例计费，费率与用于副本的数据库实例类相同。在位于同一个 Amazon Web Services 区域中的源数据库实例与只读副本之间复制数据所产生的数据传输不收费。有关更多信息，请参阅跨区域复制成本和Amazon RDS 的数据库实例计费。

主题

- Amazon RDS 只读副本概述
- 数据库引擎的只读副本之间的差异
- 创建只读副本
- 将只读副本提升为独立的数据库实例
- 监控只读复制
- 在不同的 Amazon Web Services 区域中创建只读副本

Amazon RDS 只读副本概述

以下各节讨论数据库实例只读副本。有关多可用区数据库集群 只读副本的信息，请参阅the section called “使用多可用区数据库集群只读副本”。

主题

只读副本的使用案例

为给定的源数据库实例部署一个或多个只读副本在各种情况下都可能有意义，包括：

- 扩展用于执行大量读取数据库工作负载的单个数据库实例的计算资源或 I/O 容量。可以将过多的读取流量引导到一个或多个只读副本。
- 当源数据库实例不可用时，为读取流量提供服务。在某些情况下，您的源数据库实例可能无法处理 I/O 请求（例如，由于备份或计划维护而暂停 I/O)。在这些情况下，您可以将读取流量引导到只读副本。对于这种使用案例，请注意，由于源数据库实例不可用，因而只读副本中的数据可能处于“静态”。
- 对于业务报告或数据仓库方案，您可能希望针对只读副本（而生产数据库实例）运行业务报告查询。
- 实施灾难恢复。如果主数据库实例出现故障，则将只读副本提升为独立的实例作为灾难恢复解决方案。

只读副本的工作原理

创建只读副本时，请指定现有的数据库实例作为源。紧接着，Amazon RDS 拍摄源实例的快照，并根据该快照创建一个只读实例。然后，每当主数据库实例发生更改时，Amazon RDS 就使用数据库引擎的异步复制方法来更新只读副本。

只读副本作为仅支持只读连接的数据库实例运行。例外情况是 RDS for Db2 和 RDS for Oracle DB 数据库引擎，它们分别支持处于备用模式和装载模式的副本数据库。备用副本和装载副本不接受用户连

接，因此无法为只读工作负载提供服务。备用副本和装载副本的主要用途是跨区域灾难恢复。有关更多信息，请参阅使用适用于 Amazon RDS for Db2 的副本和使用 Amazon RDS for Oracle 的只读副本。

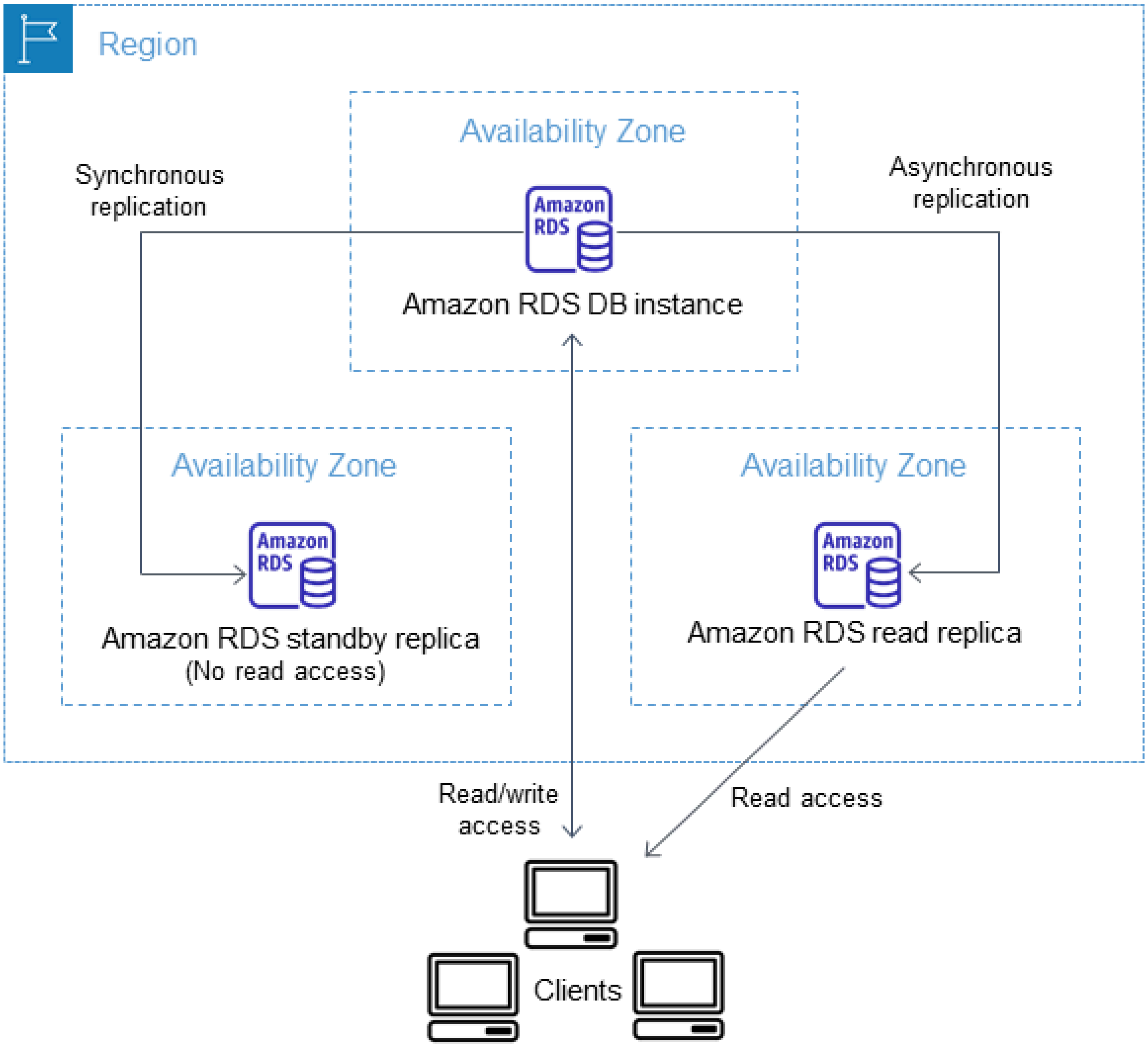
应用程序连接到只读副本就像连接到任何数据库实例一样。Amazon RDS 复制源数据库实例中的所有数据库。

您必须手动创建只读副本。RDS 不支持自动扩缩只读副本，即在读取需求发生变化时自动添加或移除只读副本。

多可用区部署中的只读副本

您可以为数据库实例配置只读副本，此外还配置了备用副本以实现多可用区部署中的高可用性。使用备用副本进行复制是同步的。与只读副本不同，备用副本无法处理读取流量。

在以下场景中，客户端拥有对一个可用区中的主数据库实例的读取/写入权限。主实例将更新异步复制到第二个可用区中的只读副本，还将更新同步复制到第三个可用区中的备用副本。客户端仅对只读副本具有读取权限。



有关高可用性和备用副本的更多信息，请参阅[配置和管理 Amazon RDS 的多可用区部署]。

跨区域只读副本

在某些情况下，只读副本与其主数据库实例位于不同的 Amazon Web Services 区域。在这些情况下，Amazon RDS 在主数据库实例与只读副本之间设置安全通道。Amazon RDS 建立启用安全通道所需的任何Amazon安全配置，例如添加安全组条目。有关跨区域只读副本的更多信息，请参阅[在不同的 Amazon Web Services 区域中创建只读副本]。

本章中的信息适用于在源数据库实例所在的相同 Amazon Web Services 区域中或在单独的 Amazon Web Services 区域中创建 Amazon RDS 只读副本。以下信息不适用于对 Amazon EC2 实例上运行的实例或本地实例设置复制操作。

只读副本存储类型

默认情况下，将创建与源数据库实例的存储类型相同的只读副本。但是，您可基于下表中列出的选项创建与源数据库实例的存储类型不同的只读副本。

源数据库实例存储类型	源数据库实例存储分配	只读副本存储类型选项
预置 IOPS	100 GiB–64 TiB	预调配 IOPS、通用型、磁性
通用	100 GiB–64 TiB	预调配 IOPS、通用型、磁性
通用	<100 GiB	通用型，磁性
磁介质	100 GiB–6 TiB	预调配 IOPS、通用型、磁性
磁介质	<100 GiB	通用型，磁性

> **Note**
>
> 增加只读副本的分配存储空间时，它必须至少增加 10%。如果您尝试将该值增加不到 10%，则会得到错误。

从副本创建副本的限制

Amazon RDS 不支持循环复制。您不能将数据库实例配置为充当现有数据库实例的复制源。您只可以基于现有数据库实例创建新的只读副本。例如，如果 **MySourceDBInstance** 复制到 **ReadReplica1**，您不能将 **ReadReplica1** 配置为复制回 **MySourceDBInstance**。

对于 RDS for MariaDB 和 RDS for MySQL 以及特定版本的 RDS for PostgreSQL，您可以基于现有只读副本创建只读副本。例如，您可以从现有副本 **ReadReplica1** 创建新的只读副本 **ReadReplica2**。对于 RDS for Db2、RDS for Oracle 和 RDS for SQL Server，您不能基于现有只读副本来创建只读副本。

删除副本时的注意事项

RDS 不支持自动扩缩只读副本。因此，RDS 不会在需求时增加副本数量，也不会在需求减少时减少副本数量。如果您不再需要只读副本，则使用用于删除数据库实例的相同机制手动删除它们。如果您在同

一 Amazon Web Services 区域中删除源数据库实例而未删除其只读副本，则每个副本都会提升为独立的数据库实例。

有关删除数据库实例的信息，请参阅删除数据库实例。有关只读副本提升的信息，请参阅将只读副本提升为独立的数据库实例。有关删除跨区域只读副本的源数据库实例的信息，请参阅跨区域复制注意事项。

数据库引擎的只读副本之间的差异

由于 Amazon RDS 数据库引擎实施复制的方式不同，因此您应了解几个显著差异。

Db2

RDS for Db2 的副本具有以下功能和行为：

- 复制方法：物理复制。
- 事务日志清除：当满足以下条件时，RDS for Db2 会清除主数据库实例中的日志：
  - 这些日志至少保留了两个小时。
  - 归档日志保留时间设置已过期。
  - RDS for Db2 已成功将日志复制到所有副本数据库实例。

  这同时适用于相同 Amazon Web Services 区域数据库实例和跨区域数据库实例。有关如何设置存档日志保留时间的信息，请参阅rdsadmin.set_archive_log_retention。
- 可写副本：Db2 副本是物理副本，而 Db2 不支持写入副本。您可以提升副本以使其变为可写。提升的副本拥有提出提升请求时的已复制数据。
- 备份：RDS for Db2 副本支持自动备份和手动快照。
- 并行复制：归档日志数据始终会从主数据库并行传输到其所有副本。
- 备用状态：备用副本的主要用途是跨区域灾难恢复。有关信息，请参阅使用适用于 Amazon RDS for Db2 的副本。

MariaDB 和 MySQL

RDS for MariaDB 和 RDS for MySQL 的只读副本具有以下功能和行为：

- 复制方法：逻辑复制。
- 事务日志清除：RDS for MariaDB 和 RDS for MySQL 保留尚未应用的任何二进制日志。
- 可写副本：可将 MariaDB 或 MySQL 只读副本启用为可写。

- 备份：RDS for MariaDB 或 RDS for MySQL 只读副本支持自动备份和手动快照。
- 并行复制：所有受支持的 MariaDB 和 MySQL 版本均支持并行复制线程。
- 装载状态：不支持。

Oracle

RDS for Oracle 的只读副本具有以下功能和行为：

- 复制方法：物理复制。
- 事务日志清除：如果主数据库实例没有跨区域只读副本，则 Amazon RDS for Oracle 会在源数据库实例上至少保留两个小时的事务日志。两小时之后或在传递存档日志保留小时数设置之后从源数据库实例中清除日志，以时间较长者为准。传递存档日志保留小时数设置之后，仅当日志已成功应用到数据库时，才会从只读副本清除日志。

  在某些情况下，主数据库实例可能具有一个或多个跨区域只读副本。如果是这样，则 Amazon RDS for Oracle 将保留源数据库实例上的事务日志，直到将它们传输并应用到所有跨区域只读副本为止。

  有关如何设置存档日志保留时间的信息，请参阅保留存档重做日志。
- 可写副本：Oracle 只读副本是物理副本，Oracle 不支持写入只读副本。您可以提升只读副本以将它设置为可写。提升的只读副本拥有提出提升请求时的已复制数据。
- 备份：RDS for Oracle 只读副本支持自动备份和手动快照。
- 并行复制：重做日志数据始终会从主数据库并行传输到其所有只读副本。
- 装载状态：装载副本的主要用途是跨区域灾难恢复。装载副本不需要具有 Active Data Guard 许可证。有关更多信息，请参阅使用 Amazon RDS for Oracle 的只读副本。

PostgreSQL

RDS for PostgreSQL 的只读副本具有以下功能和行为：

- 复制方法：物理复制。
- 事务日志清除：PostgreSQL 具有参数 `wal_keep_segments`，该参数指示保留多少个提前写入日志（WAL）文件以便为只读副本提供数据。该参数值指定要保留的日志的数量。
- 可写副本：PostgreSQL 只读副本是一个物理副本，PostgreSQL 不支持使只读副本变为可写。
- 备份：RDS for PostgreSQL 只读副本支持手动快照。仅 RDS for PostgreSQL 14.1 及更高版本支持只读副本的自动备份。您无法为早于 14.1 的 RDS for PostgreSQL 版本的 PostgreSQL 只读副本启

用自动备份。对于 RDS for PostgreSQL 13 和更早版本，如果需要只读副本的备份，请从只读副本创建快照。

- 并行复制：PostgreSQL 是一个处理复制的过程。
- 装载状态：不支持。

SQL Server

RDS for SQL Server 的只读副本具有以下功能和行为：

- 复制方法：物理复制。
- 事务日志清除：辅助副本不再需要主副本上事务日志文件的虚拟日志文件（VLF）后，可以截断该文件。

  只有在副本中稳定了日志记录时，才能将 VLF 标记为非活动。无论磁盘子系统在主副本中的速度如何，事务日志都将保留 VLF，直到最慢的副本已进行稳定。
- 可写副本：SQL Server 只读副本是物理副本，也不支持写入。您可以提升只读副本以将它设置为可写。提升的只读副本拥有提出提升请求时的已复制数据。
- 备份：RDS for SQL Server 只读副本不支持自动备份和手动快照。
- 并行复制：重做日志数据始终会从主数据库并行传输到其所有只读副本。
- 装载状态：不支持。

创建只读副本

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 从现有数据库实例创建只读副本。可通过指定 `SourceDBInstanceIdentifier`（要从其进行复制的源数据库实例的数据库实例标识符）创建只读副本。

当您创建只读副本时，Amazon RDS 会制作源数据库实例的数据库快照，然后开始复制。数据库快照操作开始时，源数据库实例会经历非常短暂的 I/O 暂停。这种 I/O 暂停通常持续大约一秒时间。如果源数据库实例为多可用区部署，则可避免 I/O 暂停，因为在这种情况下，会从辅助数据库实例制作数据库快照。

长时间运行的活动事务会减慢创建只读副本的过程。我们建议您等待长时间运行的事务完成，然后再创建只读副本。如果从同一源数据库实例同时创建多个并行只读副本，则 Amazon RDS 仅在开始第一个创建操作时创建一张快照。

创建只读副本时，有几件事情需要考虑。首先，您必须先通过将备份保留期设置为一个非零值来在源数据库实例上启用自动备份。此要求也适用于作为一个只读副本的源数据库实例的另一个只读副本。要对 RDS for MariaDB 只读副本启用自动备份，请先创建只读副本，然后修改只读副本以启用自动备份。

> **Note**
>
> 在 Amazon Web Services 区域内，我们强烈建议您在与源数据库实例相同的基于 Amazon VPC 的虚拟私有云（VPC）中创建所有只读副本。如果您在与源数据库实例不同的 VPC 中创建只读副本，则副本和 RDS 系统之间的无类域间路由 (CIDR) 范围可能会重叠。CIDR 重叠会使副本不稳定，从而对连接到副本的应用程序产生负面影响。如果您在创建只读副本时收到错误，请选择其他目标数据库子网组。有关更多信息，请参阅在 VPC 中使用数据库实例。
>
> 无法直接使用控制台或 Amazon CLI 在其他 Amazon Web Services 账户中创建只读副本。

控制台

从源数据库实例创建只读副本

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。
3. 选择要作为只读副本源的数据库实例。
4. 对于 Actions (操作)，选择 Create read replica (创建只读副本)。
5. 对于数据库实例标识符，输入该只读副本的名称。
6. 选择您的实例配置。建议您为只读副本使用与源数据库实例相同或更大的数据库实例类和存储类型。
7. 对于 Amazon Web Services 区域，指定只读副本的目标区域。
8. 对于存储，指定分配的存储大小以及是否要使用存储自动扩缩。

   如果您的源数据库实例未使用最新的存储配置，则可以使用升级存储文件系统配置选项。您可以启用此设置将只读副本的存储文件系统升级到首选配置。有关更多信息，请参阅the section called “升级存储文件系统”。
9. 对于可用性，选择是否在另一个可用区中创建副本的备用来支持副本的故障转移。

> **Note**
>
> 创建您的只读副本作为多可用区数据库实例与源数据库是否为多可用区数据库实例无关。

10. 指定其他数据库实例设置。有关每项可用设置的信息，请参阅数据库实例的设置。
11. 要创建加密的只读副本，请展开其他配置并指定以下设置：

    a. 选择启用加密。
    b. 对于 Amazon KMS key，选择 KMS 密钥的 Amazon KMS key 标识符。

    > **Note**
    >
    > 源数据库实例必须加密。要了解关于加密源数据库实例的更多信息，请参阅 加密 Amazon RDS 资源。

12. 选择 Create read replica (创建只读副本)。

创建只读副本后，您可以在 RDS 控制台的数据库页面上看到它。它在角色列中显示副本。

Amazon CLI

要从源数据库实例创建只读副本，请使用 Amazon CLI 命令 create-db-instance-read-replica。此示例还设置了分配存储大小、启用了存储自动扩展，并将文件系统升级到首选配置。

您可以指定其他设置。有关每项设置的信息，请参阅 数据库实例的设置。

> **Note**
>
> 要创建 RDS for Db2 备用副本，请在以下 Amazon CLI 命令中包括 `--replica-mode mounted`。

Example

对于 Linux、macOS 或 Unix：

```
aws rds create-db-instance-read-replica \
```

```
    --db-instance-identifier myreadreplica \
    --source-db-instance-identifier mydbinstance \
    --allocated-storage 100 \
    --max-allocated-storage 1000 \
    --upgrade-storage-config
```

对于 Windows：

```
aws rds create-db-instance-read-replica ^
    --db-instance-identifier myreadreplica ^
    --source-db-instance-identifier mydbinstance ^
    --allocated-storage 100 ^
    --max-allocated-storage 1000 ^
    --upgrade-storage-config
```

RDS API

要从源 MySQL、MariaDB、Oracle、PostgreSQL 或 SQL Server 数据库实例创建只读副本，请调用具有以下必要参数的 Amazon RDS API CreateDBInstanceReadReplica 操作：

- `DBInstanceIdentifier`
- `SourceDBInstanceIdentifier`

> **Note**
>
> 要创建 RDS for Db2 备用副本，请将可选 `ReplicaMode` 操作设置为 `mounted`。

将只读副本提升为独立的数据库实例

您可以将只读副本提升为独立的数据库实例。如果源数据库实例有多个只读副本，则将其中一个只读副本提升为数据库实例对其他副本无影响。

提升只读副本时，RDS 会在使其可用之前重启数据库实例。提升过程可能需要几分钟或更长时间才能完成，具体时间取决于只读副本的大小。

提升只读副本的使用案例

出于以下几个原因，您可能需要将只读副本提升为独立的数据库实例：

- 实施故障恢复 – 如果主数据库实例失败，您可以使用只读副本提升作为数据恢复方案。此方法可补充同步复制、自动故障检测和故障转移。

  如果您了解异步复制的影响和限制，并且仍然要使用只读副本提升功能实现数据恢复，则可以这样做。为此，请先创建只读副本，然后监控主数据库实例的故障。如果出现故障，请执行以下操作：

1. 提升只读副本。
2. 将数据库流量指引到提升的数据库实例。
3. 将提升的数据库实例作为源，创建替代只读副本。

- 升级存储配置 - 如果您的源数据库实例未使用首选存储配置，则可以创建该实例的只读副本，并升级存储文件系统配置。此选项可将只读副本的文件系统迁移到首选配置。您可以将只读副本升级为独立的数据库实例。

  您可以使用此选项来克服较旧 32 位文件系统的存储和文件大小的扩展限制。有关更多信息，请参阅 the section called "升级存储文件系统"。

  只有当您的源数据库实例未使用最新的存储配置，或者您要在同一请求中修改数据库实例类时，此选项才可用。
- 分片 – 分片体现了"无共享"架构，主要涉及将大型数据库分成几个较小的数据库。拆分数据库的一种常用方法是将未加入到同一查询中的表拆分到不同主机。另一种方法是跨多台主机复制表，然后通过哈希算法确定哪一台主机接收给定的更新。您可以创建与各分片（较小的数据库）相对应的只读副本，决定将它们转换成独立分片后即可对其进行提升。然后可以根据您的要求，为每个分区创建表的密钥空间 (如果要拆分行) 或分配。
- 执行 DDL 操作（仅限 MySQL 和 MariaDB）– 创建或重新生成索引等 DDL 操作需要耗费时间，并且会显著影响数据库实例性能。在 MySQL 或 MariaDB 只读副本与其主数据库实例同步后，可以对该只读副本执行这些操作。然后就可以提升此只读副本并指导应用程序使用提升的实例。

> **Note**
>
> 如果您的只读副本是 RDS for Oracle 数据库实例，则可以执行切换而不是提升操作。在切换中，源数据库实例成为新的副本，副本成为新的源数据库实例。有关更多信息，请参阅 执行 Oracle Data Guard 切换。

提升后的只读副本的特征

在提升只读副本后，它不再作为只读副本，而是成为独立的数据库实例。该新的独立数据库实例具有以下特性：

- 独立的数据库实例会保留提升前只读副本的选项组和参数组。
- 您可以从独立的数据库实例创建只读副本，并执行时间点还原操作。
- 由于数据库实例不再是只读副本，因此不能再使用它作为复制目标。

提升只读副本的先决条件

在提升只读副本之前，请执行以下操作：

- 查看您的备份策略：
  - 建议启用备份并至少完成一次备份。备份持续时间随在上次备份后对数据库进行的更改数而发生变化。
  - 如果您为只读副本启用了备份，请配置自动备份时段，以便每日备份不会影响只读副本提升。
  - 确保您的只读副本不处于 `backing-up` 状态。当只读副本处于这种状态时，您无法对其进行提升。
- 停止向主数据库实例写入任何事务，然后等待 RDS 将所有更新应用于只读副本。

  在主数据库实例上完成数据库更新后，只读副本进行数据库更新。复制滞后可能会有很大差异。使用 [`Replica Lag`] 指标确定只读副本完成所有更新的时间。
- （仅限 MySQL 和 MariaDB）：如果要在提升之前，对 MySQL 或 MariaDB 只读副本进行更改，必须在只读副本的数据库参数组中将 `read_only` 参数设置为 `0`。然后可在只读副本上执行所有必需的 DDL 操作，如创建索引。在只读副本上执行的操作不会影响主数据库实例的性能。

提升只读副本：基本步骤

以下步骤说明将只读副本提升为数据库实例的一般过程：

1. 使用 Amazon RDS 控制台上的 Promote（提升）选项、Amazon CLI 命令 [`promote-read-replica`] 或 [`PromoteReadReplica`] Amazon RDS API 操作来提升只读副本。

   > **Note**
   >
   > 提升过程需要几分钟才能完成。在提升只读副本时，RDS 会停止复制并重启只读副本。完成重启后，只读副本即可作为新数据库实例使用。

2. (可选) 将新的数据库实例修改为多可用区部署。有关更多信息，请参阅 [修改 Amazon RDS 数据库实例] 和 [配置和管理 Amazon RDS 的多可用区部署]。

控制台

将只读副本提升为独立的数据库实例

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在 Amazon RDS 控制台中，选择数据库。

   随后会显示数据库窗格。每个只读副本在角色列中显示副本。
3. 选择要提升的只读副本。
4. 对于操作，请选择提升。
5. 在提升只读副本页面上，输入新提升的数据库实例的备份保留期和备份时段。
6. 根据需要设置完毕后，选择 Continue。
7. 在确认页面上，选择 Promote Read Replica (提升只读副本)。

Amazon CLI

要将只读副本提升为独立的数据库实例，请使用 Amazon CLI `promote-read-replica` 命令。

Example

对于 Linux、macOS 或 Unix：

```
aws rds promote-read-replica \
    --db-instance-identifier myreadreplica
```

对于 Windows：

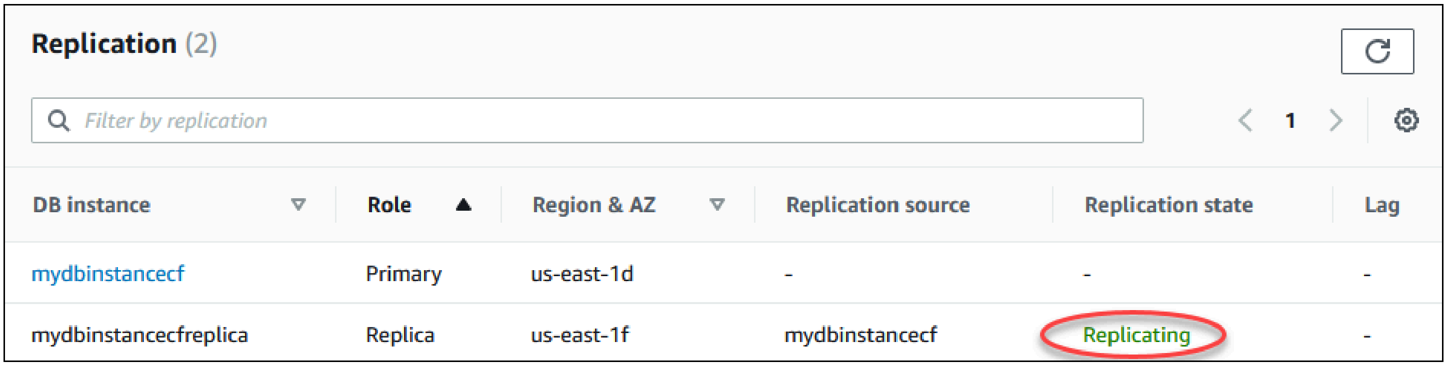
```
aws rds promote-read-replica ^
    --db-instance-identifier myreadreplica
```

RDS API

要将只读副本提升为独立的数据库实例，请调用具有必要参数 `PromoteReadReplica` 的 Amazon RDS API `DBInstanceIdentifier` 操作。

监控只读复制

您可以通过几个方法监控只读副本的状态。Amazon RDS 控制台在只读副本详细信息的 Connectivity & security（连接和安全）选项卡的 Replication（复制）部分中显示只读副本的状态。要查看只读副本的详细信息，请在 Amazon RDS 控制台的数据库实例列表中选择该只读副本的名称。



您还可以使用 Amazon CLI`describe-db-instances` 命令或 Amazon RDS API `DescribeDBInstances` 操作查看只读副本的状态。

只读副本的状态可以是以下某项：

- 正在复制 – 正在成功地复制只读副本。
- 复制已降级（仅 SQL Server 和 PostgreSQL）– 副本从主实例接收数据，但一个或多个数据库可能无法获取更新。例如，当副本正在设置新创建的数据库时，可能会发生这种情况。当在蓝绿部署的蓝色环境中进行不支持的 DDL 或大型对象更改时，也可能发生这种情况。

  除非在降级状态期间发生错误，否则状态不会从 `replication degraded` 转换为 `error`。
- 错误 – 复制出错。检查 Amazon RDS 控制台中的复制错误字段或事件日志以确定具体错误。有关复制错误故障诊断的详细信息，请参阅 排查 MySQL 只读副本问题。
- 已终止（仅限 MariaDB、MySQL 或 PostgreSQL）– 复制已终止。如果复制连续终止超过 30 天，不论是手动还是由于复制错误，都将会出现这种情况。在这种情况下，Amazon RDS 会终止主数据库实例和所有只读副本之间的复制。Amazon RDS 这样做是为了防止源数据库实例上的存储需求增长以及长失效转移时间。

  复制中断可能影响存储，因为日志的大小和数量可能因向日志写入大量错误消息而增大。复制中断还可能影响故障恢复，因为 Amazon RDS 在恢复期间需要一段时间来维护和处理大量日志。
- terminated（已终止）（仅限 Oracle）– 复制已终止。如果复制因只读副本上没有剩下足够的存储空间而停止超过 8 小时，就会发生这种情况。在这种情况下，Amazon RDS 会终止主数据库实例与受影响只读副本之间的复制。此状态为最终状态，必须重新创建只读副本。
- 已停止（仅限 MariaDB 或 MySQL）– 复制已因客户发起请求而停止。

- 复制停止点设置（仅限 MySQL）– 使用mysql.rds_start_replication_until存储过程设置客户启动的停止点，并且正在进行复制。
- 达到复制停止点（仅限 MySQL）– 使用mysql.rds_start_replication_until存储过程设置客户启动的停止点，并且复制已停止（因为已达到停止点）。

您可以查看到数据库实例的复制位置，如果是，请检查其复制状态。在 RDS 控制台的数据库页面上，它在角色列中显示主要。选择其数据库实例名称。在详细信息页面的连接性和安全性选项卡上，其复制状态位于复制下。

监控复制滞后

您可以通过查看 Amazon RDS `ReplicaLag` 指标，在 Amazon CloudWatch 中监控复制滞后。

对于 Db2，`ReplicaLag` 指标是已落后的数据库的最大滞后秒数。例如，如果两个数据库分别滞后 5 秒和 10 秒，则 `ReplicaLag` 为 10 秒。没有可用高可用性灾难恢复（HADR）状态的数据库不包括在计算中。

对于 MariaDB 和 MySQL，`ReplicaLag` 指标报告 `Seconds_Behind_Master` 命令的 `SHOW REPLICA STATUS` 字段的值。MySQL 和 MariaDB 的复制滞后的常见原因如下所示：

- 网络中断。
- 向只读副本上带索引的表写入。如果只读副本上的 `read_only` 参数未设置为 0，它可以中断复制。
- 使用 MyISAM 等非事务性存储引擎。只有 MySQL 上 InnoDB 存储引擎和 MariaDB 上的 XtraDB 存储引擎支持复制。

> **Note**
>
> 以前的 MariaDB 版本使用的是 `SHOW SLAVE STATUS`，而不是 `SHOW REPLICA STATUS`。如果您使用的 MariaDB 版本低于 10.5，请使用 `SHOW SLAVE STATUS`。

当 `ReplicaLag` 指标达到 0 时，即表示副本已赶上主数据库实例进度。如果 `ReplicaLag` 指标返回 `-1`，则当前未激活复制。`ReplicaLag = -1` 等效于 `Seconds_Behind_Master = NULL`。

对于 Oracle，`ReplicaLag` 指标是以下项的总和：`Apply Lag` 值加上当前时间和应用滞后的 `DATUM_TIME` 值之间的差。`DATUM_TIME` 值是只读副本从其源数据库实例收到数据的最后时间。有关更多信息，请参阅 Oracle 文档中的V$DATAGUARD_STATS。

对于 SQL Server，ReplicaLag 指标是已落后数据库的最大滞后秒数。例如，如果您有两个数据库分别滞后 5 秒和 10 秒，则 ReplicaLag 为 10 秒。ReplicaLag 指标返回以下查询的值。

```
SELECT MAX(secondary_lag_seconds) max_lag FROM sys.dm_hadr_database_replica_states;
```

有关更多信息，请参阅 Microsoft 文档中的 secondary_lag_seconds。

如果 RDS 无法确定滞后，例如在副本安装期间或只读副本处于 ReplicaLag 状态时，则 -1 返回 error。

> **Note**
> 只有当新数据库在只读副本上可供访问之后，才会将其包含在延迟计算中。

对于 PostgreSQL， ReplicaLag 指标返回以下查询的值。

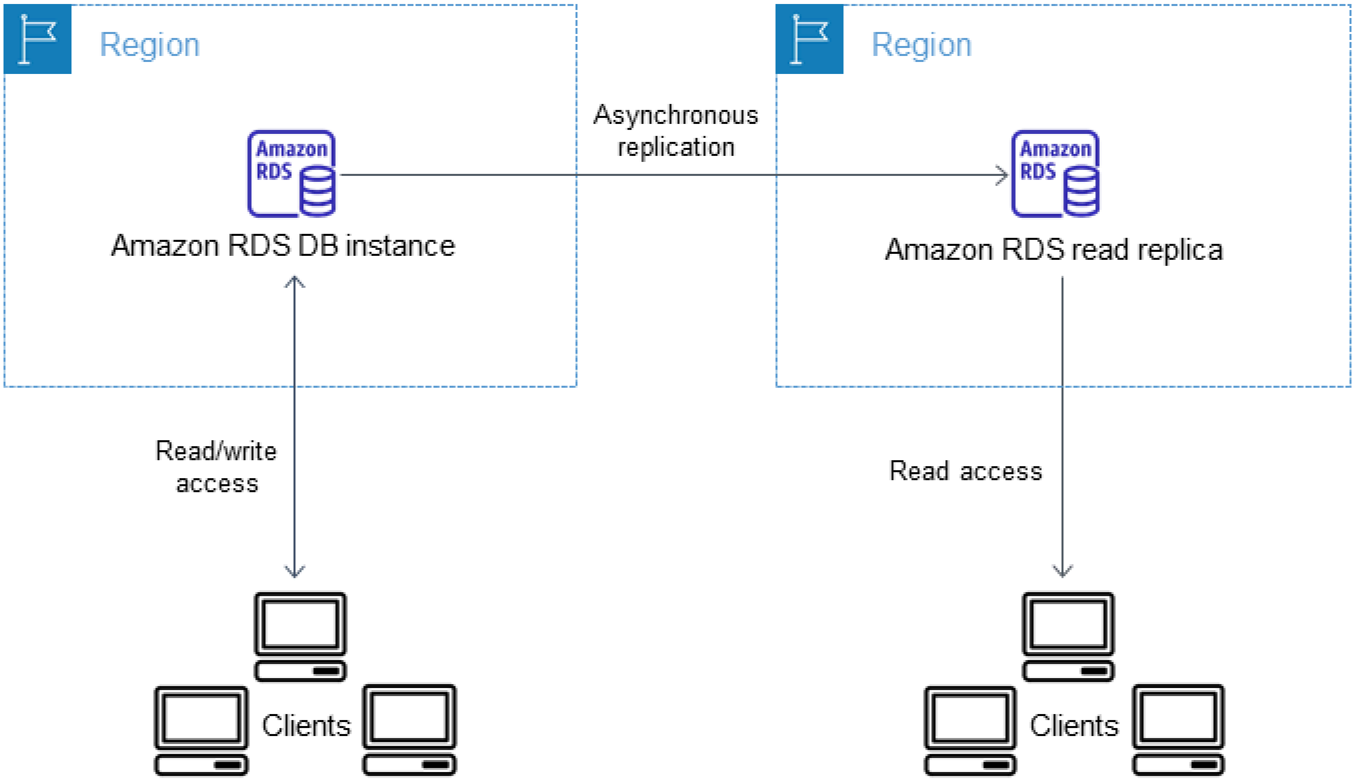
```
SELECT extract(epoch from now() - pg_last_xact_replay_timestamp()) AS reader_lag
```

PostgreSQL 9.5.2 版及更高版本使用物理复制槽来管理源实例上的提前写入日志 (WAL) 保留。对于每个跨区域只读副本实例，Amazon RDS 创建一个物理复制槽并将它与实例关联。两个 Amazon CloudWatch 指标、Oldest Replication Slot Lag 和 Transaction Logs Disk Usage 表示最滞后的副本（依据接收到的 WAL 数据）的滞后时间和用于 WAL 数据的存储空间。当跨区域只读副本长时间滞后时，Transaction Logs Disk Usage 值会显著增大。

有关使用 CloudWatch 监控数据库实例的更多信息，请参阅使用 Amazon CloudWatch 监控 Amazon RDS 指标。

在不同的 Amazon Web Services 区域中创建只读副本

借助 Amazon RDS，您可以在与源数据库实例不同的 Amazon Web Services 区域中创建只读副本。



您可在其他 Amazon Web Services 区域创建只读副本，用于执行以下操作：

- 完善灾难恢复功能。
- 将读取操作扩展到与用户更接近的 Amazon Web Services 区域。
- 使从一个 Amazon Web Services 区域中的数据中心迁移到另一个 Amazon Web Services 区域中的数据中心的操作变得更轻松。

在与源实例不同的 Amazon Web Services 区域中创建只读副本与在同一 Amazon Web Services 区域中创建副本非常相似。您可以使用 Amazon Web Services Management Console、运行 `create-db-instance-read-replica` 命令或调用 `CreateDBInstanceReadReplica` API 操作。

> **Note**
> 要在与源数据库实例不同的 Amazon Web Services 区域中创建一个加密的只读副本，源数据库实例必须加密。

主题

- 区域和版本可用性

- 创建跨区域只读副本
- Amazon RDS 如何执行跨区域复制
- 跨区域复制注意事项
- 跨区域复制成本

区域和版本可用性

功能可用性和支持因每个数据库引擎的特定版本以及 Amazon Web Services 区域而异。有关跨区域复制的版本和区域可用性的更多信息，请参阅支持 Amazon RDS 中跨区域只读副本的区域和数据库引擎。

创建跨区域只读副本

以下过程演示如何根据不同 Amazon Web Services 区域中的源 MariaDB、Microsoft SQL Server、MySQL、Oracle 或 PostgreSQL 数据库实例创建只读副本。

控制台

您可以使用 Amazon Web Services Management Console 跨不同 Amazon Web Services 区域创建只读副本。

使用控制台跨不同 Amazon Web Services 区域创建只读副本

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。
3. 选择要用作只读副本源的 MariaDB、Microsoft SQL Server、MySQL、Oracle 或 PostgreSQL 数据库实例。
4. 对于 Actions (操作)，选择 Create read replica (创建只读副本)。
5. 对于数据库实例标识符，输入该只读副本的名称。
6. 选择目标区域。
7. 选择要使用的实例规格。我们建议您为只读副本使用相同或更大的数据库实例类和存储类型。
8. 要在其他 Amazon Web Services 区域中创建加密只读副本，请执行以下操作：

    a. 选择启用加密。
    b. 对于 Amazon KMS key，请选择目标 Amazon Web Services 区域中的 KMS 密钥的 Amazon KMS key 标识符。

> **Note**
>
> 要创建加密的只读副本，必须对源数据库实例进行加密。要了解关于加密源数据库实例的更多信息，请参阅 加密 Amazon RDS 资源。

9. 选择其他选项，例如存储自动扩展。
10. 选择 Create read replica (创建只读副本)。

Amazon CLI

要从不同 Amazon Web Services 区域中的源 MariaDB、Microsoft SQL Server、MySQL、Oracle 或 PostgreSQL 数据库实例创建只读副本，可以使用 `create-db-instance-read-replica` 命令。在这种情况下，您从要在其中创建只读副本的 Amazon Web Services 区域（目标区域）使用 `create-db-instance-read-replica`，并指定源数据库实例的 Amazon 资源名称（ARN）。ARN 唯一地标识 Amazon Web Services 中创建的资源。

例如，如果您的源数据库实例位于US East (N. Virginia)区域，则 ARN 看起来类似于下例：

```
arn:aws:rds:us-east-1:123456789012:db:mydbinstance
```

有关 ARN 的信息，请参阅 Amazon RDS 中的 Amazon 资源名称（ARN）。

要在与源数据库实例不同的 Amazon Web Services 区域中创建一个只读副本，您可以在目标 Amazon Web Services 区域使用 Amazon CLI `create-db-instance-read-replica` 命令。在另一 Amazon Web Services 区域创建只读副本需要以下参数：

- `--region` – 创建只读副本的目标 Amazon Web Services 区域。
- `--source-db-instance-identifier` – 源数据库实例的数据库实例标识符。对于源 Amazon Web Services 区域，此标识符必须采用 ARN 格式。
- `--db-instance-identifier` – 目标 Amazon Web Services 区域中只读副本的标识符。

Example 跨区域只读副本

以下代码从 US East (N. Virginia) 区域中的源数据库实例，在 美国西部（俄勒冈州） 区域中创建只读副本。

对于 Linux、macOS 或 Unix：

```
aws rds create-db-instance-read-replica \
    --db-instance-identifier myreadreplica \
    --region us-west-2 \
    --source-db-instance-identifier arn:aws:rds:us-east-1:123456789012:db:mydbinstance
```

对于 Windows：

```
aws rds create-db-instance-read-replica ^
    --db-instance-identifier myreadreplica ^
    --region us-west-2 ^
    --source-db-instance-identifier arn:aws:rds:us-east-1:123456789012:db:mydbinstance
```

在另一 Amazon Web Services 区域创建加密的只读副本还需要以下参数：

- `--kms-key-id` – 要用于对目标 Amazon Web Services 区域中只读副本进行加密的 KMS 密钥的 Amazon KMS key 标识符。

Example 加密的跨区域只读副本

以下代码从 US East (N. Virginia) 区域中的源数据库实例，在 美国西部（俄勒冈州） 区域中创建加密的只读副本。

对于 Linux、macOS 或 Unix：

```
aws rds create-db-instance-read-replica \
    --db-instance-identifier myreadreplica \
    --region us-west-2 \
    --source-db-instance-identifier arn:aws:rds:us-east-1:123456789012:db:mydbinstance
 \
    --kms-key-id my-us-west-2-key
```

对于 Windows：

```
aws rds create-db-instance-read-replica ^
    --db-instance-identifier myreadreplica ^
    --region us-west-2 ^
    --source-db-instance-identifier arn:aws:rds:us-east-1:123456789012:db:mydbinstance
 ^
    --kms-key-id my-us-west-2-key
```

在中国（北京）和中国（宁夏）Amazon Web Services 区域之间创建加密只读副本时，需要 `--source-region` 选项。对于 `--source-region`，指定源数据库实例的 Amazon Web Services 区域。

如果未指定 `--source-region`，请指定 `--pre-signed-url` 值。预签名 URL 包含签名版本 4 签名的请求，该请求用于在源 Amazon Web Services 区域中调用的 `create-db-instance-read-replica` 命令。要了解有关预签名 URL 的更多信息，请参阅《Amazon RDS API 参考》中的 CreateDBInstanceReadReplica。

RDS API

要从不同 Amazon Web Services 区域中的源 MariaDB、MySQL、Microsoft SQL Server、Oracle 或 PostgreSQL 数据库实例创建只读副本，可以调用 Amazon RDS API 操作 CreateDBInstanceReadReplica。在本示例中，您从要在其中创建只读副本的 Amazon Web Services 区域（目标区域）调用 CreateDBInstanceReadReplica 并指定源数据库实例的 Amazon 资源名称（ARN）。ARN 唯一地标识 Amazon Web Services 中创建的资源。

要在与源数据库实例不同的 Amazon Web Services 区域中创建加密只读副本，您可以在目标 Amazon Web Services 区域使用 Amazon RDS API `CreateDBInstanceReadReplica` 操作。要在另一个 Amazon Web Services 区域中创建加密的只读副本，必须指定 `PreSignedURL` 的值。`PreSignedURL` 应包含对 `CreateDBInstanceReadReplica` 操作的请求，以便调用创建只读副本的源 Amazon Web Services 区域。要了解有关 `PreSignedUrl` 的更多信息，请参阅`CreateDBInstanceReadReplica`。

例如，如果您的源数据库实例位于US East (N. Virginia)区域，则 ARN 看起来类似以下内容。

```
arn:aws:rds:us-east-1:123456789012:db:mydbinstance
```

有关 ARN 的信息，请参阅Amazon RDS 中的 Amazon 资源名称（ARN）。

Example

```
https://us-west-2.rds.amazonaws.com/
    ?Action=CreateDBInstanceReadReplica
    &KmsKeyId=my-us-east-1-key
    &PreSignedUrl=https%253A%252F%252Frds.us-west-2.amazonaws.com%252F
         %253FAction%253DCreateDBInstanceReadReplica
         %2526DestinationRegion%253Dus-east-1
         %2526KmsKeyId%253Dmy-us-east-1-key
```

```
        %2526SourceDBInstanceIdentifier%253Darn%25253Aaws%25253Ards%25253Aus-
west-2%123456789012%25253Adb%25253Amydbinstance
        %2526SignatureMethod%253DHmacSHA256
        %2526SignatureVersion%253D4%2526SourceDBInstanceIdentifier%253Darn%25253Aaws
%25253Ards%25253Aus-west-2%25253A123456789012%25253Ainstance%25253Amydbinstance
        %2526Version%253D2014-10-31
        %2526X-Amz-Algorithm%253DAWS4-HMAC-SHA256
        %2526X-Amz-Credential%253DAKIADQKE4SARGYLE%252F20161117%252Fus-west-2%252Frds
%252Faws4_request
        %2526X-Amz-Date%253D20161117T215409Z
        %2526X-Amz-Expires%253D3600
        %2526X-Amz-SignedHeaders%253Dcontent-type%253Bhost%253Buser-agent%253Bx-amz-
content-sha256%253Bx-amz-date
        %2526X-Amz-Signature
%253D255a0f17b4e717d3b67fad163c3ec26573b882c03a65523522cf890a67fca613
    &DBInstanceIdentifier=myreadreplica
    &SourceDBInstanceIdentifier=&region-arn;rds:us-east-1:123456789012:db:mydbinstance
    &Version=2012-01-15
    &SignatureVersion=2
    &SignatureMethod=HmacSHA256
    &Timestamp=2012-01-20T22%3A06%3A23.624Z
    &AWSAccessKeyId=<&AWS; Access Key ID>
    &Signature=<Signature>
```

Amazon RDS 如何执行跨区域复制

Amazon RDS 使用以下过程创建跨区域只读副本。根据涉及的 Amazon Web Services 区域和数据库中的数据量，此过程可能需要几个小时才能完成。在创建跨区域只读副本时，可使用此信息来确定过程进度：

1. Amazon RDS 开始将源数据库实例作为复制源配置，并将状态设置为 modifying。
2. Amazon RDS 开始在目标 Amazon Web Services 区域中设置指定的只读副本，并将其状态设置为 creating（正在创建）。
3. Amazon RDS 在源 Amazon Web Services 区域创建源数据库实例的自动数据库快照。数据库快照名称的格式为 `rds:<InstanceID>-<timestamp>`，其中 `<InstanceID>` 是源实例的标识符，`<timestamp>` 是复制开始的日期和时间。例如，`rds:mysourceinstance-2013-11-14-09-24` 是在 `mysourceinstance` 从实例 `2013-11-14-09-24` 创建的。在创建自动数据库快照期间，源数据库实例保持正在修改 状态，只读副本保持正在创建 状态，数据库快照处于正在创建 状态。控制台中数据库快照页面的进度栏可报告数据库快照创建进度。数据库快照创建完成后，数据库快照和源数据库实例的状态都将设置为 available (可用)。

4. Amazon RDS 开始为初始数据传输复制跨区域快照。快照复制在目标 Amazon Web Services 区域作为自动快照列出，状态为 creating（正在创建）。其名称与源数据库快照相同。数据库快照的进度栏可显示复制进度。复制完成后，数据库快照的状态将被设置为 available (可用)。
5. Amazon RDS 随后使用对只读副本的初始数据加载的数据库快照副本。在此期间，只读副本位于目标区域的数据库实例列表中，状态为正在创建。加载完成后，只读副本的状态将设置为可用，数据库快照副本则被删除。
6. 当只读副本为可用状态时，Amazon RDS 启动，并复制自创建只读副本操作启动以来对源实例所做的修改。在此期间，只读副本的复制滞后时间将大于 0。

   有关使用复制滞后时间的信息，请参阅 监控只读复制。

跨区域复制注意事项

在一个 Amazon Web Services 区域内执行复制需要注意的所有事项都适用于跨区域复制。在执行跨 Amazon Web Services 区域复制时，还需要额外注意以下事项：

- 源数据库实例可在多个 Amazon Web Services 区域具有跨区域只读副本。由于源 VPC 对访问控制列表（ACL）条目数有限制，因此 RDS 无法保证 5 个以上的跨区域只读副本数据库实例。
- 您可以在 GovCloud（美国东部）和 GovCloud（美国西部）区域之间进行复制，但无法复制到 GovCloud（美国）或从其中复制。
- 对于以下 RDS 引擎，只有当源 Amazon RDS 数据库实例不是其他 Amazon RDS 数据库实例的只读副本时，才能从该源创建跨区域的 Amazon RDS 只读副本。
  - RDS for SQL Server
  - RDS for Oracle
  - 版本低于 14.1 的 RDS for PostgreSQL

  此限制不适用于运行 RDS for PostgreSQL 版本 14.1 版及更高版本、RDS for MariaDB 和 RDS for MySQL 版本的数据库实例。
- 对于所在 Amazon Web Services 区域与源实例不同的任何只读副本，其滞后时间将更长。产生此滞后时间是因为区域数据中心之间的网络通道更长。
- 对于跨区域只读副本，所有指定 `--db-subnet-group-name` 参数的只读副本创建命令必须从同一 VPC 指定数据库子网组。
- 大多数情况下，只读副本使用指定数据库引擎的默认数据库参数组和数据库选项组。

  对于 Db2 数据库引擎，必须为副本指定自定义参数组。对于使用 BYOL 模型的副本，此自定义参数组必须包括您的 IBM Site ID 和 IBM Customer ID。您可以使用 Amazon Web Services

Management Console、Amazon CLI 或 RDS API 为副本指定此自定义参数组。有关更多信息，请参阅 Db2 自带许可的 IBM ID。

对于 MySQL 和 Oracle 数据库引擎，您可以在 Amazon CLI 命令 create-db-instance-read-replica 的 `--db-parameter-group-name` 选项中为只读副本指定自定义参数组。使用 Amazon Web Services Management Console 时，无法指定自定义参数组。

- 只读副本使用默认安全组。
- 对于 MariaDB、Microsoft SQL Server、MySQL 和 Oracle 数据库实例，删除跨区域只读副本的源数据库实例时，将提升只读副本。
- 对于 PostgreSQL 数据库实例，删除跨区域只读副本的源数据库实例时，只读副本的复制状态将设置为 `terminated`。不会提升只读副本。

  您必须手动提升只读副本或将其删除。

请求跨区域只读副本

要与源区域通信以请求创建跨区域只读副本，请求者（IAM 角色或 IAM 用户）必须能够访问源数据库实例和源区域。

请求者的 IAM 策略中的某些条件可能会导致请求失败。以下示例假定源数据库实例位于美国东部（俄亥俄州），只读副本在US East (N. Virginia)中创建。这些示例显示了请求者的 IAM 策略中导致请求失败的条件：

- 请求者的策略对于 `aws:RequestedRegion` 具有条件限制。

```
...
"Effect": "Allow",
"Action": "rds:CreateDBInstanceReadReplica",
"Resource": "*",
"Condition": {
    "StringEquals": {
        "aws:RequestedRegion": "us-east-1"
    }
}
```

  请求失败，因为该策略不允许访问源区域。要想请求成功，请指定源区域和目标区域。

```
...
"Effect": "Allow",
```

```
"Action": "rds:CreateDBInstanceReadReplica",
"Resource": "*",
"Condition": {
    "StringEquals": {
        "aws:RequestedRegion": [
            "us-east-1",
            "us-east-2"
        ]
    }
}
```

- 请求者的策略不允许访问源数据库实例。

```
...
"Effect": "Allow",
"Action": "rds:CreateDBInstanceReadReplica",
"Resource": "arn:aws:rds:us-east-1:123456789012:db:myreadreplica"
...
```

  要想请求成功，请指定源实例和副本。

```
...
"Effect": "Allow",
"Action": "rds:CreateDBInstanceReadReplica",
"Resource": [
    "arn:aws:rds:us-east-1:123456789012:db:myreadreplica",
    "arn:aws:rds:us-east-2:123456789012:db:mydbinstance"
]
...
```

- 请求者的政策拒绝 `aws:ViaAWSService`。

```
...
"Effect": "Allow",
"Action": "rds:CreateDBInstanceReadReplica",
"Resource": "*",
"Condition": {
    "Bool": {"aws:ViaAWSService": "false"}
}
```

  与源区域的通信由 RDS 代表请求者进行。要想请求成功，请不要拒绝 Amazon 服务发出的调用。

- 请求者的策略对于 `aws:SourceVpc` 或 `aws:SourceVpce` 具有条件限制。

这些请求可能会失败，因为 RDS 对远程区域进行调用时，它并非来自指定的 VPC 或 VPC 端点。

如果您需要使用可能导致请求失败的上述某个条件，则可以在策略中包含第二个带有 `aws:CalledVia` 的语句，以便请求成功。例如，您可以使用带有 `aws:CalledVia` 的 `aws:SourceVpce`，如下所示：

```
...
"Effect": "Allow",
"Action": "rds:CreateDBInstanceReadReplica",
"Resource": "*",
"Condition": {
    "Condition" : {
        "ForAnyValue:StringEquals" : {
          "aws:SourceVpce": "vpce-1a2b3c4d"
        }
     }
},
{
    "Effect": "Allow",
    "Action": [
        "rds:CreateDBInstanceReadReplica"
    ],
    "Resource": "*",
    "Condition": {
        "ForAnyValue:StringEquals": {
            "aws:CalledVia": [
                "rds.amazonaws.com"
            ]
        }
    }
}
```

有关更多信息，请参阅 IAM 用户指南 中的 [IAM 中的策略和权限](#) 。

授权只读副本

跨区域数据库只读副本创建请求返回后`success`，RDS 开始在后台创建副本。创建了 RDS 访问源数据库实例的授权。此授权将源数据库实例链接到只读副本，并允许 RDS 仅复制到指定的只读副本。

RDS 使用服务链接的 IAM 角色中的 `rds:CrossRegionCommunication` 权限对授权进行验证。如果副本获得授权，RDS 将与源区域通信并完成副本创建。

RDS 无权访问之前未经 `CreateDBInstanceReadReplica` 请求授权的数据库实例。只读副本创建完成后，将撤销授权。

RDS 使用服务链接的角色来验证源区域中的授权。如果在复制创建过程中删除了服务链接的角色，则创建将失败。

有关更多信息，请参阅 IAM 用户指南 中的使用服务相关角色。

使用 Amazon Security Token Service 凭证

来自全局 Amazon Security Token Service (Amazon STS) 端点的会话令牌仅在默认启用的 Amazon Web Services 区域（商业区域）中有效。在 `assumeRole` 中使用来自 Amazon STS API 操作的凭证时， 如果源区域是选择加入区域，请使用区域端点。否则，请求将失败。发生这种情况的原因是，您的凭证必须在两个区域都有效，而在使用区域 Amazon STS 端点时，仅选择加入区域有效。

要使用全局端点，请确保在操作中为两个区域都启用了该端点。在 Amazon STS 账户设置中将全局端点设置为 `Valid in all Amazon Web Services ##`。

同样的规则也适用于预签名 URL 参数中的凭证。

有关更多信息，请参阅《IAM 用户指南》中的在 Amazon Web Services 区域中管理 Amazon STS。

跨区域复制成本

跨区域复制时传输的数据可产生 Amazon RDS 数据传输费用。这些跨区域复制操作会对传输到源 Amazon Web Services 区域以外的数据收取费用：

- 创建只读副本时，Amazon RDS 将创建源实例的快照，并将快照传输到只读副本 Amazon Web Services 区域。
- 对于源数据库中做出的每个数据修改，Amazon RDS 都会将数据从源 Amazon Web Services 区域传输到只读副本 Amazon Web Services 区域。

对于 MySQL 和 MariaDB 实例，可以通过减少创建的跨区域只读副本的数量来降低数据传输费用。例如，假设您在一个 Amazon Web Services 区域中有一个源数据库实例并希望在另一个 Amazon Web Services 区域中有三个只读副本。在这种情况下，您只能从源数据库实例创建一个只读副本。您可以从第一个只读副本而不是源数据库实例创建另外两个副本。

例如，如果您在一个 `source-instance-1`有 Amazon Web Services 区域，则可以执行以下操作：

- 在新的 `read-replica-1`创建 Amazon Web Services 区域，指定 `source-instance-1` 作为源。

- 从 `read-replica-2` 创建 `read-replica-1`。
- 从 `read-replica-3` 创建 `read-replica-1`。

在此例中，只需支付从 `source-instance-1` 到 `read-replica-1` 的数据传输费用。您无需支付从 `read-replica-1` 到另外两个副本的数据传输费用，因为它们位于同一 Amazon Web Services 区域。如果从其它 Amazon Web Services 区域中的 `source-instance-1` 直接创建所有三个副本，则您需要为传输到所有三个副本的数据付费。

为 Amazon RDS 资源添加标签

Amazon RDS 标签是由您定义的名称-值对，与某种 Amazon RDS 资源（例如，数据库实例或数据库快照）关联。此名称也叫键。您可以选择为键提供值。

可以使用 Amazon Web Services Management Console、Amazon CLI 或 Amazon RDS API 添加、列出和删除 Amazon RDS 资源上的标签。在使用 CLI 或 API 时，确保提供要使用的 RDS 资源的 Amazon 资源名称（ARN）。有关构造 ARN 的详细信息，请参阅构建 Amazon RDS 的 ARN。

可以使用标签向您的 Amazon RDS 资源添加元数据。您可以使用标签自行添加有关数据库实例、快照、Aurora 集群等资源的符号。此举可以帮助您记录 Amazon RDS 资源。您还可以将标签与自动维护程序结合使用。

特别是，您可以将这些标签用于 IAM policy。您可以使用这些标签管理对 Amazon RDS 资源的访问，并控制可将什么操作应用于这些资源。您还可以将具有类似标签的资源的费用分组在一起，使用标签来跟踪成本。

您可以标记以下 Amazon RDS 资源：

- 数据库实例数
- 数据库集群
- Aurora 全局集群
- 数据库集群端点
- 只读副本
- 数据库快照
- 数据库集群快照
- 预留数据库实例
- 事件订阅
- 数据库选项组
- 数据库参数组
- 数据库集群参数组
- 数据库子网组
- RDS 代理
- RDS Proxy 端点

- 蓝绿部署
- 零 ETL 集成

> **Note**
>
> 目前，您无法使用 Amazon Web Services Management Console 标记 RDS 代理和 RDS 代理端点。

主题

为什么要使用 Amazon RDS 资源标签？

您可使用标签执行以下操作：

- 按应用程序、项目、部门、环境等对 RDS 资源进行分类。例如，您可以使用标签键定义类别，其中标签值是类别中的项目。您可以创建标签 `environment=prod`。或者，您可以定义标签键 `project` 和标签值 `Salix`，这表示 Amazon RDS 资源将分配给 Salix 项目。
- 自动执行资源管理任务。例如，您可以为已标记 `environment=prod` 的实例创建一个维护时段，该时段不同于已标记 `environment=test` 的实例的时段。您还可以为已标记 `environment=prod` 的实例配置自动数据库快照。
- 使用 IAM 策略控制对 RDS 资源的访问。您可以使用全局 `aws:ResourceTag/`*`tag-key`* 条件键完成此操作。例如，策略可能只允许 `DBAdmin` 组中的用户修改已标记 `environment=prod` 的数据库实例。有关使用 IAM 策略管理对已标记资源的访问的信息，请参阅 Amazon RDS 的 Identity and Access Management，以及《Amazon Identity and Access Management 用户指南》中的控制对 Amazon 资源的访问。
- 根据标签监控资源。例如，您可以为已标记 `environment=prod` 的数据库实例创建 Amazon CloudWatch 控制面板。

• 通过将具有类似标签的资源的费用分组在一起来跟踪成本。例如，如果您使用 `project=Salix` 标记与 Salix 项目关联的 RDS 资源，则可以为该项目生成成本报告并为该项目分配费用。有关更多信息，请参阅 如何将 Amazon 账单与 Amazon RDS 中的标签结合使用。

Amazon RDS 资源标签的工作原理

Amazon不会对您的标签应用任何语义意义。标签严格按字符串进行解释。

主题

Amazon RDS 中的标签集

每个 Amazon RDS 资源都有一个名为标签集的容器。该容器包含所有分配给该资源的标签。一个资源只有一个标签集。

一个标签集包含 0 至 50 个标签。如果向 RDS 资源添加一个标签，而该标签的键与某个现有资源标签相同，则新值将覆盖旧值。

Amazon RDS 中的标签结构

RDS 标签的结构如下所示：

标签键

键是标签的必需名称。该字符串值的长度必须在 1 到 128 个 Unicode 字符之间，并且不能带有前缀 `aws:` 或 `rds:`。字符串只能包含 Unicode 字母、数字、空格、`_`、`.`、`:`、`/`、`=`、`+`、`-` 和 `@` 的集合。Java 正则表达式是 `"^([\\p{L}\\p{Z}\\p{N}_.:/=+\\-@]*)$"`。标签键区分大小写。因此，键 `project` 和 `Project` 是不同的。

键对于标签集是唯一的。例如，标签集中不能有键相同但值不同的键-值对，如 `project=Trinity` 和 `project=Xanadu`。

标签值

值是标签的可选字符串值。该字符串值的长度必须在 1 到 256 个 Unicode 字符之间。字符串只能包含 Unicode 字母、数字、空格、`_`、`.`、`:`、`/`、`=`、`+`、`-` 和 `@` 的集合。Java 正则表达式是 `"^([\\p{L}\\p{Z}\\p{N}_.:/=+\\-@]*)$"`。标签值区分大小写。因此，值 `prod` 和 `Prod` 是不同的。

在标签集中，值无需具有唯一性，且可为空值。例如，在 `project=Trinity` 和 `cost-center=Trinity` 的标签集中，可以存在键-值对。

符合标记条件的 Amazon RDS 资源

您可以标记以下 Amazon RDS 资源：

- 数据库实例
- 数据库集群
- 数据库集群端点
- 只读副本
- 数据库快照
- 数据库集群快照
- 预留数据库实例
- 事件订阅
- 数据库选项组
- 数据库参数组
- 数据库集群参数组
- 数据库子网组
- RDS 代理
- RDS 代理端点

> **Note**
>
> 目前，您无法使用 Amazon Web Services Management Console 标记 RDS 代理和 RDS 代理端点。

- 蓝绿部署
- 零 ETL 集成（预览版）

如何将 Amazon 账单与 Amazon RDS 中的标签结合使用

使用标签来组织 Amazon 账单以反映您自身的成本结构。要执行此操作，请注册以获取包含标签键值的 Amazon Web Services 账户账单。然后，如需查看组合资源的成本，请按有同样标签键值的资源组织您的账单信息。例如，您可以将特定的应用程序名称用作几个资源的标签，然后组织账单信息，以查看在数个服务中的使用该应用程序的总成本。有关更多信息，请参阅 Amazon Billing 用户指南 中的使用成本分配标签。

成本分配标签如何与数据库快照结合使用

您可以向数据库快照添加标签。但是，您的账单不会反映此分组。必须满足以下条件，才能将成本分配标签应用于数据库快照：

- 标签必须附加到父数据库实例。
- 父数据库实例必须与数据库快照在同一 Amazon Web Services 账户中。
- 父数据库实例必须与数据库快照在同一 Amazon Web Services 区域中。

如果数据库快照与父数据库实例不在同一区域中，或者如果删除了父数据库实例，则这些快照将被视为孤立快照。孤立的数据库快照不支持成本分配标签。孤立快照的成本会汇总到单个未标记的行项目中。满足以下条件时，跨账户数据库快照将不会被视为孤立快照：

- 这些快照与父数据库实例在同一区域中。
- 父数据库实例由源账户所有。

> **Note**
>
> 如果父数据库实例由其他账户所有，则成本分配标签不适用于目标账户中的跨账户快照。

标记 Amazon RDS 资源的最佳实践

在使用标签时，建议您遵循以下最佳实践：

- 记录组织中所有团队所遵循的标签使用惯例。具体而言，请确保名称既具有描述性又保持一致。例如，对格式 `environment:prod` 进行标准化，而不是使用 `env:production` 标记某些资源。

> **Important**
> 请勿在标签中存储个人身份信息（PII）或其他机密或敏感信息。

- 自动添加标记以确保一致性。例如，您可以使用以下方法：
  - 在 Amazon CloudFormation 模板中包含标签。在使用模板创建资源时，系统会自动标记资源。
  - 使用 Amazon Lambda 函数定义和应用标签。
  - 创建一个 SSM 文档，其中包含向 RDS 资源添加标签的步骤。
- 仅在必要时使用标签。您最多可以为单个 RDS 资源添加 50 个标签，但最佳实践是避免不必要的标签激增和复杂性。
- 定期检查标签的相关性和准确性。根据需要删除或修改已过时的标签。
- 考虑在 Amazon Web Services Management Console中使用 Amazon 标签编辑器创建标签。您可以使用标签编辑器同时向多个受支持的 Amazon 资源（包括 RDS 资源）添加标签。有关更多信息，请参阅 Amazon Resource Groups 用户指南中的标签编辑器。

将标签复制到数据库快照

在创建或还原数据库实例时，您可以指定将数据库实例中的标签复制到数据库实例的快照。复制标签可确保数据库快照的元数据与源数据库实例的元数据匹配。它还确保数据库快照的任何访问策略也与源数据库实例的访问策略相匹配。

您可以为以下操作指定将标签复制到数据库快照：

- 创建数据库实例
- 还原数据库实例
- 创建只读副本
- 复制数据库快照

要复制上述操作的标签，请在 Amazon Web Services Management Console中选择将标签复制到快照，或者在 Amazon CLI 中指定 `--copy-tags-to-snapshot`。

在大多数情况下，默认不复制标签。不过，从数据库快照还原数据库实例时，RDS 会检查您是否指定新标签。如果是，新标签将添加到还原的数据库实例中。如果没有新的标签，RDS 会在创建快照时将源数据库实例中的标签添加到还原的数据库实例中。

为防止来自源数据库实例的标签添加到还原的数据库实例中，建议在还原数据库实例时指定新标签。

> **Note**
>
> 在某些情况下，您可以为 Amazon CLI 命令 create-db-snapshot 的 `--tags` 参数包含一个值。或者，您可以为 CreateDBSnapshot API 操作提供至少一个标签。在这些情况下，RDS 不会将标签从源数据库实例复制到新的数据库快照。即使源数据库实例已开启 `--copy-tags-to-snapshot`（`CopyTagsToSnapshot`）选项，此功能也适用。
> 如果您采用此方法，则可以从数据库快照创建数据库实例的副本。这种方法可避免添加不适用于新数据库实例的标签。您可以使用 Amazon CLI `create-db-snapshot` 命令（或 `CreateDBSnapshot` RDS API 操作）创建数据库快照。创建数据库快照后，可以按本主题后面所述添加标签。

在 Amazon RDS 中添加和删除标签

您可执行以下操作：

- 在创建资源的同时创建标签，例如在运行 Amazon CLI 命令 `create-db-instance` 时。
- 使用命令 `add-tags-to-resource` 向现有资源添加标签。
- 使用命令 `list-tags-for-resource` 列出与特定资源关联的标签。
- 使用命令 `add-tags-to-resource` 更新标签。
- 使用命令 `remove-tags-from-resource` 从资源中删除标签。

以下步骤说明如何对与数据库实例相关的资源执行典型的标记操作。注意，对标签进行缓存以用于授权。因此，当您在 Amazon RDS 资源上添加或更新标签时，可能需要等待几分钟，之后修改才能生效。

控制台

为 Amazon RDS 资源加标签的过程对于所有资源均类似。以下过程展示如何为 Amazon RDS 数据库实例加标签。

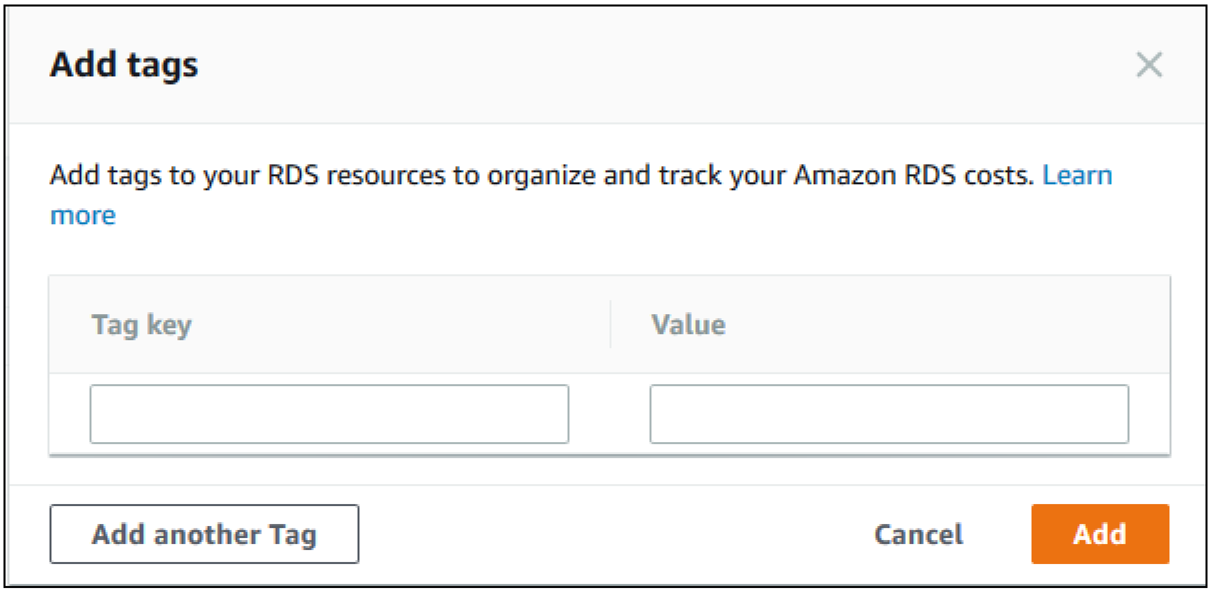
要向数据库实例添加标签，请执行以下操作：

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。

2. 在导航窗格中，选择 Databases (数据库)。

> **Note**
>
> 要在数据库窗格中筛选数据库实例的列表，对于筛选数据库，输入文本字符串。只会显示包含该字符串的数据库实例。

3. 选择要标记的数据库实例的名称以显示其详细信息。
4. 在详细信息部分中，向下滚动到标签部分。
5. 选择 Add。将显示添加标签窗口。



6. 为标签键和值输入一个值。
7. 要添加其他标签，您可以选择添加其他标签，并为其标签键和值输入一个值。

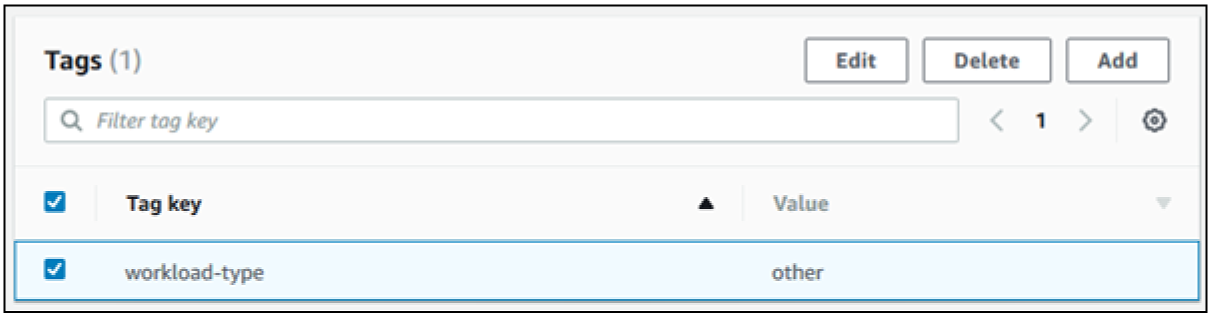
   将该步骤重复执行所需的次数。
8. 选择 Add。

删除数据库实例的标签

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。

> **Note**
>
> 要在数据库窗格中筛选数据库实例的列表，请在筛选数据库框输入文本字符串。只会显示包含该字符串的数据库实例。

3. 选择数据库实例的名称以显示其详细信息。
4. 在详细信息部分中，向下滚动到标签部分。
5. 选择要删除的标签。



6. 选择 Delete (删除)，然后在 Delete tags (删除标签) 窗口中选择 Delete (删除)。

Amazon CLI

可以使用 Amazon CLI 为数据库实例添加、列出或删除标签。

- 要将一个或多个标签添加到 Amazon RDS 资源，请使用 Amazon CLI 命令 `add-tags-to-resource`。
- 要列出 Amazon RDS 资源上的标签，请使用 Amazon CLI 命令 `list-tags-for-resource`。
- 要从 Amazon RDS 资源中删除一个或多个标签，请使用 Amazon CLI 命令 `remove-tags-from-resource`。

要了解有关如何构建所需 ARN 的更多信息，请参阅构建 Amazon RDS 的 ARN。

RDS API

您可使用 Amazon RDS API 为数据库实例添加、列出或删除标签。

- 要向 Amazon RDS 资源添加标签，请使用 `AddTagsToResource` 操作。
- 要列出分配给 Amazon RDS 资源的标签，请使用 `ListTagsForResource`。
- 要从 Amazon RDS 资源删除标签，请使用 `RemoveTagsFromResource` 操作。

要了解有关如何构建所需 ARN 的更多信息，请参阅构建 Amazon RDS 的 ARN。

在通过 Amazon RDS API 使用 XML 时，标签会使用如下架构：

```
<Tagging>
      <TagSet>
          <Tag>
              <Key>Project</Key>
              <Value>Trinity</Value>
          </Tag>
          <Tag>
              <Key>User</Key>
              <Value>Jones</Value>
          </Tag>
      </TagSet>
  </Tagging>
```

下表提供了允许使用的 XML 标签及其特征的列表。Key 和 Value 的值区分大小写。例如，project=Trinity 和 PROJECT=Trinity 是不同的标签。

标签元素	描述
标签集	标签集是分配给 Amazon RDS 资源的所有标签的容器。每个资源只能有一个标签集。您只可以通过 Amazon RDS API 使用标签集。
标签	标签是用户定义的键值对。一个标签集中可以有 1 到 50 个标签。
密钥	密钥是标签必需的名称。有关限制，请参阅 Amazon RDS 中的标签结构。 该字符串值的长度可以在 1 到 128 个 Unicode 字符之间，并且不能带有前缀 aws: 或 rds:。该字符串只能包含 Unicode 字母、数字、空格、“_”、“.”、“/”、“=”、“+”、“-”的集合（Java 正则表达式：“^([\\p{L}\\p{Z}\\p{N}_.:/=+\\-]*)$”）。 密钥在标签集中必须具有唯一性。例如，标签集中不能有键相同但值不同的键-值对，如 project/Trinity 和 project/Xanadu。
值	值是标签的可选内容。有关限制，请参阅 Amazon RDS 中的标签结构。 该字符串值的长度可以在 1 到 256 个 Unicode 字符之间，并且不能带有前缀 aws: 或 rds:。该字符串只能包含 Unicode 字母、数字、空格、“_”、“.”、“/”、“=”、“+”、“-”的集合（Java 正则表达式：“^([\\p{L}\\p{Z}\\p{N}_.:/=+\\-]*)$”）。

标签元素	描述
	在标签集中，值不必具有唯一性，且可为空值。例如，在“项目/Trinity”和“成本 - 中心/Trinity”的一个标签集中，可以存在键值对。

教程：使用标签指定要停止的数据库实例

本教程假设您在开发或测试环境中有多个数据库实例。您需要将这些数据库数据库实例保留几天。一些数据库实例在夜间运行测试，而其他数据库实例可在夜间停止运行，并在第二天重新开始运行。

以下教程说明如何为适合在夜间停止运行的数据库实例分配标签。本教程说明脚本如何检测哪些数据库实例具有该标签及随后停止已标记的数据库实例。在此示例中，键值对的值部分无关紧要。存在`stoppable`标签表示数据库实例具有此用户定义的属性。

在以下教程中，用于标记的命令和 API 使用 ARN，这可让 RDS 跨 Amazon 区域、Amazon 账户以及不同类型的资源（这些资源可能具有相同的短名称）无缝工作。您可以在数据库实例上运行的 CLI 命令中指定 ARN 代替数据库实例 ID。

指定要停止的数据库实例

1. 确定您要指定为“可停止”的数据库实例的 ARN。

   在以下示例中，将自己的数据库实例名称替换为 *dev-test-db-instance*。在使用 ARN 参数的后续命令中，替换自己的数据库实例的 ARN。ARN 包含您自己的 Amazon 账户 ID 和数据库实例所在的 Amazon 区域的名称。

   ```
   $ aws rds describe-db-instances --db-instance-identifier dev-test-db-instance \
     --query "*[].{DBInstance:DBInstanceArn}" --output text
   arn:aws:rds:us-east-1:123456789102:db:dev-test-db-instance
   ```

2. 将标签 `stoppable` 添加至此数据库实例。

   由您选择此标签的名称。由于此示例将标签视为存在或不存在的属性，因此它忽略了 `Value=` 参数的 `--tags` 部分。这种方法意味着，您可以避免设计一种在名称中对所有相关信息进行编码的命名约定。在此类约定中，您可以在数据库实例名称或其他资源的名称中对信息进行编码。

   ```
   $ aws rds add-tags-to-resource \
     --resource-name arn:aws:rds:us-east-1:123456789102:db:dev-test-db-instance \
     --tags Key=stoppable
   ```

3. 确认数据库实例中存在标签。

以下命令以 JSON 格式和以制表符分隔的纯文本检索数据库实例中的标签信息。

```
$ aws rds list-tags-for-resource \
  --resource-name arn:aws:rds:us-east-1:123456789102:db:dev-test-db-instance
{
    "TagList": [
        {
            "Key": "stoppable",
            "Value": ""

        }
    ]
}
aws rds list-tags-for-resource \
  --resource-name arn:aws:rds:us-east-1:123456789102:db:dev-test-db-instance --
output text
TAGLIST stoppable
```

4. 停止所有被指定为 `stoppable` 的数据库实例。

以下示例创建一个文本文件，其中列出了所有数据库实例。Shell 命令循环访问该列表，并检查每个数据库实例是否都标记有相关属性，并为每个数据库实例运行命令 `aws rds stop-db-instance`。

```
$ aws rds describe-db-instances --query "*[].[DBInstanceArn]" --output text >/tmp/
db_instance_arns.lst
$ for arn in $(cat /tmp/db_instance_arns.lst)
do
  match="$(aws rds list-tags-for-resource --resource-name $arn --output text | grep
 stoppable)"
  if [[ ! -z "$match" ]]
  then
      echo "DB instance $arn is tagged as stoppable. Stopping it now."
# Note that you need to get the DB instance identifier from the ARN.
      dbid=$(echo $arn | sed -e 's/.*://')
      aws rds stop-db-instance --db-instance-identifier $dbid
  fi
done

DB instance arn:arn:aws:rds:us-east-1:123456789102:db:dev-test-db-instance is
 tagged as stoppable. Stopping it now.
```

```
{
    "DBInstance": {
        "DBInstanceIdentifier": "dev-test-db-instance",
        "DBInstanceClass": "db.t3.medium",
        ...
```

您可以在每天结束时运行与上一个脚本类似的脚本，以确保停止非必要的数据库实例。还可以使用实用程序（例如 cron）安排任务，如每晚执行此类检查。例如，您可以这样做，以防某些数据库实例被误运行。在此，您可以对准备待检查数据库实例列表的命令进行微调。

以下命令生成数据库实例列表，但只生成处于`available`状态的数据库实例的列表。该脚本可以忽略已停止的数据库实例，因为它们将具有不同的状态值，如`stopped`或`stopping`。

```
$ aws rds describe-db-instances \
  --query '*[].{DBInstanceArn:DBInstanceArn,DBInstanceStatus:DBInstanceStatus}|[?
DBInstanceStatus == `available`]|[].{DBInstanceArn:DBInstanceArn}' \
  --output text
arn:aws:rds:us-east-1:123456789102:db:db-instance-2447
arn:aws:rds:us-east-1:123456789102:db:db-instance-3395
arn:aws:rds:us-east-1:123456789102:db:dev-test-db-instance
arn:aws:rds:us-east-1:123456789102:db:pg2-db-instance
```

> **Tip**
>
> 您可以将分配标签的过程与查找带这些标签的数据库实例的过程结合使用，以其他方式降低成本。我们以用于开发和测试的数据库实例为例。在这种情况下，您可以指定在每天结束时删除一些数据库实例。或者，您可以指定它们在预期使用率较低的时期，将其数据库实例更改为小型数据库实例类。

Amazon RDS 中的 Amazon 资源名称（ARN）

Amazon Web Services 中创建的资源分别使用 Amazon Resource Name (ARN) 进行唯一标识。对于某些 Amazon RDS 操作，您必须通过指定 Amazon RDS 资源的 ARN 来唯一标识该资源。例如，当您创建 RDS 数据库实例只读副本时，必须提供源数据库实例的 ARN。

有关构造 ARN 和获取现有 ARN 的信息，请参阅以下主题。

主题

构建 Amazon RDS 的 ARN

Amazon Web Services 中创建的资源分别使用 Amazon Resource Name (ARN) 进行唯一标识。您可以使用以下语法为 Amazon RDS 资源构建 ARN。

```
arn:aws:rds:<region>:<account number>:<resourcetype>:<name>
```

区域名称	区域	端点	协议
美国东部（俄亥俄州）	us-east-2	rds.us-east-2.amazonaws.com	HTTPS
		rds-fips.us-east-2.api.aws	HTTPS
		rds.us-east-2.api.aws	HTTPS
		rds-fips.us-east-2.amazonaws.com	HTTPS
美国东部（弗吉尼亚州北部）	us-east-1	rds.us-east-1.amazonaws.com	HTTPS
		rds-fips.us-east-1.api.aws	HTTPS
		rds-fips.us-east-1.amazonaws.com	HTTPS
		rds.us-east-1.api.aws	HTTPS
美国西部（加利福	us-west-1	rds.us-west-1.amazonaws.com	HTTPS
		rds.us-west-1.api.aws	HTTPS

区域名称	区域	端点	协议
尼亚北部）		rds-fips.us-west-1.amazonaws.com rds-fips.us-west-1.api.aws	HTTPS HTTPS
美国西部（俄勒冈州）	us-west-2	rds.us-west-2.amazonaws.com rds-fips.us-west-2.amazonaws.com rds.us-west-2.api.aws rds-fips.us-west-2.api.aws	HTTPS HTTPS HTTPS HTTPS
非洲（开普敦）	af-south-1	rds.af-south-1.amazonaws.com rds.af-south-1.api.aws	HTTPS HTTPS
亚太地区（香港）	ap-east-1	rds.ap-east-1.amazonaws.com rds.ap-east-1.api.aws	HTTPS HTTPS
亚太地区（海得拉巴）	ap-south-2	rds.ap-south-2.amazonaws.com rds.ap-south-2.api.aws	HTTPS HTTPS
亚太地区（雅加达）	ap-southeast-3	rds.ap-southeast-3.amazonaws.com rds.ap-southeast-3.api.aws	HTTPS HTTPS
亚太地区（马来西亚）	ap-southeast-5	rds.ap-southeast-5.amazonaws.com	HTTPS
亚太地区（墨尔本）	ap-southeast-4	rds.ap-southeast-4.amazonaws.com rds.ap-southeast-4.api.aws	HTTPS HTTPS
亚太地区（孟买）	ap-south-1	rds.ap-south-1.amazonaws.com rds.ap-south-1.api.aws	HTTPS HTTPS

区域名称	区域	端点	协议
亚太地区（大阪）	ap-northeast-3	rds.ap-northeast-3.amazonaws.com rds.ap-northeast-3.api.aws	HTTPS HTTPS
亚太地区（首尔）	ap-northeast-2	rds.ap-northeast-2.amazonaws.com rds.ap-northeast-2.api.aws	HTTPS HTTPS
亚太地区（新加坡）	ap-southeast-1	rds.ap-southeast-1.amazonaws.com rds.ap-southeast-1.api.aws	HTTPS HTTPS
亚太地区（悉尼）	ap-southeast-2	rds.ap-southeast-2.amazonaws.com rds.ap-southeast-2.api.aws	HTTPS HTTPS
亚太地区（台北）	ap-east-2	rds.ap-east-2.amazonaws.com	HTTPS
亚太地区（泰国）	ap-southeast-7	rds.ap-southeast-7.amazonaws.com	HTTPS
亚太地区（东京）	ap-northeast-1	rds.ap-northeast-1.amazonaws.com rds.ap-northeast-1.api.aws	HTTPS HTTPS
加拿大（中部）	ca-central-1	rds.ca-central-1.amazonaws.com rds.ca-central-1.api.aws rds-fips.ca-central-1.api.aws rds-fips.ca-central-1.amazonaws.com	HTTPS HTTPS HTTPS HTTPS
加拿大西部（卡尔加里）	ca-west-1	rds.ca-west-1.amazonaws.com rds-fips.ca-west-1.amazonaws.com	HTTPS HTTPS

区域名称	区域	端点	协议
中国（北京）	cn-north-1	rds.cn-north-1.amazonaws.com.cn rds.cn-north-1.api.amazonwebservices.com.cn	HTTPS HTTPS
中国（宁夏）	cn-northwest-1	rds.cn-northwest-1.amazonaws.com.cn rds.cn-northwest-1.api.amazonwebservices.com.cn	HTTPS HTTPS
欧洲地区（法兰克福）	eu-central-1	rds.eu-central-1.amazonaws.com rds.eu-central-1.api.aws	HTTPS HTTPS
欧洲地区（爱尔兰）	eu-west-1	rds.eu-west-1.amazonaws.com rds.eu-west-1.api.aws	HTTPS HTTPS
欧洲地区（伦敦）	eu-west-2	rds.eu-west-2.amazonaws.com rds.eu-west-2.api.aws	HTTPS HTTPS
欧洲地区（米兰）	eu-south-1	rds.eu-south-1.amazonaws.com rds.eu-south-1.api.aws	HTTPS HTTPS
欧洲地区（巴黎）	eu-west-3	rds.eu-west-3.amazonaws.com rds.eu-west-3.api.aws	HTTPS HTTPS
欧洲（西班牙）	eu-south-2	rds.eu-south-2.amazonaws.com rds.eu-south-2.api.aws	HTTPS HTTPS
欧洲地区（斯德哥尔摩）	eu-north-1	rds.eu-north-1.amazonaws.com rds.eu-north-1.api.aws	HTTPS HTTPS

区域名称	区域	端点	协议
欧洲（苏黎世）	eu-central-2	rds.eu-central-2.amazonaws.com rds.eu-central-2.api.aws	HTTPS HTTPS
以色列（特拉维夫）	il-central-1	rds.il-central-1.amazonaws.com rds.il-central-1.api.aws	HTTPS HTTPS
墨西哥（中部）	mx-central-1	rds.mx-central-1.amazonaws.com	HTTPS
中东（巴林）	me-south-1	rds.me-south-1.amazonaws.com rds.me-south-1.api.aws	HTTPS HTTPS
中东（阿联酋）	me-central-1	rds.me-central-1.amazonaws.com rds.me-central-1.api.aws	HTTPS HTTPS
南美洲（圣保罗）	sa-east-1	rds.sa-east-1.amazonaws.com rds.sa-east-1.api.aws	HTTPS HTTPS

下表显示在构建特定 Amazon RDS 资源类型的 ARN 时应使用的格式。

资源类型	ARN 格式
数据库实例	arn:aws:rds:*<region>*:*<account>* :db:*<name>* 例如： `arn:aws:rds: us-east-2 :123456789012 :db:my-mysql-instance-1`
数据库群集	arn:aws:rds:*<region>*:*<account>* :cluster:*<name>* 例如：

资源类型	ARN 格式
	`arn:aws:rds: us-east-2 :123456789012 :cluster: my-aurora-cluster-1`
事件订阅	arn:aws:rds:*<region>*:*<account>* :es:*<name>* 例如： `arn:aws:rds: us-east-2 :123456789012 :es:my-subscription`
数据库选项组	arn:aws:rds:*<region>*:*<account>* :og:*<name>* 例如： `arn:aws:rds: us-east-2 :123456789012 :og:my-og`
数据库参数组	arn:aws:rds:*<region>*:*<account>* :pg:*<name>* 例如： `arn:aws:rds: us-east-2 :123456789012 :pg:my-param-enable-logs`
数据库集群参数组	arn:aws:rds:*<region>*:*<account>* :cluster-pg:*<name>* 例如： `arn:aws:rds: us-east-2 :123456789012 :cluster-pg: my-cluster-param-timezone`
预留数据库实例	arn:aws:rds:*<region>*:*<account>* :ri:*<name>* 例如： `arn:aws:rds: us-east-2 :123456789012 :ri:my-reserved-postgresql`

资源类型	ARN 格式
数据库安全组	arn:aws:rds:*<region>*:*<account>* :secgrp:*<name>* 例如： `arn:aws:rds: us-east-2 :123456789012 :secgrp:my-public`
自动数据库快照	arn:aws:rds:*<region>*:*<account>* :snapshot:rds:*<name>* 例如： `arn:aws:rds: us-east-2 :123456789012 :snapshot:rds: my-mysql-db-2019-07-22-07-23`
自动数据库集群快照	arn:aws:rds:*<region>*:*<account>* :cluster-snapshot:rds:*<name>* 例如： `arn:aws:rds: us-east-2 :123456789012 :cluster-snapshot:rds: my-aurora-cluster-2019-07-22-16-16`
手动数据库快照	arn:aws:rds:*<region>*:*<account>* :snapshot:*<name>* 例如： `arn:aws:rds: us-east-2 :123456789012 :snapshot: my-mysql-db-snap`
手动数据库集群快照	arn:aws:rds:*<region>*:*<account>* :cluster-snapshot:*<name>* 例如： `arn:aws:rds: us-east-2 :123456789012 :cluster-snapshot: my-aurora-cluster-snap`

资源类型	ARN 格式
数据库子网组	arn:aws:rds:*<region>*:*<account>* :subgrp:*<name>* 例如： `arn:aws:rds: us-east-2 :123456789012 :subgrp:my-subnet-10`

获取 Amazon RDS 的现有 ARN

您可以使用 Amazon Web Services Management Console、Amazon Command Line Interface (Amazon CLI) 或 RDS API 来获取 RDS 资源的 ARN。

控制台

要从 Amazon Web Services Management Console中获取一个 ARN，请导航到要获取 ARN 的资源，然后查看该资源的详细信息。

例如，您可以从数据库实例详细信息的配置选项卡获取数据库实例的 ARN。

Amazon CLI

要从 Amazon CLI 为特定 RDS 资源获取 ARN，请对该资源使用 `describe` 命令。下表显示了每个 Amazon CLI 命令，以及与命令配合使用以获取 ARN 的 ARN 属性。

Amazon CLI command	ARN 属性
describe-event-subscriptions	EventSubscriptionArn
describe-certificates	CertificateArn
describe-db-parameter-groups	DBParameterGroupArn
describe-db-cluster-parameter-groups	DBClusterParameterGroupArn
describe-db-instances	DBInstanceArn
describe-db-security-groups	DBSecurityGroupArn

Amazon CLI command	ARN 属性
describe-db-snapshots	DBSnapshotArn
describe-events	SourceArn
describe-reserved-db-instances	ReservedDBInstanceArn
describe-db-subnet-groups	DBSubnetGroupArn
describe-option-groups	OptionGroupArn
describe-db-clusters	DBClusterArn
describe-db-cluster-snapshots	DBClusterSnapshotArn

例如，以下 Amazon CLI 命令获取数据库实例的 ARN。

Example

对于 Linux、macOS 或 Unix：

```
aws rds describe-db-instances \
--db-instance-identifier DBInstanceIdentifier \
--region us-west-2 \
--query "*[].{DBInstanceIdentifier:DBInstanceIdentifier,DBInstanceArn:DBInstanceArn}"
```

对于 Windows：

```
aws rds describe-db-instances ^
--db-instance-identifier DBInstanceIdentifier ^
--region us-west-2 ^
--query "*[].{DBInstanceIdentifier:DBInstanceIdentifier,DBInstanceArn:DBInstanceArn}"
```

该命令的输出如下所示：

```
[
    {
        "DBInstanceArn": "arn:aws:rds:us-west-2:account_id:db:instance_id",
        "DBInstanceIdentifier": "instance_id"
```

```
    }
]
```

RDS API

要为特定 RDS 资源获取 ARN，您可以调用以下 RDS API 操作并使用如下所示的 ARN 属性。

RDS API 操作	ARN 属性
DescribeEventSubscriptions	EventSubscriptionArn
DescribeCertificates	CertificateArn
DescribeDBParameterGroups	DBParameterGroupArn
DescribeDBClusterParameterGroups	DBClusterParameterGroupArn
DescribeDBInstances	DBInstanceArn
DescribeDBSecurityGroups	DBSecurityGroupArn
DescribeDBSnapshots	DBSnapshotArn
DescribeEvents	SourceArn
DescribeReservedDBInstances	ReservedDBInstanceArn
DescribeDBSubnetGroups	DBSubnetGroupArn
DescribeOptionGroups	OptionGroupArn
DescribeDBClusters	DBClusterArn
DescribeDBClusterSnapshots	DBClusterSnapshotArn

使用 Amazon RDS 数据库实例的存储

要指定在 Amazon RDS 中存储数据的方式，请在创建或修改数据库实例时选择存储类型并提供存储大小。以后，您可以通过修改数据库实例来增加存储量或更改存储类型。有关将哪些个存储类型用于工作负载的更多信息，请参阅Amazon RDS 存储类型。

主题

- 增加数据库实例存储容量
- 使用 Amazon RDS 存储自动扩展功能自动管理容量
- 升级数据库实例的存储文件系统
- 修改预调配 IOPS SSD 存储的设置
- I/O 密集型存储修改
- 修改通用型 SSD（gp3）存储的设置
- 使用专用日志卷（DLV）

增加数据库实例存储容量

如果需要空间来存储更多数据，则可以扩展现有数据库实例的存储。为此，您可以使用 Amazon RDS 控制台、Amazon RDS API 或 Amazon Command Line Interface（Amazon CLI）。有关存储限制的信息，请参阅Amazon RDS 数据库实例存储。

> **Note**
>
> 您无法在已分配数据库实例的存储后减少存储量。当您增加分配的存储空间时，必须至少增加 10%。如果您尝试将该值增加不到 10%，则会得到错误。
> 仅通用型 SSD 和预调配 IOPS SSD 存储类型支持扩展 RDS for SQL Server 数据库实例的存储。

要监控数据库实例的可用存储量以便您能够在需要时做出响应，我们建议您创建 Amazon CloudWatch 警报。有关设置 CloudWatch 警报的更多信息，请参阅使用 CloudWatch 警报。

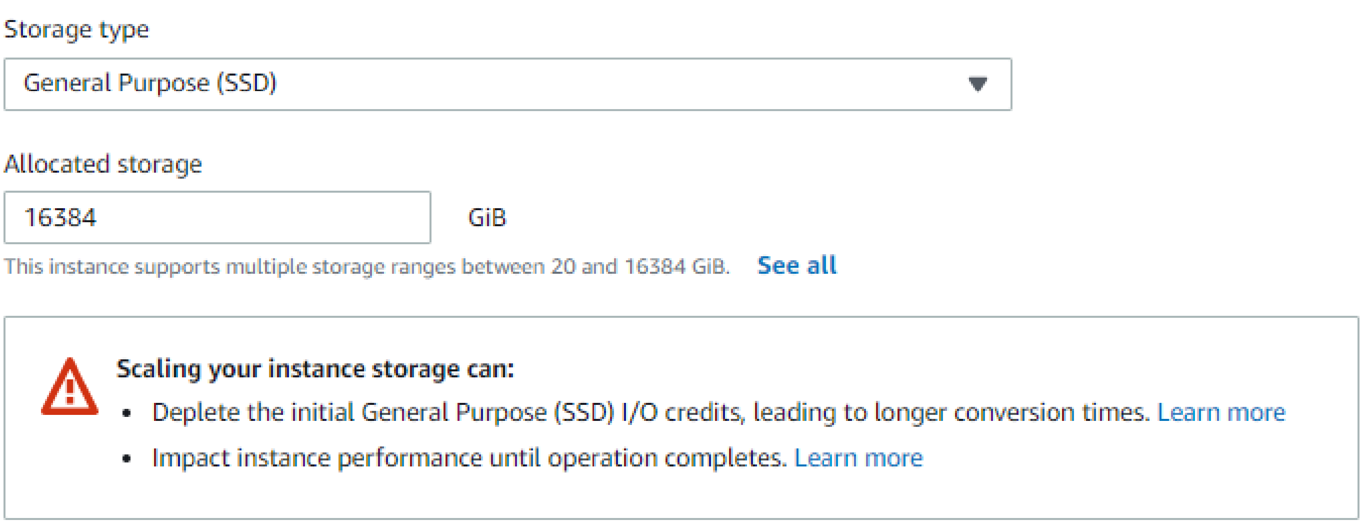
扩展存储通常不会导致数据库实例的任何中断或性能下降。在修改数据库实例的存储大小后，数据库实例的状态为 storage-optimization。

> **Note**
>
> 存储优化可能耗时数小时。六（6）小时后或在实例上完成存储优化后，您才能进行进一步的存储修改。您可以在 Amazon Web Services Management Console中或使用 [describe-db-instances](#) Amazon CLI 命令查看存储优化进度。

控制台

增加数据库实例的存储

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。
3. 选择要修改的数据库实例。
4. 选择修改。
5. 为 Allocated Storage (分配的存储) 输入新值。它必须大于当前值。



6. 选择继续，进入下一个屏幕。
7. 选中修改计划部分中的立即应用来立即将存储更改应用于数据库实例。

   或选择 Apply during the next scheduled maintenance window (在下一个计划的维护时段内应用) 以在下一个维护时段内应用更改。
8. 根据需要设置完毕后，选择修改数据库实例。

Amazon CLI

要增加数据库实例的存储，请使用 Amazon CLI 命令 [modify-db-instance]。设置以下参数：

- `--allocated-storage` – 要为数据库实例分配的存储量（以 GiB 为单位）。
- `--apply-immediately` – 使用 `--apply-immediately` 可立即应用存储更改。

  或使用 `--no-apply-immediately` (原定设置) 以在下一个维护时段内应用更改。在应用更改时，将会立即发生中断。

有关存储的更多信息，请参阅 Amazon RDS 数据库实例存储。

RDS API

要增加数据库实例的存储，请使用 Amazon RDS API 操作 `ModifyDBInstance`。设置以下参数：

- `AllocatedStorage` – 要为数据库实例分配的存储量（以 GiB 为单位）。
- `ApplyImmediately` – 将此选项设置为 `True` 可立即应用存储更改。将此选项设置为 `False` (原定设置) 可在下一个维护时段内应用更改。在应用更改时，将会立即发生中断。

有关存储的更多信息，请参阅 Amazon RDS 数据库实例存储。

使用 Amazon RDS 存储自动扩展功能自动管理容量

如果您的工作负载是不可预测的，则可以为 Amazon RDS 数据库实例启用存储自动扩展。为此，您可以使用 Amazon RDS 控制台、Amazon RDS API 或 Amazon CLI。

例如，您可以将此功能用于用户快速采用的新的移动游戏应用程序。在这种情况下，快速增加的工作负载可能会超过可用的数据库存储。要避免必须手动扩展数据库存储，您可以使用 Amazon RDS 存储自动扩展。

如果启用了存储自动扩展，在 Amazon RDS 检测到可用数据库空间不足时，则会自动扩展存储。在以下因素适用时，Amazon RDS 会为启用了自动扩展的数据库实例启动存储修改：

- 可用空间小于或等于所分配的存储空间的 10%。
- 存储空间不足状态至少持续 5 分钟。
- 自上次存储修改以来，至少已过去 6 小时；或在实例上完成存储优化后，至少已过去 6 小时。

附加存储的增量采用下面的较大值：

- 10 GiB
- 10% 的当前分配的存储空间
- 根据过去一小时的 `FreeStorageSpace` 指标，在接下来的 7 小时内，预计存储空间增长将超过当前分配的存储空间大小。有关指标的更多信息，请参阅使用 Amazon CloudWatch 进行监控。

最大存储阈值是您为数据库实例自动扩展设置的限制。它具有以下限制：

- 您必须将最大存储阈值设置为比当前分配的存储空间多至少 10%。我们建议将其设置为至少多 26%，以避免收到有关存储大小接近最大存储阈值的事件通知。

  例如，如果您的数据库实例分配了 1000 GiB 的存储空间，则应将最大存储阈值设置为至少 1100 GiB。如果不这样设置，您会收到错误提示消息，例如 Invalid max storage size for *engine_name*（engine_name 的最大存储空间大小无效）。但是，我们建议您将最大存储阈值设置为至少 1260GiB，以避免出现事件通知。
- 对于使用预调配 IOPS（io1 或 io2 Block Express）存储的数据库实例，IOPS 与最大存储阈值（以 GiB 为单位）的比率必须在一定范围内。有关更多信息，请参阅 预置 IOPS SSD 存储。
- 对于启用了自动扩展的实例，您为其设置的最大存储阈值不得大于数据库引擎和数据库实例类的最大分配存储。

  例如，db.m5.xlarge 上的 SQL Server Standard Edition 为实例默认分配的存储为 20GiB（最小值），分配的最大存储为 16384GiB。自动扩展的默认最大存储阈值为 1000GiB。如果使用此默认值，则实例不会自动扩展到 1000GiB 以上。即使为实例分配的最大存储为 16384GiB，也是如此。

> **Note**
>
> 我们建议您根据使用模式和客户需求谨慎选择最大存储阈值。如果使用模式中存在任何偏差，则最大存储阈值可以防止在 Autoscaling 预测一个非常高的阈值时将存储扩展到一个意外的高值。自动扩展数据库实例后，无法减少其分配的存储空间。

主题

- 存储自动扩缩限制
- 为新的数据库实例启用存储自动扩展功能
- 更改数据库实例的存储自动扩展设置
- 手动纵向缩减或横向缩减数据库实例

- 为数据库实例禁用存储自动扩展功能

存储自动扩缩限制

以下限制适用于存储自动扩展：

- 如果存储增量超出最大存储阈值，则不会进行自动扩展。
- 自动扩展时，RDS 会预测后续自动扩展操作的存储大小。如果预计后续操作将超过最大存储阈值，则 RDS 将自动扩展到最大存储阈值。
- 自动缩放无法完全防止大量数据加载导致存储已满的情况。这是因为，在六 (6) 小时后或在实例上完成存储优化后 (以较长者为准)，您才能进行进一步的存储修改。

  如果执行大型数据加载，并且自动扩展不能提供足够的空间，则数据库可能会保持存储已满状态几小时。这会损害数据库。
- 如果您在 Amazon RDS 启动自动扩展操作的同时启动存储扩展操作，则您的存储修改优先。将取消自动扩展操作。
- 自动扩展无法减少分配的存储。您无法在已分配存储后减少数据库实例的存储量。
- 自动扩展功能不能与磁性存储一起使用。
- 自动扩展功能不能与以下可排序存储空间小于 6 TiB 的上一代实例类别一起使用：db.m3.large、db.m3.xlarge 和 db.m3.2xlarge。
- Amazon CloudTrail 不记录自动扩展操作。有关 CloudTrail 的更多信息，请参阅 监控 Amazon CloudTrail 中的 Amazon RDS API 调用。

虽然自动扩展可以帮助您动态增加 Amazon RDS 数据库实例上的存储空间，但您仍应将数据库实例的初始存储空间配置为适合典型工作负载的大小。

为新的数据库实例启用存储自动扩展功能

在创建新的 Amazon RDS 数据库实例时，可以选择是否启用存储自动扩展功能。您还可设置 Amazon RDS 可以为数据库实例分配的存储空间的上限。

> **Note**
>
> 在克隆已启用存储自动扩展的 Amazon RDS 数据库实例时，克隆的实例不会自动继承该设置。新的数据库实例具有与原始实例相同的分配的存储空间量。如果克隆的实例继续增加其存储要求，则可以再次为新实例启用存储自动扩展功能。

控制台

为新的数据库实例启用存储自动扩展功能

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在 Amazon RDS 控制台的右上角，选择要在其中创建数据库实例的Amazon区域。
3. 在导航窗格中，选择 Databases (数据库)。
4. 选择创建数据库。在选择引擎页面上，选择您的数据库引擎并指定数据库实例信息，如开始使用 Amazon RDS中所述。
5. 在存储自动扩展部分中，设置数据库实例的最大存储阈值值。
6. 指定其余的数据库实例信息，如开始使用 Amazon RDS中所述。

Amazon CLI

要为新的数据库实例启用存储自动扩展功能，请使用 Amazon CLI 命令 `create-db-instance`。设置以下参数：

- `--max-allocated-storage` – 打开存储自动扩展功能，并设置存储大小的上限（以 GiB 为单位）。

要验证 Amazon RDS 存储自动扩展功能是否适用于数据库实例，请使用 Amazon CLI `describe-valid-db-instance-modifications` 命令。要在创建实例前基于实例类进行检查，请使用 `describe-orderable-db-instance-options` 命令。检查返回值中的以下字段：

- `SupportsStorageAutoscaling` – 指示数据库实例或实例类是否支持存储自动扩展功能。

有关存储的更多信息，请参阅Amazon RDS 数据库实例存储。

RDS API

要为新的数据库实例启用存储自动扩展，请使用 Amazon RDS API 操作 `CreateDBInstance`。设置以下参数：

- `MaxAllocatedStorage` – 打开 Amazon RDS 存储自动扩展功能，并设置存储大小的上限（以 GiB 为单位）。

要验证 Amazon RDS 存储自动扩展功能是否适用于数据库实例，请对现有实例使用 Amazon RDS API `DescribeValidDbInstanceModifications` 操作或在创建实例前使用 `DescribeOrderableDBInstanceOptions` 操作。检查返回值中的以下字段：

- `SupportsStorageAutoscaling` – 指示数据库实例是否支持存储自动扩展功能。

有关存储的更多信息，请参阅Amazon RDS 数据库实例存储。

更改数据库实例的存储自动扩展设置

您可以为现有 Amazon RDS 数据库实例打开存储自动扩展功能。您还可以更改 Amazon RDS 为数据库实例分配的存储空间的上限。

控制台

更改数据库实例的存储自动扩展设置

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。
3. 选择要修改的数据库实例，然后选择修改。将显示修改数据库实例页面。
4. 在 Autoscaling (自动扩展) 部分中，更改存储限制。有关更多信息，请参阅"修改 Amazon RDS 数据库实例"。
5. 当所有更改都达到您的要求时，选择 Continue (继续) 并查看修改。
6. 在确认页面上，检查您的更改。如果更改正确无误，请选择修改数据库实例以保存更改。如果更改错误，请选择 Back (返回) 以编辑您的更改，或者选择 Cancel (取消) 以取消更改。

   立即更改存储自动扩展限制。此设置忽略立即应用设置。

Amazon CLI

要更改数据库实例的存储自动扩展设置，请使用 Amazon CLI 命令 `modify-db-instance`。设置以下参数：

- `--max-allocated-storage` – 设置存储大小的上限（以 GiB 为单位）。如果该值大于 `--allocated-storage` 参数，则打开存储自动扩展功能。如果该值与 `--allocated-storage` 参数相等，则关闭存储自动扩展功能。

要验证 Amazon RDS 存储自动扩展功能是否适用于数据库实例，请使用 Amazon CLI describe-valid-db-instance-modifications 命令。要在创建实例前基于实例类进行检查，请使用 describe-orderable-db-instance-options 命令。检查返回值中的以下字段：

- SupportsStorageAutoscaling – 指示数据库实例是否支持存储自动扩展功能。

有关存储的更多信息，请参阅Amazon RDS 数据库实例存储。

RDS API

要更改数据库实例的存储自动扩展设置，请使用 Amazon RDS API 操作 ModifyDBInstance。设置以下参数：

- MaxAllocatedStorage – 设置存储大小的上限（以 GiB 为单位）。

要验证 Amazon RDS 存储自动扩展功能是否适用于数据库实例，请对现有实例使用 Amazon RDS API DescribeValidDbInstanceModifications 操作或在创建实例前使用 DescribeOrderableDBInstanceOptions 操作。检查返回值中的以下字段：

- SupportsStorageAutoscaling – 指示数据库实例是否支持存储自动扩展功能。

有关存储的更多信息，请参阅Amazon RDS 数据库实例存储。

手动纵向缩减或横向缩减数据库实例

Amazon RDS 提供存储自动扩展功能，用来满足不断增长的需求。但是，在纵向缩减和横向缩减方面存在限制：

- RDS 存储：虽然 RDS 支持在需求增加时自动纵向扩展存储，但它不会自动缩减。
- 只读副本：RDS 不支持自动横向扩展（添加）或横向缩减（删除）只读副本。您必须根据您的负载要求手动添加或移除只读副本。

要纵向缩减 RDS 资源，请执行以下手动操作：

- 对于存储，您无法使用 modify-db-instance 命令手动减少为数据库实例分配的存储空间。而是选择以下方法之一：
  - 如果您的数据库引擎支持蓝绿部署，请使用蓝绿部署。创建存储容量较小的绿色数据库，然后将绿色数据库提升为蓝色数据库。有关更多信息，请参阅 修改存储和性能设置。

- 使用分配的较低存储空间创建新的数据库实例，手动将数据从当前数据库迁移到新创建的数据库实例，然后切换数据库端点。
- 对于只读副本，请通过 RDS 控制台或 Amazon CLI 手动删除任何未使用的副本。

为数据库实例禁用存储自动扩展功能

如果您不再需要 Amazon RDS 自动增加 Amazon RDS 数据库实例的存储空间，则可以关闭存储自动扩展功能。执行此操作后，您仍可以手动增加数据库实例的存储空间量。

控制台

为数据库实例打开存储自动扩展功能

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。
3. 选择要修改的数据库实例，然后选择修改。将显示修改数据库实例页面。
4. 清除存储自动扩展部分中的启用存储自动扩展复选框。有关更多信息，请参阅“修改 Amazon RDS 数据库实例”。
5. 当所有更改都达到您的要求时，选择 Continue (继续) 并查看修改。
6. 在确认页面上，检查您的更改。如果更改正确无误，请选择修改数据库实例以保存更改。如果更改错误，请选择 Back (返回) 以编辑您的更改，或者选择 Cancel (取消) 以取消更改。

立即更改存储自动扩展限制。此设置忽略立即应用设置。

Amazon CLI

要为数据库实例关闭存储自动扩展功能，请使用 Amazon CLI 命令 `modify-db-instance` 和以下参数：

- `--max-allocated-storage` – 指定一个与 `--allocated-storage` 设置相等的值可进一步防止指定数据库实例的 Amazon RDS 存储自动扩展。

有关存储的更多信息，请参阅Amazon RDS 数据库实例存储。

RDS API

要为数据库实例关闭存储自动扩展，请使用 Amazon RDS API 操作 `ModifyDBInstance`。设置以下参数：

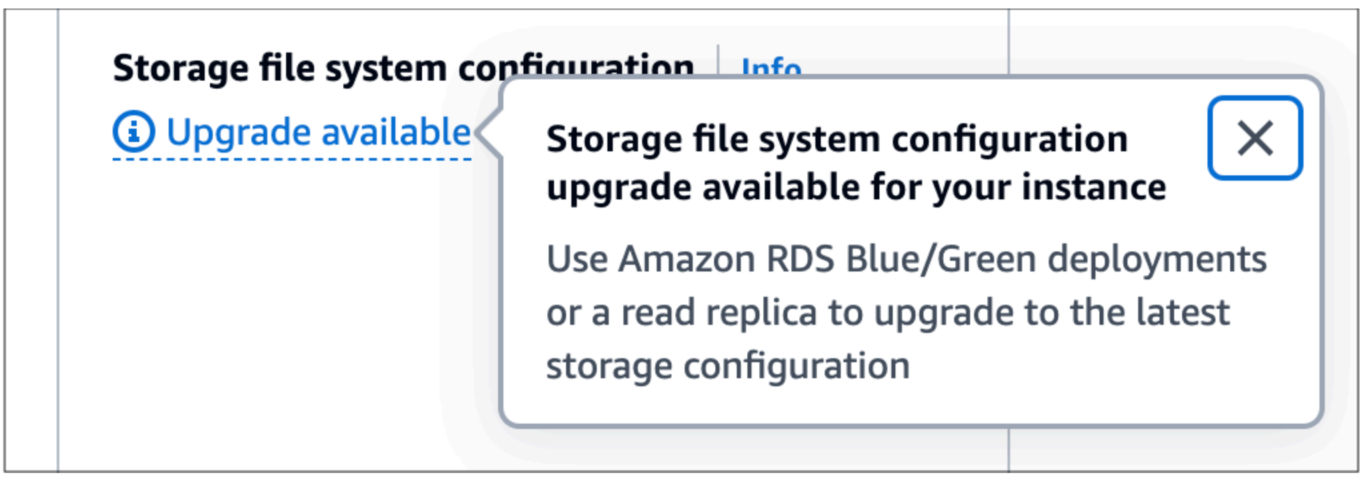
- `MaxAllocatedStorage` – 指定一个与 `AllocatedStorage` 设置相等的值可进一步防止指定数据库实例的 Amazon RDS 存储自动扩展。

有关存储的更多信息，请参阅Amazon RDS 数据库实例存储。

升级数据库实例的存储文件系统

大多数 RDS 数据库实例为 RDS for MariaDB、MySQL 和 PostgreSQL 数据库提供的最大存储大小为 64 TiB。但是，一些较旧的 32 位文件系统的存储容量较低。要确定数据库实例的存储容量，请使用 describe-valid-db-instance-modifications Amazon CLI 命令。

RDS 检查存储系统是否具有 16 TiB 的存储大小、2 TiB 的文件大小限制或未优化的写入。如果数据库实例满足这些条件，RDS 会提醒您文件系统配置符合升级条件。您可以在数据库实例详细信息页面的存储面板上检查数据库实例的升级资格。



如果数据库实例符合文件系统升级的条件，请使用以下任一方法：

- 创建蓝绿部署并指定升级存储文件系统配置。

  此选项将绿色环境中的文件系统升级到首选配置。然后，您可以切换蓝绿部署，以将绿色环境升级为新的生产环境。有关详细说明，请参阅the section called “创建蓝绿部署”。

- 创建数据库实例只读副本并指定升级存储文件系统配置。

此选项将只读副本的文件系统升级到首选配置。您可以将只读副本升级为独立的数据库实例。有关详细说明，请参阅the section called “创建只读副本”。

在存储升级期间，数据库引擎不可用。升级存储配置是一项 I/O 密集型操作，会导致只读副本和蓝绿部署的创建时间更长。当满足以下两个条件时，存储升级过程会更快：

- 源数据库实例使用预调配 IOPS 固态硬盘（io1 或 io2 Block Express）存储。
- 您为绿色环境或只读副本预调配的实例大小为 4xlarge 或更大。

涉及通用 SSD（gp2）存储的存储升级可能会耗尽您的 I/O 积分余额，从而导致升级时间更长。有关更多信息，请参阅 the section called “数据库实例存储”。

在存储升级期间，当同时满足以下两个条件时，RDS 会将为绿色实例或只读副本分配的存储大小增加 10%：

- 源数据库实例的存储消耗量大于或等于已分配存储大小的 90%。
- 存储自动扩缩已启用。

当新的存储大小大于或等于为实例设置的最大分配存储时，RDS 会关闭自动扩缩。如果在存储升级开始之前禁用了存储自动扩缩，则存储大小在升级过程中不会增加。

修改预调配 IOPS SSD 存储的设置

可利用 Amazon RDS 控制台、Amazon CLI 或 Amazon RDS API 修改使用预置 IOPS SSD 存储的数据库实例的设置。指定存储类型、分配的存储和需要的预调配 IOPS 的数量。该范围取决于数据库引擎和实例类型。

尽管您可以减少为实例预调配的 IOPS 数量，但无法减小存储大小。

在大多数情况下，扩展存储无需停机，也不会降低服务器的性能。在修改数据库实例的存储 IOPS 后，数据库实例的状态为 storage-optimization。

> **Note**
>
> 存储优化可能耗时数小时。六（6）小时后或在实例上完成存储优化后，您才能进行进一步的存储修改。

有关每个数据库引擎可用的分配存储空间和预调配 IOPS 范围的信息，请参阅 [预置 IOPS SSD 存储]。

控制台

更改数据库实例的预置 IOPS 设置

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。

   要筛选数据库实例的列表，对于 Filter databases (筛选数据库)，请输入 Amazon RDS 用于筛选结果的文本字符串。将仅显示其名称包含该字符串的数据库实例。
3. 选择具有要修改的预置 IOPS 的数据库实例。
4. 选择修改。
5. 在修改数据库实例页面上，为存储类型选择预调配 IOPS SSD（io1）或预调配 IOPS SSD（io2）。
6. 对于 Provisioned IOPS（预调配 IOPS），输入一个值。

   如果为 Allocated storage（分配的存储）或 Provisioned IOPS（预调配 IOPS）指定的值超过其他参数支持的上限，则会显示一条警告消息。此消息提供了其他参数所需的值范围。
7. 选择 Continue (继续)。
8. 在 Scheduling of modifications（修改计划）部分中选择 Apply immediately（立即应用），以立即将存储更改应用于数据库实例。或选择 Apply during the next scheduled maintenance window (在下一个计划的维护时段内应用) 以在下一个维护时段内应用更改。
9. 检查将更改的参数，然后选择修改数据库实例以完成修改。

   分配的存储空间或预置 IOPS 的新值会显示在状态列中。

Amazon CLI

要更改数据库实例的预置 IOPS 设置，请使用 Amazon CLI 命令 `modify-db-instance`。设置以下参数：

- `--storage-type` – 对于预调配 IOPS，设置为 `io1` 或 `io2`。
- `--allocated-storage` – 要为数据库实例分配的存储量（以 GiB 为单位）。
- `--iops` – 数据库实例预置 IOPS 的新数量，以每秒的输入/输出操作数量表示。

- `--apply-immediately` – 使用 `--apply-immediately` 可立即应用更改。使用 `--no-apply-immediately`（默认值）可在下一个维护时段内应用更改。

RDS API

要更改数据库实例的预调配 IOPS 设置，请使用 Amazon RDS API 操作 [`ModifyDBInstance`]。设置以下参数：

- `StorageType` – 对于预调配 IOPS，设置为 `io1` 或 `io2`。
- `AllocatedStorage` – 要为数据库实例分配的存储量（以 GiB 为单位）。
- `Iops` – 数据库实例的新 IOPS 速率，以每秒的输入/输出操作数量表示。
- `ApplyImmediately` – 将此选项设置为 `True` 可立即应用更改。将此选项设置为 `False`（默认值）可在下一个维护时段内应用更改。

I/O 密集型存储修改

Amazon RDS 数据库实例使用 Amazon Elastic Block Store（EBS）卷进行数据库和日志存储。根据所请求的存储量，RDS（RDS for SQL Server 除外）自动跨多个 Amazon EBS 卷进行条带化以增强性能。具有 SSD 存储类型的 RDS 数据库实例由 RAID 0 配置中的一个或四个条带化 Amazon EBS 卷提供支持。根据设计，RDS 数据库实例的存储修改操作对正在进行的数据库操作的影响极小。

在大多数情况下，存储扩展修改完全卸载到 Amazon EBS 层，并且对于数据库是透明的。该过程通常在几分钟内完成。但是，一些较旧的 RDS 存储卷需要不同的过程来修改大小、预调配 IOPS 或存储类型。这涉及使用潜在 I/O 密集型操作制作数据的完整副本。

如果以下任何因素适用，则存储修改将使用 I/O 密集型操作：

- 源存储类型是磁性的。磁性存储不支持弹性卷修改。
- RDS 数据库实例不在单卷或四卷的 Amazon EBS 布局上。您可以使用增强监控指标查看 RDS 数据库实例上正在使用的 Amazon EBS 卷的数量。有关更多信息，请参阅 [在 RDS 控制台中查看操作系统指标]。
- 修改请求的目标大小对于 RDS for MariaDB、MySQL 和 PostgreSQL 实例将分配的存储空间增加到 400GiB 以上，而对于 RDS for Oracle 则增加到 200GiB 以上。当存储自动扩展操作将数据库实例的分配存储大小增加到超过这些阈值时，其效果相同。

如果您的存储修改涉及 I/O 密集型操作，则会消耗 I/O 资源并增加数据库实例的负载。如果使用涉及通用 SSD（gp2）存储的 I/O 密集型操作进行存储修改，则可能会耗尽 I/O 积分余额，从而导致转换时间更长。

作为最佳做法，我们建议将这些存储修改请求安排在高峰时段以外，以帮助减少完成存储修改操作所需的时间。或者，您可以创建数据库实例的只读副本，并在只读副本上执行存储修改。然后，将只读副本提升为主数据库实例。有关更多信息，请参阅 使用数据库实例只读副本。

有关更多信息，请参阅当我尝试增加分配的存储时，为什么 Amazon RDS 数据库实例卡在修改状态？

修改通用型 SSD（gp3）存储的设置

可以利用 Amazon RDS 控制台、Amazon CLI 或 Amazon RDS API 修改使用通用型 SSD（gp3）存储的数据库实例的设置。指定所需的存储类型、分配的存储、预调配 IOPS 数量以及存储吞吐量。

尽管您可以减少数据库实例的预调配 IOPS 数量和存储吞吐量，但无法减小存储大小。

在大多数情况下，扩展存储无需任何停机。在修改数据库实例的存储 IOPS 后，数据库实例的状态为 storage-optimization。在存储优化期间，您可以预期延迟会增加，但仍处于个位数毫秒范围内。在修改存储后，数据库实例可以完全正常运行。

> **Note**
>
> 在实例上完成存储优化的六 (6) 小时后，您才能进行进一步的存储修改。

有关每个数据库引擎可用的已分配存储、预调配 IOPS 和存储吞吐量范围的信息，请参阅 gp3 存储（推荐）。

控制台

更改数据库实例的存储性能设置

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。

   要筛选数据库实例的列表，对于 Filter databases (筛选数据库)，请输入 Amazon RDS 用于筛选结果的文本字符串。将仅显示其名称包含该字符串的数据库实例。
3. 选择具有要修改的 gp3 存储的数据库实例。

4. 选择修改。
5. 在修改数据库实例页面上，对于存储类型选择通用型 SSD（gp3），然后执行以下操作：

   a. 对于 Provisioned IOPS（预调配 IOPS），选择一个值。

      如果为 Allocated storage（分配的存储）或 Provisioned IOPS（预调配 IOPS）指定的值超过其他参数支持的上限，则会显示一条警告消息。此消息提供了其他参数所需的值范围。

   b. 对于 Storage throughput（存储吞吐量），选择一个值。

      如果为 Provisioned IOPS（预调配 IOPS）或 Storage throughput（存储吞吐量）指定的值超过其他参数支持的上限，则会显示一条警告消息。此消息提供了其他参数所需的值范围。
6. 选择 Continue (继续)。
7. 在 Scheduling of modifications（修改计划）部分中选择 Apply immediately（立即应用），以立即将存储更改应用于数据库实例。或选择 Apply during the next scheduled maintenance window (在下一个计划的维护时段内应用) 以在下一个维护时段内应用更改。
8. 检查将更改的参数，然后选择修改数据库实例以完成修改。

   预调配 IOPS 的新值会显示在 Status（状态）列中。

Amazon CLI

要更改数据库实例的存储性能设置，请使用 Amazon CLI 命令 [`modify-db-instance`]。设置以下参数：

- `--storage-type` – 对于通用型 SSD（gp3），设置为 `gp3`。
- `--allocated-storage` – 要为数据库实例分配的存储量（以 GiB 为单位）。
- `--iops` – 数据库实例预置 IOPS 的新数量，以每秒的输入/输出操作数量表示。
- `--storage-throughput` – 数据库实例新的存储吞吐量，以 MiBps 表示。
- `--apply-immediately` – 使用 `--apply-immediately` 可立即应用更改。使用 `--no-apply-immediately`（默认值）可在下一个维护时段内应用更改。

RDS API

要更改数据库实例的存储性能设置，请使用 Amazon RDS API 操作 [`ModifyDBInstance`]。设置以下参数：

- `StorageType` – 对于通用型 SSD（gp3），设置为 `gp3`。

- `AllocatedStorage` – 要为数据库实例分配的存储量（以 GiB 为单位）。
- `Iops` – 数据库实例的新 IOPS 速率，以每秒的输入/输出操作数量表示。
- `StorageThroughput` – 数据库实例新的存储吞吐量，以 MiBps 表示。
- `ApplyImmediately` – 将此选项设置为 `True` 可立即应用更改。将此选项设置为 `False`（默认值）可在下一个维护时段内应用更改。

使用专用日志卷（DLV）

您可以将专用日志卷（DLV）用于使用预调配 IOPS（PIOPS）存储的数据库实例。DLV 将 PostgreSQL 数据库事务日志、MySQL/MariaDB 重做日志和二进制日志移动到与包含数据库表的卷不同的存储卷中。DLV 可以提高事务写入日志的效率和一致性。DLV 非常适合分配了较大存储空间、每秒 I/O（IOPS）要求较高或工作负载对延迟敏感的数据库。

DLV 支持 PIOPS 存储（io1 和 io2 Block Express），创建它们时的固定大小为 1,024 GiB，预调配 IOPS 为 3000。

对于以下版本，Amazon RDS 在所有 Amazon Web Services 区域都支持 DLV：

- MariaDB 10.6.7 及更高的 10 版本
- MySQL 8.0.28 及更高的 8.0 版本，MySQL 8.4.3 及更高的 8.4 版本
- PostgreSQL 13.10 及更高的 13 版本、14.7 及更高的 14 版本，以及 15.2 及更高的 15 版本

RDS 对多可用区部署支持 DLV。修改或创建多可用区实例时，会同时为主实例和辅助实例创建 DLV。

RDS 对于只读副本支持 DLV。如果主数据库实例启用了 DLV，则启用 DLV 后创建的所有只读副本也将具有 DLV。除非经过明确修改，否则在切换到 DLV 之前创建的任何只读副本都不会启用 DLV。我们建议在启用 DLV 之前连接到主实例的所有只读副本也要手动修改为具有 DLV。

> **Note**
>
> 对于 5 TiB 或更大的数据库配置，建议使用 DLV。

有关 DLV 的好处的更多信息，请参阅以下博客文章：

- [使用 Amazon RDS 专用日志卷提高数据库性能]
- [使用专用日志卷对 Amazon RDS for PostgreSQL 进行基准测试]

- Percona 文档中的 Maximizing performance of Amazon RDS for MySQL with dedicated log volumes

有关每个数据库引擎可用的已分配存储、预调配 IOPS 和存储吞吐量范围的信息，请参阅 预置 IOPS SSD 存储。

主题

启用和禁用 DLV 时的注意事项

启用和禁用 DLV 可能很耗时并导致停机。该过程包括在启用时将所有事务日志或重做和二进制日志（具体取决于数据库引擎）复制到新卷，或者在禁用时复制回原始存储。此操作的持续时间受多个因素影响：

- 事务日志数量：
  - 具有更多事务的较大数据库会生成更多日志，从而增加复制所需的时间。
  - 如果复制插槽处于非活动状态或复制正处于滞后状态，则事务日志可能会在主数据库实例上累积，从而增加复制所需的时间。确保复制是最新的，并移除所有不必要的插槽。
- 存储配置：
  - 数据库实例 EBS 带宽 – 带宽越高，则支持的数据传输速度越快。
  - 预调配 IOPS 数 – 每秒进行读写操作的次数（IOPS）越多，复制过程就越快。
- 数据库活动 - 配置期间的高水平数据库活动可能会减慢此过程。

为了显著缩短停机时间，建议您将此过程计划和安排在活动较少的期间或维护时段进行。

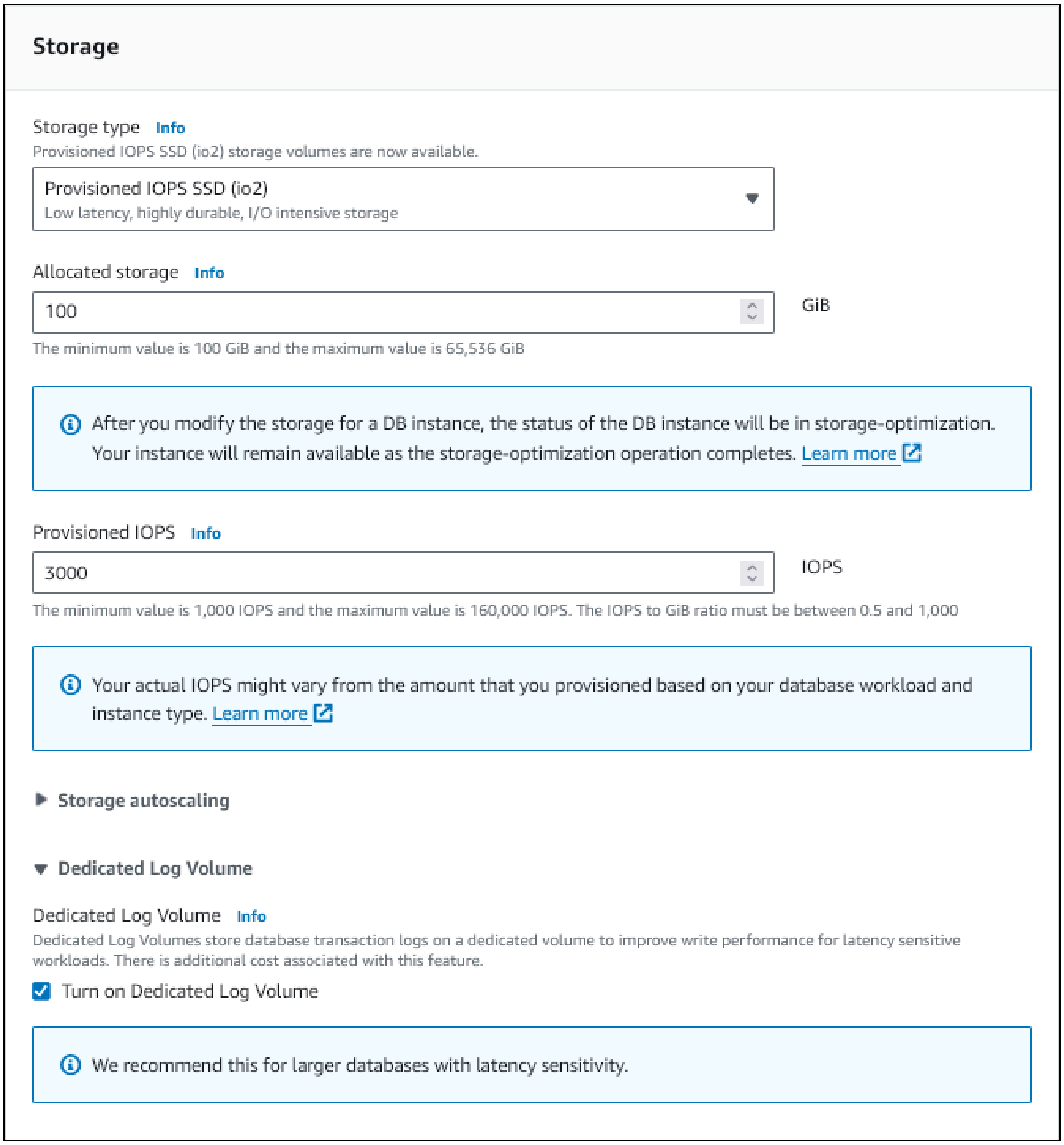
创建数据库实例时启用 DLV

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 创建启用了 DLV 的数据库实例。

控制台

在新数据库实例上启用 DLV

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 选择创建数据库。
3. 在创建数据库实例页面上，选择支持 DLV 的数据库引擎。
4. 对于存储：
    a. 选择预调配 IOPS SSD（io1）或预调配 IOPS SSD（io2）。
    b. 输入所需的分配的存储空间和预调配 IOPS。
    c. 展开专用日志卷，然后选择开启专用日志卷。



5. 根据需要选择其它设置。
6. 选择创建数据库。

创建数据库后，专用日志卷的值将显示在数据库详细信息页面的配置选项卡上。

CLI

要在创建数据库实例时使用预调配 IOPS 存储启用 DLV，请使用 Amazon CLI 命令 create-db-instance。设置以下参数：

- `--dedicated-log-volume` – 启用专用日志卷。

- `--storage-type` – 对于预调配 IOPS，设置为 `io1` 或 `io2`。
- `--allocated-storage` – 要为数据库实例分配的存储量（以 GiB 为单位）。
- `--iops` – 数据库实例的预调配 IOPS 的数量，以每秒的 I/O 操作数量表示。

RDS API

要在创建数据库实例时使用预调配 IOPS 存储启用 DLV，请使用 Amazon RDS API 操作 CreateDBInstance。设置以下参数：

- `DedicatedLogVolume` – 要启用专用日志卷，请设置为 `true`。
- `StorageType` – 对于预调配 IOPS，设置为 `io1` 或 `io2`。
- `AllocatedStorage` – 要为数据库实例分配的存储量（以 GiB 为单位）。
- `Iops` – 数据库实例的 IOPS 速率，以每秒的 I/O 操作数量表示。

在现有数据库实例上启用 DLV

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 修改数据库实例来启用 DLV。

在修改数据库实例的 DLV 设置后，必须重启数据库实例。

控制台

在现有数据库实例上启用 DLV

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。

   要筛选数据库实例的列表，对于 Filter databases (筛选数据库)，请输入 Amazon RDS 用于筛选结果的文本字符串。将仅显示其名称包含该字符串的数据库实例。
3. 选择具有要修改的预调配 IOPS 存储的数据库实例。
4. 选择 Modify(修改)。
5. 在修改数据库实例页面上：

   - 对于存储，展开专用日志卷，然后选择开启专用日志卷。
6. 选择继续。

7. 选择立即应用可立即将更改应用到数据库实例。或选择 Apply during the next scheduled maintenance window (在下一个计划的维护时段内应用) 以在下一个维护时段内应用更改。
8. 检查将更改的参数，然后选择修改数据库实例以完成修改。

专用日志卷的新值将显示在数据库详细信息页面的配置选项卡上。

CLI

要使用预调配 IOPS 存储在现有数据库实例上启用或禁用 DLV，请使用 Amazon CLI 命令 `modify-db-instance`。设置以下参数：

- `--dedicated-log-volume` – 启用专用日志卷。

  使用 `--no-dedicated-log-volume`（默认值）将禁用专用日志卷。
- `--apply-immediately` – 使用 `--apply-immediately` 可立即应用更改。

  使用 `--no-apply-immediately`（默认值）可在下一个维护时段内应用更改。

RDS API

要使用预调配 IOPS 存储在现有数据库实例上启用或禁用 DLV，请使用 Amazon RDS API 操作 `ModifyDBInstance`。设置以下参数：

- `DedicatedLogVolume` – 将此选项设置为 `true` 可启用专用日志卷。

  将此选项设置为 `false` 可禁用专用日志卷。这是默认值。
- `ApplyImmediately` – 将此选项设置为 `True` 可立即应用更改。

  将此选项设置为 `False`（默认值）可在下一个维护时段内应用更改。

监控 DLV 存储

可以使用 CloudWatch 中的 `FreeStorageSpaceLogVolume` 指标来监控 DLV 存储使用情况。

可以对 RDS for PostgreSQL 使用以下查询来查找事务日志占用的大小：

```
SELECT pg_size_pretty(COALESCE(sum(size), 0)) AS total_wal_generated_size
FROM pg_catalog.pg_ls_waldir();
```

如果 DLV 的存储空间用完，则数据库实例将进入 `storage-full` 状态，从而导致停机。

删除数据库实例

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 删除数据库实例。如果要删除 Aurora 数据库集群中的数据库实例，请参阅删除 Aurora 数据库集和数据库实例。

主题

删除数据库实例的先决条件

在尝试删除数据库实例之前，确保删除保护已关闭。原定设置情况下，对于使用控制台创建的数据库实例，删除保护处于开启状态。

如果您的数据库实例开启了删除保护，则可以通过修改实例设置将该功能关闭。在数据库详细信息页面中选择修改或调用 modify-db-instance 命令。此操作不会导致中断。有关更多信息，请参阅 数据库实例的设置。

删除数据库实例时的注意事项

删除数据库实例会影响实例的可恢复性、备份可用性和只读副本状态。请考虑以下问题：

- 您可以选择是否创建最终数据库快照。您有以下选项：
  - 如果拍摄最终快照，则可以使用此快照来还原删除的数据库实例。RDS 会保留最终快照和您之前拍摄的任何手动快照。如果数据库实例不是 `Available` 状态，则无法为其创建最终数据库快照。有关更多信息，请参阅 查看Amazon RDS 数据库实例状态。
  - 如果您不创建最终快照，删除实例的速度会更快。缺点是没有稍后可以还原的最终快照。如果您决定还原已删除的数据库实例，请保留自动备份或使用较早的手动快照将数据库实例还原到该较早快照的时间点。
- 您可以选择是否保留自动备份。您有以下选项：
  - 如果您保留自动备份，RDS 会在您删除数据库实例时对该实例有效的保留期内保留这些备份。您可以使用自动备份将数据库实例还原到保留期内的某个时间，但不能还原到保留期后的某个时间。无论是否创建最终数据库快照，保留期都有效。要删除保留的自动备份，请参阅删除保留的自动备份。

  - 保留的自动备份和手动快照会一直产生账单费用，直到将其删除为止。有关更多信息，请参阅 保留成本。
  - 如果您不保留自动备份，RDS 会删除与您的数据库实例位于同一 Amazon Web Services 区域中的自动备份。您无法恢复这些备份。如果您的自动备份已复制到另一个 Amazon Web Services 区域，即使您不选择保留自动备份，RDS 也会保留这些备份。有关更多信息，请参阅 将自动备份复制到其他 Amazon Web Services 区域。

> **Note**
>
> 通常，如果您创建最终数据库快照，则不需要保留自动备份。

- 当您删除数据库实例时，RDS 不会删除手动数据库快照。有关更多信息，请参阅 为 Amazon RDS 的单可用区数据库实例创建数据库快照。
- 如果您想删除所有 RDS 资源，请注意以下资源会产生账单费用：
  - 数据库实例
  - 数据库快照
  - 数据库集群

  如果您购买了预留实例，则将根据您在购买实例时达成的合同对这些实例进行计费。有关更多信息，请参阅 Amazon RDS 的预留数据库实例。您可以使用 Amazon Cost Explorer 获取所有 Amazon 资源的账单信息。有关更多信息，请参阅使用 Amazon Cost Explorer 分析成本。
- 如果您在同一 Amazon Web Services 区域中删除具有只读副本的数据库实例，则每个只读副本都会自动提升为独立的数据库实例。有关更多信息，请参阅 将只读副本提升为独立的数据库实例。如果您的数据库实例在不同 Amazon Web Services 区域中具有只读副本，请参阅跨区域复制注意事项以了解与删除跨区域只读副本的源数据库实例相关的信息。
- 当数据库实例的状态为 `deleting` 时，其 CA 证书值不会显示在 RDS 控制台中或 Amazon CLI 命令或 RDS API 操作的输出中。有关 CA 证书的更多信息，请参阅 使用 SSL/TLS 加密与数据库实例或集群的连接。
- 删除数据库实例所需的时间会有所不同，具体取决于备份保留期（也即，要删除的备份数）、删除的数据量以及是否拍摄最终快照。

删除数据库实例

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 删除数据库实例。您必须执行以下操作：

- 提供数据库实例的名称
- 启用或禁用制作实例的最终数据库快照的选项
- 启用或禁用保留自动备份的选项

> **Note**
>
> 当数据库实例启用了删除保护时，您无法删除该实例。有关更多信息，请参阅 删除数据库实例的先决条件。

控制台

删除数据库实例

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)，然后选择要删除的数据库实例。
3. 对于 Actions，选择 Delete。
4. 要为数据库实例创建最终数据库快照，请选择是否创建最终快照?。
5. 如果选择创建最终快照，请输入最终快照名称。
6. 要保留自动备份，请选择 Retain automated backups (保留自动备份)。
7. 在框中输入 **delete me**。
8. 选择删除。

Amazon CLI

要查找您的账户中数据库实例的实例 ID，请调用 describe-db-instances 命令：

```
aws rds describe-db-instances --query 'DBInstances[*].[DBInstanceIdentifier]' --output
 text
```

要使用 Amazon CLI 删除数据库实例，请结合以下选项调用 delete-db-instance 命令：

- `--db-instance-identifier`
- `--final-db-snapshot-identifier` 或者 `--skip-final-snapshot`

Example 有最终快照，但无保留的自动备份

对于 Linux、macOS 或 Unix：

```
aws rds delete-db-instance \
    --db-instance-identifier mydbinstance \
    --final-db-snapshot-identifier mydbinstancefinalsnapshot \
    --delete-automated-backups
```

对于 Windows：

```
aws rds delete-db-instance ^
    --db-instance-identifier mydbinstance ^
    --final-db-snapshot-identifier mydbinstancefinalsnapshot ^
    --delete-automated-backups
```

Example 有保留的自动备份，但无最终快照

对于 Linux、macOS 或 Unix：

```
aws rds delete-db-instance \
    --db-instance-identifier mydbinstance \
    --skip-final-snapshot \
    --no-delete-automated-backups
```

对于 Windows：

```
aws rds delete-db-instance ^
    --db-instance-identifier mydbinstance ^
    --skip-final-snapshot ^
    --no-delete-automated-backups
```

RDS API

要使用 Amazon RDS API 删除数据库实例，请使用以下参数调用 [`DeleteDBInstance`] 操作：

- `DBInstanceIdentifier`
- `FinalDBSnapshotIdentifier` 或者 `SkipFinalSnapshot`

教程：从开发到生产，管理 MySQL 数据库实例环境

简介

管理 Amazon RDS 数据库实例时，常见的场景是监督该实例从初始开发到生产部署的生命周期。本教程为处理关键任务提供了指导，来确保数据库以上佳状态运行，并适应不断变化的运营需求。此外，还概述了在开发环境和生产环境之间同步所做更改的选项，来确保一致性和可靠性。

通过完成这些步骤，您将学习：

- 如何对 MySQL 数据库实例执行特定任务，例如添加和更新 Amazon RDS 标签、扩展存储、创建只读副本和删除资源。
- 如何将更新从生产环境同步到开发环境，来进行全面测试和验证。

要完成本教程，请执行以下任务：

1. 创建 MySQL 数据库实例。
2. 添加 Amazon RDS 标签，来将数据库实例归类为开发环境。
3. 增加数据库实例的存储容量，来适应增加的工作负载。
4. 创建只读副本，来增强数据库实例的弹性和可用性。
5. 更新 Amazon RDS 标签，来将数据库实例归类为生产环境。
6. 删除您不再需要的数据库实例，这样它们就不会产生额外费用。
7. 后续步骤：将开发实例与生产同步以实现跨环境的一致性

先决条件

在开始之前，请完成以下各节中的步骤：

- 注册 Amazon Web Services 账户
- 保护 IAM 用户

添加标签以将数据库实例归类为开发环境

要将数据库实例归类为开发环境，请向您创建的实例添加一个 Amazon RDS 标签。Amazon RDS 标签是您定义的键值对，且与您的 RDS 实例关联。为 Amazon 资源添加标签有助于区分开发和生产 Amazon 资源。有关 Amazon RDS 标签的更多信息，请参阅为 Amazon RDS 资源添加标签。

1. 在 Amazon RDS 控制台中，选择数据库。
2. 选择要添加标签的数据库实例。
3. 在详细信息部分中，滚动到标签部分。
4. 选择管理标签，然后选择添加新标签。
5. 为标签键和值输入一个值。例如，可以使用值为 dev 的标签键环境，来指定数据库实例是开发环境的一部分。
6. 选择添加新标签和保存更改。

数据库实例现已标记为开发环境。这样，就可以更轻松地识别数据库实例，并管理与此资源关联的成本。

增加数据库实例的存储容量来适应不断增长的数据需求

接下来，修改 MySQL 数据库实例的存储容量，来容纳更多数据。最初，数据库实例的存储容量设置为满足应用程序的即时需求。但是，随着数据量增长，可能需要调整存储设置，来确保数据库的持续性能和稳定性。此过程涉及增加数据库实例的已分配存储空间。有关修改数据库实例的存储容量的更多信息，请参阅使用 Amazon RDS 数据库实例的存储。

1. 在 Amazon RDS 控制台中，选择数据库。
2. 选择要修改的数据库实例。
3. 选择修改。
4. 在“存储”中，增加分配的存储空间。修改后的存储值必须大于当前值。
5. 选择继续。
6. 在修改计划中，可以选择立即应用，以便立即将存储更改应用于数据库实例；也可以选择在下一个计划的维护时段内应用，以便在下一个维护时段内应用更改。
7. 根据需要设置完毕后，选择修改数据库实例。

数据库实例的存储容量现已增加。这使该实例能够有效地处理更大的数据量，并确保随着应用程序数据需求的增长而持续保持性能和稳定性。

创建只读副本来增强数据库实例的弹性和可用性

创建 MySQL 数据库实例的只读副本。只读副本可增强数据库实例的弹性和可用性。要减少主数据库实例上的读取流量，请创建数据库实例的只读副本。这会将查询路由到只读副本，有助于分散负载并提高数据库的整体性能。有关数据库实例只读副本的更多信息，请参阅使用数据库实例只读副本。

在 MySQL 数据库实例可充当复制源之前，必须在源数据库实例上启用自动备份。这可以通过将备份保留期设置为非 0 值来实现。有关 MySQL 只读副本的更多信息，请参阅使用 MySQL 只读副本。

1. 在 Amazon RDS 控制台中，选择数据库。
2. 选择要用作只读副本的源的数据库实例。
3. 在“操作”中，选择创建只读副本。
4. 对于数据库实例标识符，输入只读副本以全小写字母表示的名称。
5. 选择您的实例配置。建议您为只读副本使用与源数据库实例相同或更大的数据库实例类和存储类型。
6. 对于 Amazon Web Services 区域，指定只读副本的目标区域。
7. 保留默认设置，或根据需要对其进行修改。
8. 选择创建只读副本。

在 RDS 控制台中的数据库页面上，只读副本显示在源数据库实例的下方。它在“角色”列中显示副本。

更新标签以将数据库实例归类为生产环境

当数据库实例准备好从开发阶段转到生产阶段时，更新其标签来反映其转换非常重要。要使数据库实例与运营和监控策略保持一致，请更新初始标签，来指示该数据库实例现在已成为生产环境的一部分。这样可以确保数据库的更好可见性和管理。

1. 在 Amazon RDS 控制台中，选择数据库。
2. 选择要更新的数据库实例。
3. 在详细信息部分中，滚动到标签部分。
4. 选择管理标签。
5. 移除表示开发环境的初始标签。
6. 选择添加新标签。
7. 为标签键和值输入新值。例如，可以使用值为 prod 的标签键环境，来指定数据库实例是生产环境的一部分。
8. 选择添加新标签和保存更改。

数据库实例上的标签已更新，来表示数据库转换到生产环境。

当不再需要数据库实例时，将其删除以免产生额外费用

在本教程结束之前，解决资源管理问题至关重要。如果您有任何不再需要的资源，应该继续删除它们，以免产生额外的成本并优化云环境。

1. 在 Amazon RDS 控制台中，选择数据库。
2. 选择要删除的数据库实例
3. 在“操作”中，选择删除。删除数据库实例将永久删除该实例及其所有内容和相关资源。
4. 确认删除数据库实例，然后选择删除。

或者，如果您选择维护数据库实例以备将来使用，则可以继续将其作为生产环境的一部分进行管理。这包括维护同步的开发环境来促进全面的测试和验证。有关更多信息，请参阅 后续步骤：将开发实例与生产同步以实现跨环境的一致性。

后续步骤：将开发实例与生产同步以实现跨环境的一致性

创建开发环境

要管理生产环境，务必维护同步的开发环境来进行全面的测试和验证。要创建新的开发环境，请先创建当前生产数据库实例的数据库快照。数据库快照通过创建存储卷快照来捕获整个数据库实例。有关如何在 Amazon RDS 控制台上创建数据库快照的说明，请参阅为 Amazon RDS 的单可用区数据库实例创建数据库快照。

创建生产环境的数据库快照后，通过还原数据库快照，为开发环境创建一个新的数据库实例。还原的数据库实例自动与默认数据库参数和选项组关联。但是，您可以通过在还原期间指定自定义参数组和选项组来应用它们。有关还原数据库快照的说明，请参阅教程：从数据库快照还原 Amazon RDS 数据库实例。

最后，通过更新新数据库实例的 Amazon RDS 标签，将其指定为新的开发环境。有关更新 Amazon RDS 标签来反映此项更改的指南，请参阅上一节更新标签以将数据库实例归类为生产环境。

现在，您有了新的开发环境，它镜像生产环境的数据库配置。

将开发环境与生产环境同步

一旦建立了新的开发环境，就务必使其与生产环境中发生的任何更改保持同步。这可以确保开发环境准确地反映当前的生产状态，这对于进行有效的测试、验证和故障排除至关重要。Amazon RDS 提供了多种不同的方法，来使开发环境与生产环境保持同步。有关这些选项的更多信息，请参阅 Orchestrating database refreshes for Amazon RDS and Amazon Aurora。

同步开发环境和生产环境的主要方法之一是创建和还原数据库快照。数据库快照可让您创建开发环境，该环境将反映在创建快照时生产环境的数据库配置。有关数据库快照的更多信息，请参阅管理手动备份。有关还原数据库实例的更多信息，请参阅还原到数据库实例。

数据库快照对于以下用例特别有价值。

- 开发环境的初始设置：数据库快照对于创建用于测试的初始开发环境很有用，因为它提供了一致的基准，可镜像在创建快照时生产环境的确切状态。
- 高流量应用程序：在持续运行极为重要的生产环境中，对快照使用多可用区部署可避免主数据库上发生 I/O 暂停，从而确保不间断的性能和高可用性。
- 在不同的 RDS 账户间共享数据：可以在不同的 Amazon Web Services 账户间共享数据库快照，从而促进账户或区域之间的数据传输。这对于协作项目或出于各种目的而需要共享数据的场景很有用。有关更多信息，请参阅 共享 Amazon RDS 的数据库快照。

在本教程中，您探讨了在数据库实例的整个生命周期中对其进行管理的基本任务。您学习了如何创建数据库实例、添加和更新 Amazon RDS 标签、扩展存储空间以及创建只读副本。您还学习了在这些基本操作的基础上进行构建以及有效管理生产环境的方法。这包括建立用于测试的开发环境，并将其与生产环境同步来保持一致性。这些任务有助于维护充满弹性和可扩展的数据库基础设施，确保 Amazon RDS 环境高效运行。

配置和管理 Amazon RDS 的多可用区部署

多可用区部署可以有一个备用或两个备用数据库实例。当部署有一个备用数据库实例时，称为多可用区数据库实例部署。多可用区数据库实例部署有一个备用数据库实例，可提供故障转移支持，但不提供读取流量。当部署有两个备用数据库实例时，称为多可用区数据库集群部署。多可用区数据库集群部署具有备用数据库实例，可提供故障转移支持，还可以提供读取流量。

您可以使用 Amazon Web Services Management Console 来确定多可用区部署是多可用区数据库实例部署还是多可用区数据库集群部署。在导航窗格中，选择 Databases（数据库），然后选择数据库标识符。

- 多可用区数据库实例部署具有以下特征：
  - 数据库实例只有一行。
  - Role（角色）的值为 Instance（实例）或 Primary（主要）。
  - Multi-AZ（多可用区）的值为 Yes（是）。
- 多可用区数据库集群部署具有以下特征：
  - 有一个集群级行，其下有三个数据库实例行。
  - 对于群集级行，Role（角色）的值为 Multi-AZ DB cluster（多可用区数据库集群）。
  - 对于每个实例级行，Role（角色）的值为 Writer instance（写入器实例）或 Reader instance（读取器实例）。
  - 对于每个实例级行，Multi-AZ（多可用区）的值为 3 Zones（3 个区域）。

主题

- Amazon RDS 的多可用区数据库实例部署
- Amazon RDS 的多可用区数据库集群部署

此外，以下主题同时适用于数据库实例和多可用区数据库集群。

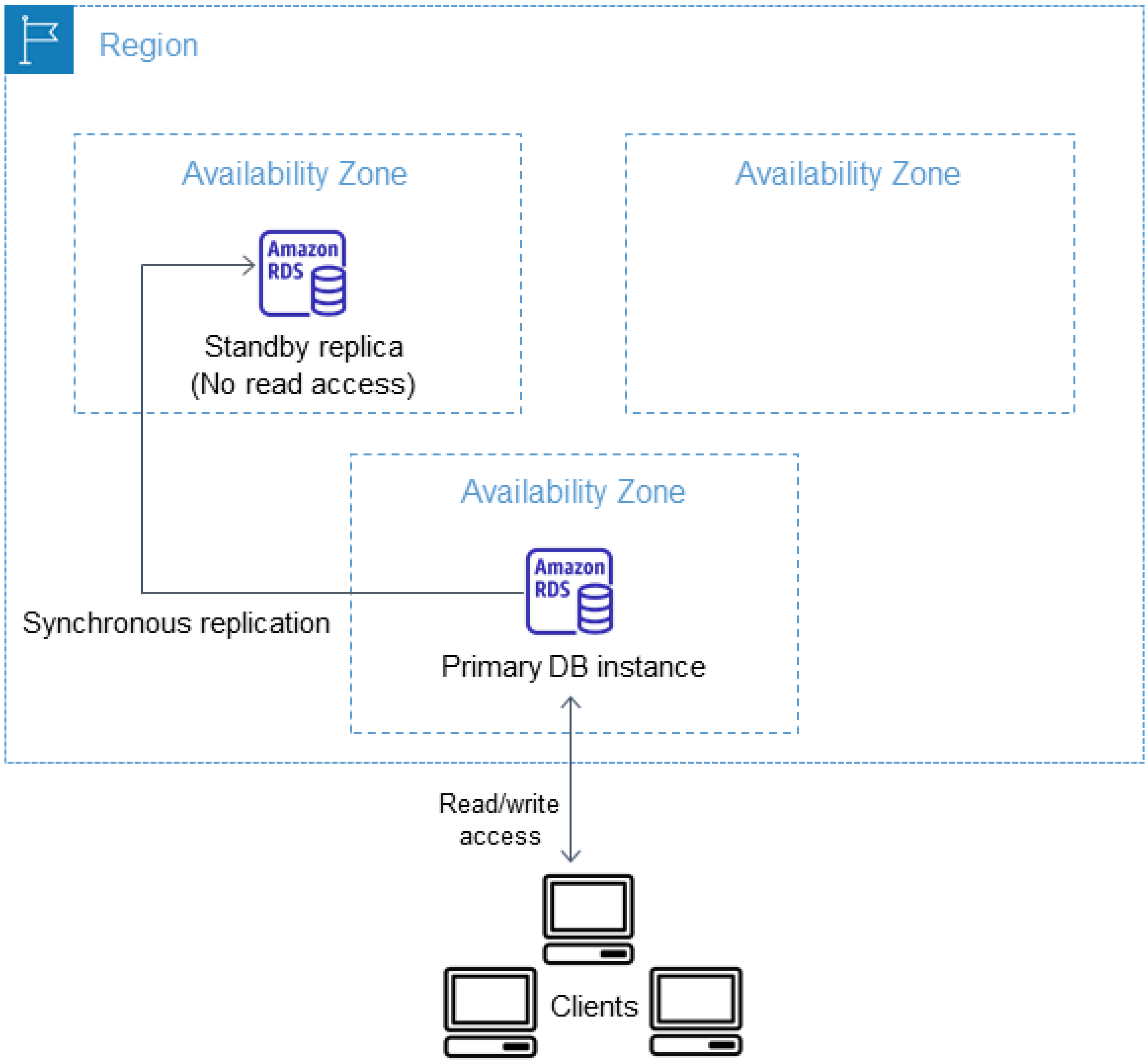
- the section called “为 RDS 资源添加标签”
- the section called “Amazon RDS 中的 ARN”
- the section called “使用存储”
- the section called “维护数据库实例”
- the section called “升级引擎版本”

Amazon RDS 的多可用区数据库实例部署

Amazon RDS 使用带有一个备用数据库实例的多可用区部署为数据库实例提供高可用性和故障转移支持。这种类型的部署称为多可用区数据库实例部署。Amazon RDS 使用几种不同的技术来提供此故障转移支持。用于 MariaDB、MySQL、Oracle、PostgreSQL 和 RDS Custom for SQL Server 数据库实例的多可用区部署使用 Amazon 失效转移技术。Microsoft SQL Server 数据库实例使用 SQL Server 数据库镜像 (DBM) 或 Always On 可用性组 (AG)。有关多可用区的 SQL Server 版本支持的信息，请参阅Amazon RDS for Microsoft SQL Server 多可用区部署。有关将 RDS Custom for SQL Server 用于多可用区的信息，请参阅管理 RDS Custom for SQL Server 的多可用区部署。

在多可用区数据库实例部署中，Amazon RDS 会自动在不同可用区中配置和维护一个同步备用副本。主数据库实例将跨可用区同步复制到备用副本，以提供数据冗余并在系统备份期间将延迟峰值降至最小。在计划内的系统维护期间，运行具有高可用性的数据库实例可以提高可用性。它还可以帮助您保护数据库，以防数据库实例发生故障和可用区中断。有关可用区的更多信息，请参阅区域、可用区和 Local Zones。

> **Note**
>
> 高可用性选项不是只读场景的扩缩解决方案。您不能使用备用副本来提供读取流量。要提供只读流量，请使用多可用区数据库集群或只读副本。有关多可用区数据库集群的更多信息，请参阅 Amazon RDS 的多可用区数据库集群部署。有关只读副本的更多信息，请参阅 使用数据库实例只读副本。



使用 RDS 控制台，您只需在创建数据库实例时指定多可用区，即可创建多可用区数据库部署。您可以使用控制台修改数据库实例并指定多可用区选项，从而将现有数据库实例转换为多可用区数据库实例部署。您还可以使用 Amazon CLI 或 Amazon RDS API 指定多可用区数据库实例部署。使用 create-db-instance 或 modify-db-instance CLI 命令，或者 CreateDBInstance 或 ModifyDBInstance API 操作。

RDS 控制台显示备用副本的可用区（称为辅助可用区）。您还可以使用 describe-db-instances CLI 命令或 DescribeDBInstances API 操作来查找辅助可用区。

与单可用区部署相比，使用多可用区数据库部署的数据库实例会增加写入和提交延迟。这可能是因为发生了同步数据复制。尽管 Amazon 设计用于在可用区之间提供低延迟网络连接，但如果您的部署故障转移到备用副本，延迟可能会发生变化。对于生产工作负载，我们建议您使用预置 IOPS（每秒输入/输出操作数）以获得快速、一致的性能。有关数据库实例类的更多信息，请参阅 数据库实例类。

将 Amazon RDS 的数据库实例转换为多可用区部署

将数据库实例修改为多可用区部署，可通过在另一个可用区中添加一个备用实例来提高可用性。该过程涉及的停机时间极少，需要围绕存储和性能影响进行仔细的规划。此项更改可增强容错能力并减少出现故障时的恢复时间，使其成为高可用性环境的理想之选。

如果您有一个采用单可用区部署的数据库实例，并且要将它修改为多可用区数据库实例部署，则 Amazon RDS 执行以下操作：

1. 拍摄主数据库实例的 Amazon Elastic Block Store（EBS）卷的快照。
2. 从快照中为备用副本创建新卷。这些卷在后台初始化，并在数据完全初始化后达到最大卷性能。
3. 开启主副本卷与备用副本卷之间的同步块级复制。

> **Important**
>
> 在单可用区转换到多可用区期间，从快照创建备用数据库实例可以避免停机时间，但可能会影响性能，特别是对于写入敏感型工作负载。同步复制会增加 I/O 延迟，从而影响数据库性能。作为最佳做法，请避免将生产数据库实例转换为多可用区数据库实例。
> 而是创建只读副本，对该副本启用备份，将其转换为多可用区，将数据加载到其卷中，然后将它提升为主数据库实例。有关更多信息，请参阅 使用数据库实例只读副本。

有两种方法可将数据库实例修改为多可用区数据库实例部署：

主题

使用 RDS 控制台转换为多可用区数据库实例部署

您可以使用 RDS 控制台将数据库实例转换为多可用区数据库实例部署。

您只能使用控制台来完成转换。要使用 Amazon CLI 或 RDS API，请按照将数据库实例修改为多可用区数据库部署中的说明操作。

使用 RDS 控制台转换为多可用区数据库实例部署

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择数据库，然后选择要修改的数据库实例。
3. 从 Actions（操作）中，选择 Convert to Multi-AZ deployment（转换为多可用区部署）。
4. 在确认页面上，选择 Apply immediately（立即应用）以立即应用更改。选择此选项不会导致停机，但可能会对性能产生影响。或者，您可以选择在下一个维护时段内应用更新。有关更多信息，请参阅 使用计划修改设置。
5. 选择 Convert to Multi-AZ（转换为多可用区）。

将数据库实例修改为多可用区数据库部署

可以通过以下方式将数据库实例修改为多可用区数据库实例部署：

- 使用 RDS 控制台，修改数据库实例，并将 Multi-AZ deployment（多可用区部署）设置为 Yes（是）。
- 使用 Amazon CLI，调用 modify-db-instance 命令，然后设置 `--multi-az` 选项。
- 使用 RDS API，调用 ModifyDBInstance 操作并将 `MultiAZ` 参数设置为 `true`。

有关修改数据库实例的信息，请参阅修改 Amazon RDS 数据库实例。在修改完成后，Amazon RDS 会触发事件 (RDS-EVENT-0025)，表示该过程已完成。您可以监控 Amazon RDS 事件。有关事件的更多信息，请参阅使用 Amazon RDS 事件通知。

对 Amazon RDS 的多可用区数据库实例执行失效转移

如果由于基础设施缺陷而导致多可用区数据库实例发生计划内或计划外中断，Amazon RDS 会自动切换到另一个可用区中的备用副本。

完成故障转移所用的时间取决于在主数据库实例变为不可用时的数据库活动和其他条件。故障转移时间通常为 60–120 秒。不过，事务较多或时间较长的恢复过程可能延长故障转移时间。完成故障转移后，RDS 控制台还需要一段时间才能反映新的可用区。

> **Note**
>
> 在重启多可用区数据库实例时，可以手动强制执行失效转移。有关更多信息，请参阅 重启中的数据库实例。

Amazon RDS 会自动处理故障转移，因此，您可以尽快恢复数据库操作而无需管理干预。如果出现下表中描述的任一情况，主数据库实例会自动切换到备用副本：您可以在事件日志中查看这些故障转移原因。

故障转移原因	描述
RDS 数据库实例所在的操作系统正在脱机操作中安装补丁。	在操作系统补丁或安全更新的维护时段内触发了故障切换。 有关更多信息，请参阅“维护数据库实例”。
RDS 多可用区实例的主要主机运行状况不佳。	多可用区数据库实例部署检测到受损的主数据库实例并进行故障转移。
由于网络连接断开，无法访问 RDS 多可用区实例的主机。	RDS 监控检测到主数据库实例的网络可达性故障并触发了故障转移。
客户修改了 RDS 实例。	RDS 数据库实例修改触发了故障转移。 有关更多信息，请参阅“修改 Amazon RDS 数据库实例”。
RDS 多可用区主实例正忙且无响应。	主数据库实例没有响应。建议您执行以下操作： • 检查事件和 CloudWatch Logs 是否存在 CPU、内存或交换空间使用过量的情况。有关更多信息，请参阅“使用 Amazon RDS 事件通知”和“创建对 Amazon RDS 事件触发的规则”。 •

故障转移原因	描述
	评估您的工作负载，以确定是否使用了适当的数据库实例类。有关更多信息，请参阅“数据库实例类”。 • 使用增强监控对实时操作系统指标进行监控。有关更多信息，请参阅“使用增强监控来监控操作系统指标”。 • 使用性能详情来帮助分析影响数据库实例性能的任何问题。有关更多信息，请参阅“在 Amazon RDS 上使用性能详情监控数据库负载”。 有关这些建议的更多信息，请参阅 Amazon RDS 的监控工具 和 Amazon RDS 的最佳实践。
RDS 多可用区实例的主要主机所在的存储卷出现故障。	多可用区数据库实例部署在主数据库实例上检测到存储问题并进行故障转移。
用户请求数据库实例的故障转移。	您重新启动了数据库实例，并选择了通过故障转移重启。 有关更多信息，请参阅 重启中的数据库实例。

要确定多可用区数据库实例是否发生故障转移，您可以执行以下操作：

- 将数据库事件订阅设置为在故障转移启动时向您发送电子邮件或 SMS 通知。有关事件的更多信息，请参阅 使用 Amazon RDS 事件通知。
- 使用 RDS 控制台或 API 操作查看数据库事件。
- 使用 RDS 控制台和 API 操作查看多可用区数据库实例部署的当前状态。

有关如何响应故障转移、缩短恢复时间以及 Amazon RDS 的其他最佳实践的信息，请参阅 Amazon RDS 的最佳实践。

设置 DNS 名称查找的 JVM TTL

故障转移机制自动更改数据库实例的域名系统 (DNS) 记录，使其指向备用数据库实例。因此，您需要重新建立与数据库实例之间的所有现有连接。在 Java 虚拟机 (JVM) 环境中，由于 Java DNS 缓存机制的工作原理，您可能需要重新配置 JVM 设置。

JVM 缓存 DNS 名称查找。当 JVM 将主机名解析为 IP 地址时，它会在指定时间段内 (称为存活时间 (TTL)) 缓存 IP 地址。

由于 Amazon 资源使用偶尔变更的 DNS 名称条目，因此建议您为 JVM 配置的 TTL 值不超过 60 秒。这样做可确保在资源的 IP 地址发生更改时，您的应用程序可以通过重新查询 DNS 来接收和使用资源的新 IP 地址。

对于一些 Java 配置，将设置 JVM 默认 TTL，以便在重新启动 JVM 之前绝不刷新 DNS 条目。因此，如果 Amazon 资源的 IP 地址在应用程序仍在运行时发生更改，则在您手动重新启动 JVM 并刷新缓存的 IP 信息之前，将无法使用该资源。在此情况下，设置 JVM 的 TTL，以便定期刷新其缓存的 IP 信息是极为重要的。

您可以通过检索 `networkaddress.cache.ttl` 属性值来获取 JVM 的原定设置 TTL：

```
String ttl = java.security.Security.getProperty("networkaddress.cache.ttl");
```

> **Note**
>
> 默认 TTL 是变化的，具体取决于 JVM 的版本以及是否安装安全管理器。许多 JVM 提供的默认 TTL 小于 60 秒。如果您使用此类 JVM 并且未使用安全管理器，则您可以忽略本主题的剩余内容。有关 Oracle 中安全管理器的更多信息，请参阅 Oracle 文档中的安全管理器。

要修改 JVM 的 TTL，请设置 `networkaddress.cache.ttl` 属性值。根据您的需求，使用下列方法之一：

- 要为使用 JVM 的所有应用程序全局设置属性值，请在 `networkaddress.cache.ttl` 文件中设置 `$JAVA_HOME/jre/lib/security/java.security`。

  ```
  networkaddress.cache.ttl=60
  ```

- 要仅在本地为应用程序设置属性，请在建立任何网络连接之前，在应用程序的初始化代码中设置 `networkaddress.cache.ttl`。

```
java.security.Security.setProperty("networkaddress.cache.ttl" , "60");
```

Amazon RDS 的多可用区数据库集群部署

多可用区数据库集群部署是 Amazon RDS 的半同步、高可用性部署模式，具有两个可读副本数据库实例。多可用区数据库集群在同一个 Amazon Web Services 区域 的三个独立可用区中有一个写入器数据库实例和两个读取器数据库实例。与多可用区数据库实例部署相比，多可用区数据库集群可提供高可用性、增加读取工作负载容量以及更低的写入延迟。

您可以使用 将数据导入 Amazon RDS for MySQL 数据库实例并减少停机时间 中的说明将本地数据库中的数据导入到多可用区数据库集群。

您可以为多可用区数据库集群购买预留数据库实例。有关更多信息，请参阅 多可用区数据库集群的预留数据库实例。

功能可用性和支持因每个数据库引擎的特定版本以及 Amazon Web Services 区域而异。有关使用多可用区数据库集群的 Amazon RDS 的版本和区域可用性的更多信息，请参阅支持 Amazon RDS 中多可用区数据库集群的区域和数据库引擎。

主题

- 设置从 Amazon RDS 的多可用区数据库集群进行外部复制
- 删除 Amazon RDS 的多可用区数据库集群
- Amazon RDS 的多可用区数据库集群的限制

> **Important**
>
> 多可用区数据库集群与 Aurora 数据库集群不同。有关使用 Aurora 数据库集群的更多信息，请参阅《Amazon Aurora 用户指南》。

多可用区数据库集群的实例类可用性

以下数据库实例类支持多可用区数据库集群部署：`db.m5d`、`db.m6gd`、`db.m6id`、`db.m6idn`、`db.r5d`、`db.r6gd`、`db.x2iedn`、`db.r6id`、`db.r6i` 和 `db.c6gd`。

> **Note**
>
> c6gd 实例类是仅有的支持 `medium` 实例大小的实例类。

有关数据库实例类的更多信息，请参阅 the section called “数据库实例类”。

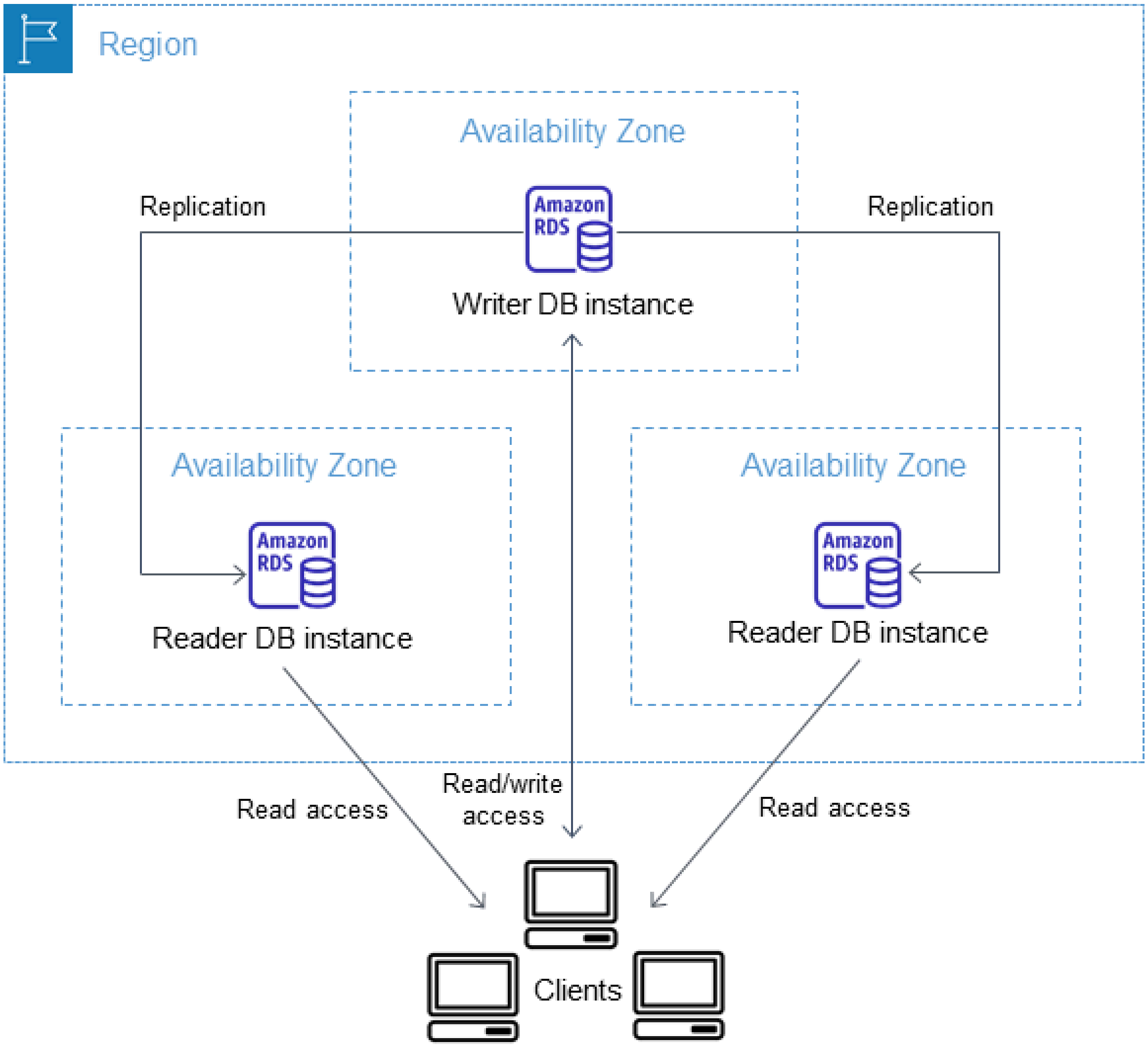
多可用区数据库集群架构

使用多可用区数据库集群，Amazon RDS 使用数据库引擎的本机复制功能将数据从写入器数据库实例复制到两个读取器数据库实例。当对写入器数据库实例进行更改时，它会发送给每个读取器数据库实例。

多可用区数据库集群部署使用半同步复制，这需要来自至少一个读取器数据库实例的确认才能提交更改。它不需要确认事件已在所有副本上完全执行和提交。

读取器数据库实例可充当自动故障转移目标，还可提供读取流量以提高应用程序读取吞吐量。如果写入器数据库实例发生中断，RDS 将管理到其中一个读取器数据库实例的故障转移。RDS 会根据具有最近更改记录的读取器数据库实例来执行此操作。

下图显示了一个多可用区数据库集群。



与多可用区数据库实例部署相比，多可用区数据库集群的写入延迟通常较低。其还允许在读取器数据库实例上运行只读工作负载。RDS 控制台显示了写入器数据库实例的可用区和读取器数据库实例的可用区。您也可以使用 describe-db-clusters CLI 命令或 DescribeDBClusters API 操作来查找此信息。

> **Important**
>
> 为防止 RDS for MySQL 多可用区数据库集群出现复制错误，我们强烈建议所有表都具有主键。

多可用区数据库集群的参数组

在多可用区数据库集群中，数据库集群参数组就像是引擎配置值的容器，这些值可应用于多可用区数据库集群中的每个数据库实例。

在多可用区数据库集群中，数据库参数组设置为数据库引擎和数据库引擎版本的默认数据库参数组。数据库集群参数组中的设置用于集群中的所有数据库实例。

有关参数组的信息，请参阅 the section called “数据库集群参数组”。

RDS 代理与多可用区数据库集群

您可以使用 Amazon RDS 代理为您的多可用区数据库集群创建代理。通过使用 RDS 代理，您的应用程序可以池化和共享数据库连接，以提高其扩展能力。每个代理都执行连接多路复用，也称为连接重用。对于多路复用，RDS 代理使用一个底层数据库连接对事务执行所有操作。RDS 代理还可以将多可用区数据库集群的次要版本升级的停机时间缩短到一秒或更短。有关 RDS 代理优点的更多信息，请参阅 Amazon RDS 代理。

要为多可用区数据库集群设置代理，请在创建集群时选择创建 RDS 代理。有关创建和管理 RDS 代理端点的说明，请参阅 the section called “使用 RDS Proxy 终端节点”。

副本滞后和多可用区数据库集群

副本滞后是写入器数据库实例上的最新事务与读取器数据库实例上的最新应用事务之间的时间差异。Amazon CloudWatch 指标 `ReplicaLag` 表示这种时间差。有关 CloudWatch 指标的更多信息，请参阅使用 Amazon CloudWatch 监控 Amazon RDS 指标。

尽管多可用区数据库集群允许高写入性能，但基于引擎的复制的性质仍可能导致副本滞后。由于任何故障转移都必须先解决副本滞后问题，然后才能提升新的写入器数据库实例，因此要考虑监控和管理此副本滞后。

对于 RDS for MySQL 多可用区数据库集群，故障转移时间取决于两个剩余读取器数据库实例的副本滞后。两个读取器数据库实例必须先应用未应用的事务，然后才能将其中一个升级为新写入器数据库实例。

对于 RDS for PostgreSQL 多可用区数据库集群，故障转移时间取决于两个剩余读取器数据库实例的最低副本滞后。具有最低副本滞后的读取器数据库实例必须先应用未应用的事务，然后才能将其升级为新写入器数据库实例。

有关演示副本滞后超过设置时间量时如何创建 CloudWatch 告警的教程，请参阅 教程：为 Amazon RDS 的多可用区数据库集群副本滞后创建 Amazon CloudWatch 警报。

副本滞后的常见原因

一般来说，写入工作负载过高，导致读取器数据库实例无法有效应用事务时，就会出现副本滞后。各种工作负载都可能会产生临时或持续副本滞后。以下是一些常见示例：

- 写入器数据库实例的高写入并发或大量批处理更新，导致读取器数据库实例上的应用过程落后。
- 在一个或多个读取器数据库实例上使用资源的繁重的读取工作负载。运行速度慢或查询数量多都可能会影响应用进程，继而导致副本滞后。
- 由于数据库必须保留提交顺序，因此修改大量数据或 DDL 语句的事务有时会暂时加剧副本滞后问题。

缓解副本滞后问题

对于适用于 RDS for MySQL 和 RDS for PostgreSQL 的多可用区数据库集群，您可以通过减少写入器数据库实例的负载来缓解副本滞后问题。您还可以使用流量控制来减少副本滞后。流量控制通过限制写入器数据库实例上的写入操作来发挥作用，从而确保副本滞后时间不会继续无限延长。通过在事务结束时添加延迟来实现写入限制，以此降低写入器数据库实例的写入吞吐量。流量控制不能保证会消除滞后，但可以帮助缩短许多工作负载的总体滞后时间。以下各节提供有关将流量控制与 RDS for MySQL 和 RDS for PostgreSQL 结合使用的信息。

使用 RDS for MySQL 的流量控制来缓解副本滞后

当您使用 RDS for MySQL 多可用区数据库集群时，默认情况下，使用动态参数 `rpl_semi_sync_master_target_apply_lag` 启用流量控制。此参数指定副本滞后的上限。当副本滞后接近这一配置的限制时，流量控制会限制写入器数据库实例上的写入事务，以尝试将副本滞后控制在指定值以下。在某些情况下，副本滞后可能会超过指定的限制。默认情况下，该参数设置为 120 秒。要禁用流量控制，请将该参数设置为其最大值 86400 秒（一天）。

要查看流量控制注入的当前延迟，请通过运行以下查询来显示参数 `Rpl_semi_sync_master_flow_control_current_delay`。

```
SHOW GLOBAL STATUS like '%flow_control%';
```

您的输出应类似于以下内容。

```
+-------------------------------------------------+-------+
| Variable_name                                   | Value |
+-------------------------------------------------+-------+
| Rpl_semi_sync_master_flow_control_current_delay | 2010  |
+-------------------------------------------------+-------+
1 row in set (0.00 sec)
```

> **Note**
>
> 延迟以微秒为单位显示。

当您为 RDS for MySQL 多可用区数据库集群启用 Performance Insights 时，您可以监控与指示查询通过流量控制延迟的 SQL 语句对应的等待事件。当流量控制引入延迟时，您可以查看与 Performance Insights 控制面板上的 SQL 语句对应的等待事件 `/wait/synch/cond/semisync/semi_sync_flow_control_delay_cond`。要查看这些指标，请确保性能架构已启用。有关 Performance Insights 的信息，请参阅 在 Amazon RDS 上使用性能详情监控数据库负载。

使用 RDS for PostgreSQL 的流量控制来缓解副本滞后

使用 RDS for PostgreSQL 多可用区数据库集群时，流量控制作为扩展部署。其为数据库集群中的所有数据库实例开启后台工件。默认情况下，读取器数据库实例上的后台工件会将当前副本滞后情况告知写入器数据库实例上的后台工件信。如果读取器数据库实例的滞后时间超过两分钟，则写入器数据库实例上的后台工件会在事务结束时添加延迟。若要控制滞后阈值，请使用参数 `flow_control.target_standby_apply_lag`。

流量控制限制 PostgreSQL 进程时，`pg_stat_activity` 和 Performance Insights 中的 `Extension` 等待事件会表明该情况。函数 `get_flow_control_stats` 显示当前所添加延迟时长的详细信息。

流量控制可以给大多数短时、高量并发联机事务处理 (OLTP) 工作负载带来益处。如果滞后由长时间运行的事务（如分批操作）造成，流量控制便无法提供特别大的助益。

从 `shared_preload_libraries` 中移除扩展程序，然后重启数据库实例，即可关闭流量控制。

多可用区数据库集群快照

Amazon RDS 会在所配置的备份时段内创建并保存多可用区数据库集群的自动备份。RDS 创建数据库集群的存储卷快照，同时备份整个集群而不仅仅是各个实例。

也可以手动备份多可用区数据库集群。对于非常长期的备份，请考虑将快照数据导出到 Amazon S3。有关更多信息，请参阅 the section called “创建多可用区数据库集群的快照”。

您可以将多可用区数据库集群还原到特定时间点，从而创建新的多可用区数据库集群。有关说明，请参阅 the section called “将多可用区数据库集群还原到指定时间”。

或者，可以将多可用区数据库集群快照还原到单可用区部署或多可用区数据库实例部署。有关说明，请参阅 the section called “从多可用区数据库集群快照还原到数据库实例”。

创建 Amazon RDS 的多可用区数据库集群

多可用区数据库集群在三个独立可用区中有一个写入器数据库实例和两个读取器数据库实例。与多可用区部署相比，多可用区数据库集群可提供高可用性、增加读取工作负载容量以及更低的延迟。有关多可用区数据库集群的更多信息，请参阅 Amazon RDS 的多可用区数据库集群部署。

> **Note**
> 仅 MySQL 和 PostgreSQL 数据库引擎支持多可用区数据库集群。

数据库集群先决条件

> **Important**
> 在创建多可用区数据库集群之前，必须先完成设置 Amazon RDS 环境中的任务。

以下是创建多可用区数据库集群之前需要完成的先决条件。

主题

- 为数据库集群配置网络
- 其他先决条件

为数据库集群配置网络

您只能在虚拟私有云（VPC）中基于 Amazon VPC 服务创建多可用区数据库集群。它必须位于至少有三个可用区的 Amazon Web Services 区域 中。为数据库集群选择的数据库子网组必须包含至少三个可用区。此配置可确保数据库集群中的每个数据库实例都位于不同的可用区。

要在同一 VPC 中的新数据库集群和 Amazon EC2 实例之间设置连接，可以在创建数据库集群时执行此操作。要从同一 VPC 中的 EC2 实例以外的资源连接到数据库集群，请手动配置网络连接。

主题

- 配置与 EC2 实例的自动网络连接
- 手动配置网络

配置与 EC2 实例的自动网络连接

创建多可用区数据库集群时，可以使用 Amazon Web Services Management Console 在 EC2 实例和新数据库集群之间设置连接。当您这样做时，RDS 会自动配置您的 VPC 和网络设置。数据库集群在与 EC2 实例相同的 VPC 中创建，以便 EC2 实例可以访问数据库集群。

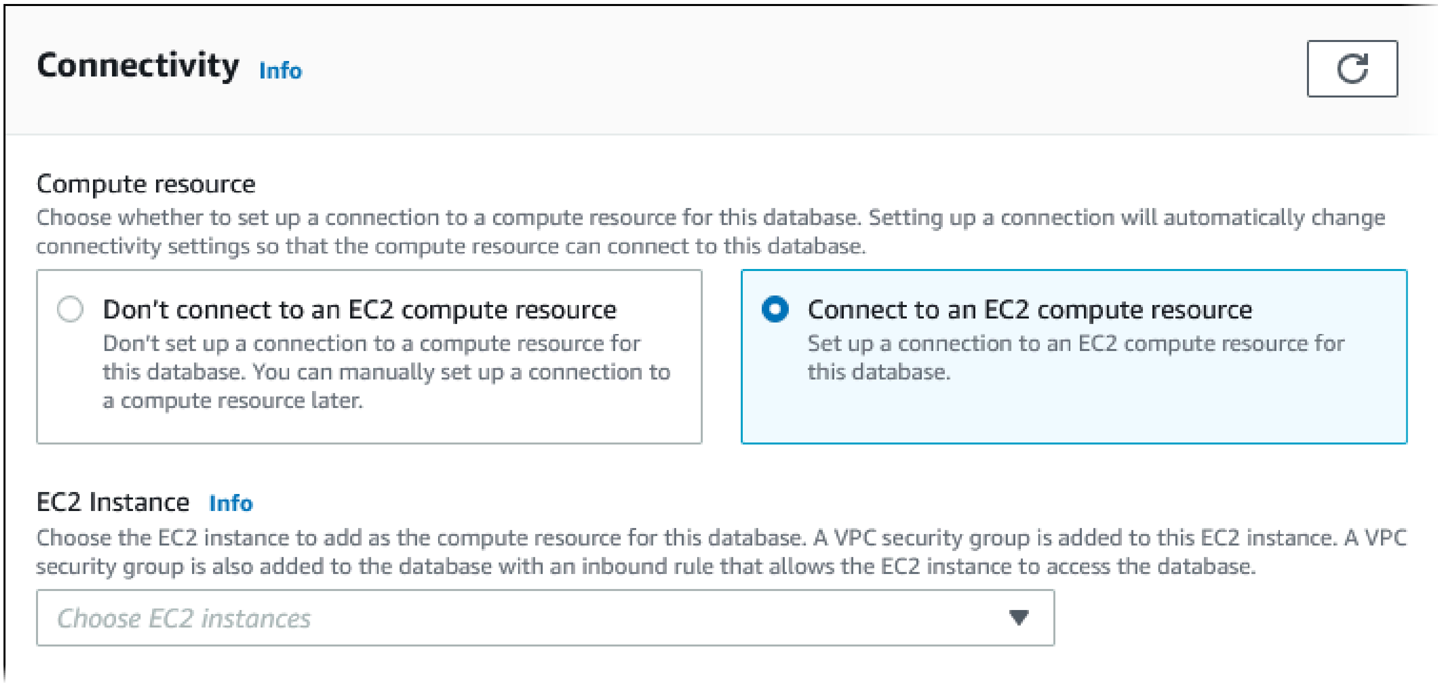
以下是将 EC2 实例与数据库集群连接的要求：

- 在创建数据库集群之前，EC2 实例必须存在于 Amazon Web Services 区域中。

  如果 Amazon Web Services 区域中不存在任何 EC2 实例，控制台将提供创建一个此类实例的链接。
- 创建数据库集群的用户必须具有执行以下操作的权限：
  - `ec2:AssociateRouteTable`
  - `ec2:AuthorizeSecurityGroupEgress`
  - `ec2:AuthorizeSecurityGroupIngress`
  - `ec2:CreateRouteTable`
  - `ec2:CreateSubnet`
  - `ec2:CreateSecurityGroup`
  - `ec2:DescribeInstances`
  - `ec2:DescribeNetworkInterfaces`
  - `ec2:DescribeRouteTables`
  - `ec2:DescribeSecurityGroups`
  - `ec2:DescribeSubnets`
  - `ec2:ModifyNetworkInterfaceAttribute`
  - `ec2:RevokeSecurityGroupEgress`

使用此选项创建私有数据库集群。数据库集群使用仅包含私有子网的数据库子网组，来限制对 VPC 内资源的访问。

要将 EC2 实例连接到数据库集群，请在 Create database（创建数据库）页面上的 Connectivity（连接）部分中，选择 Connect to an EC2 compute resource（连接到 EC2 计算资源）。



当您选择 Connect to an EC2 compute resource（连接到 EC2 计算资源）时，RDS 会自动设置以下选项。除非您通过选择 Don't connect to an EC2 compute resource（不要连接到 EC2 计算资源）来选择不与 EC2 实例建立连接，否则您无法更改这些设置。

控制台选项	自动设置
Virtual Private Cloud (VPC)	RDS 将 VPC 设置为与 EC2 实例关联的 VPC。
DB subnet group（数据库子网组）	RDS 要求在与 EC2 实例相同的可用区中具有带私有子网的数据库子网组。如果存在符合此要求的数据库子网组，则 RDS 将使用现有的数据库子网组。默认情况下，此选项设置为 Automatic setup（自动设置）。 当您选择 Automatic setup（自动设置）但没有满足此要求的数据库子网组时，将执行以下操作。RDS 在三个可用区中使用三个可用的私有子网，其中一个可用区与 EC2 实例相同。如果私有子网在可用区中不可用，则 RDS 会在可用区中创建私有子网。然后，RDS 创建数据库子网组。 当私有子网可用时，RDS 使用与该子网关联的路由表，并将它创建的任何子网添加到该路由表中。当没有可用的私有子网时，R

控制台选项	自动设置
	DS 会创建一个没有互联网网关访问权限的路由表，并将它创建的子网添加到该路由表中。 RDS 还允许您使用现有的数据库子网组。如果您想使用您选择的现有数据库子网组，请选择 Choose existing（选择现有）。
公有访问权限	RDS 选择 No（否），以使数据库集群不可供公开访问。 出于安全考虑，最好实践是保持数据库为私有，并确保不能从互联网访问数据库。
VPC security group (firewall) [VPC 安全组（防火墙）]	RDS 创建与数据库集群关联的新安全组。安全组命名为 `rds-ec2-n`，其中 *n* 是一个数值。该安全组包含一条以 EC2 VPC 安全组（防火墙）作为源的入站规则。这个与数据库集群关联的安全组允许 EC2 实例访问数据库集群。 RDS 还会创建一个与 EC2 实例关联的新安全组。安全组命名为 `ec2-rds-n`，其中 *n* 是一个数值。该安全组包含一条以数据库集群的 VPC 安全组作为源的出站规则。该安全组允许 EC2 实例向数据库集群发送流量。 您可以通过选择 Create new（新建）并键入新安全组的名称，添加另一个新安全组。 您可以通过选择 Choose existing（选择现有）并选择要添加的安全组，添加现有安全组。
可用区	RDS 在多可用区数据库集群部署中为一个数据库实例选择 EC2 实例的可用区。RDS 为其他两个数据库实例随机选择不同的可用区。写入器数据库实例在与 EC2 实例相同的可用区中创建。如果发生失效转移且写入器数据库实例位于不同的可用区，则可能会产生跨可用区成本。

有关这些设置的更多信息，请参阅 [创建多可用区数据库集群的设置]。

如果您在创建数据库集群后更改这些设置，则这些更改可能会影响 EC2 实例与数据库集群之间的连接。

手动配置网络

要从同一 VPC 中的 EC2 实例以外的资源连接到数据库集群，请手动配置网络连接。如果您使用 Amazon Web Services Management Console 创建多可用区数据库集群，您可以让 Amazon RDS 自动为您创建 VPC。或者，您也可以使用现有 VPC 或为您的多可用区数据库集群创建新的 VPC。要在多可用区数据库集群中使用您的 VPC，该 VPC 必须最少在三个可用区中均拥有一个子网。有关 VPC 的信息，请参阅 Amazon VPC 和 Amazon RDS。

如果您没有默认 VPC 或尚未创建 VPC，并且您不打算使用控制台，请执行以下操作：

- 在要部署数据库集群的 Amazon 区域中，创建最少在三个可用区中均至少有一个子网的 VPC。有关更多信息，请参阅 在 VPC 中使用数据库实例。
- 指定授权与您的 数据库集群的连接的 VPC 安全组。有关更多信息，请参阅通过创建安全组提供对 VPC 中的数据库实例的访问 和使用安全组控制访问权限。
- 指定 RDS 数据库子网组，该子网组在 VPC 中定义至少三个可由多可用区数据库集群使用的子网。有关更多信息，请参阅 使用数据库子网组。

有关适用于多可用区数据库集群的限制的信息，请参阅Amazon RDS 的多可用区数据库集群的限制。

如果您想连接到与多可用区数据库集群不在同一 VPC 中的资源，请参阅 在 VPC 中访问数据库实例的场景 中的相应方案。

其他先决条件

在创建多可用区数据库集群前，请考虑以下附加先决条件：

- 要定制您的数据库集群的配置参数，请通过必需的参数设置来指定数据库集群参数组。有关创建或修改数据库集群参数组的信息，请参阅 多可用区数据库集群的参数组。
- 请确定要为数据库集群指定的 TCP/IP 端口号。有些公司的防火墙将会阻止通往这些默认端口的连接。如果您的公司防火墙阻止默认端口，请为数据库集群选择其他端口。数据库集群中的所有数据库实例均使用相同的端口。
- 如果数据库的主引擎版本已到 RDS 标准支持终止日期，则必须使用 扩展支持CLI 选项或 RDS API 参数。有关更多信息，请参阅创建多可用区数据库集群的设置中的 RDS 扩展支持。

创建数据库集群

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 创建多可用区数据库集群。

控制台

您可以通过选择 Availability and durability（可用性与持久性）部分中的 Multi-AZ DB cluster（多可用区数据库集群）来创建多可用区数据库集群。

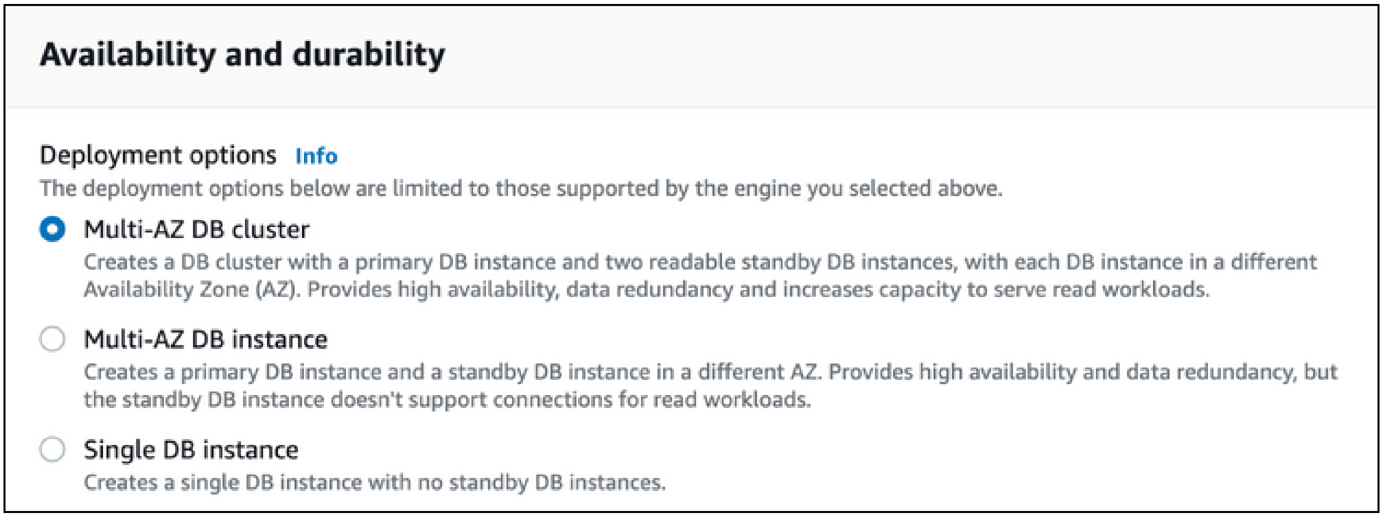
要使用控制台创建多可用区数据库集群

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在 Amazon Web Services Management Console 的右上角，选择要在其中创建数据库集群的 Amazon Web Services 区域。

   有关支持多可用区数据库集群的 Amazon Web Services 区域的更多信息，请参阅Amazon RDS 的多可用区数据库集群的限制。
3. 在导航窗格中，选择 Databases（数据库）。
4. 选择创建数据库。

   要创建多可用区数据库集群，请确保已选择 Standard Create（标准创建）并未选择 Easy Create（轻松创建）。
5. 在 Engine type（引擎类型）中，请选择 MySQL 或 PostgreSQL。
6. 对于 Version（版本），请选择数据库引擎版本。

   有关支持多可用区数据库集群的数据库引擎版本的信息，请参阅Amazon RDS 的多可用区数据库集群的限制。
7. 在 Templates（模板）中，请为您的部署选择适当的模板。
8. 在 Availability and durability（可用性与持久性）中，选择 Multi-AZ DB cluster（多可用区数据库集群）。



9. 在 DB cluster identifier（数据库集群标识符）中，请输入数据库集群的标识符。
10. 在 Master username（主用户名）中，请输入您的主用户名，或者保持默认设置。
11. 请输入您的主密码：

    a. 在设置部分中，打开凭证设置。
    b. 如果要指定密码，请取消勾选已选择的 Auto generate a password（自动生成密码）复选框。
    c. （可选）更改 Master username（主用户名）的值。
    d. 请在 Master password（主密码）和 Confirm password（确认密码）中输入相同的密码。
12. 对于数据库实例类，请选择数据库实例类。有关支持的数据库实例类的列表，请参阅the section called “多可用区数据库集群的实例类可用性”。
13. （可选）为该数据库集群设置与计算资源的连接。

    在创建数据库集群期间，您可以配置 Amazon EC2 实例和新数据库集群之间的连接。有关更多信息，请参阅 配置与 EC2 实例的自动网络连接。
14. 在连接部分的 VPC 安全组（防火墙）下，如果您选择新建，则会创建一个 VPC 安全组，其入站规则允许本地计算机的 IP 地址访问该数据库。
15. 对于其余部分，请指定数据库集群设置。有关每项设置的信息，请参阅 创建多可用区数据库集群的设置。
16. 选择创建数据库。

    如果选择使用自动生成的密码，则数据库页面上将显示查看凭证详细信息按钮。

    要查看数据库集群的主用户名和密码，请选择查看凭证详细信息。

    要以主用户身份连接到数据库集群，请使用显示的用户名和密码。

> **Important**
>
> 您无法再次查看主用户密码。

17. 对于 Databases（数据库），请选择新数据库集群的名称。

在 RDS 控制台上，将显示新数据库集群的详细信息。在此数据库集群完成创建、可供使用之前，此数据库集群的状态将为 Creating（正在创建）。当状态变为 Available（可用）时，您便可以连接到该数据库集群。根据所分配的数据库集群类和存储的不同，新数据库集群可能需要数分钟时间才可用。

Amazon CLI

在使用 Amazon CLI 创建多可用区数据库集群之前，确保满足所需的先决条件。其中包括创建 VPC 和 RDS 数据库子网组。有关更多信息，请参阅 数据库集群先决条件。

要使用 Amazon CLI 创建多可用区数据库集群，请调用 create-db-cluster 命令。请指定 `--db-cluster-identifier`。对于 `--engine` 选项，请指定 `mysql` 或 `postgres`。

有关各选项的信息，请参阅 创建多可用区数据库集群的设置。

有关支持多可用区数据库集群的 Amazon Web Services 区域、数据库引擎和数据库引擎版本的信息，请参阅Amazon RDS 的多可用区数据库集群的限制。

`create-db-cluster` 命令将为您的数据库集群创建写入器数据库实例，以及两个读取器数据库实例。每个数据库实例都位于不同的可用区。

例如，以下命令将创建一个名为 `mysql-multi-az-db-cluster` 的 MySQL 8.0 多可用区数据库集群。

Example

对于 Linux、macOS 或 Unix：

```
aws rds create-db-cluster \
   --db-cluster-identifier mysql-multi-az-db-cluster \
   --engine mysql \
   --engine-version 8.0.32  \
   --master-username admin \
   --manage-master-user-password  \
   --port 3306 \
   --backup-retention-period 1  \
```

```
    --db-subnet-group-name default \
    --allocated-storage 4000 \
    --storage-type io1 \
    --iops 10000 \
    --db-cluster-instance-class db.m5d.xlarge
```

对于 Windows：

```
aws rds create-db-cluster ^
    --db-cluster-identifier mysql-multi-az-db-cluster ^
    --engine mysql ^
    --engine-version 8.0.32 ^
    --manage-master-user-password ^
    --master-username admin ^
    --port 3306 ^
    --backup-retention-period 1 ^
    --db-subnet-group-name default ^
    --allocated-storage 4000 ^
    --storage-type io1 ^
    --iops 10000 ^
    --db-cluster-instance-class db.m5d.xlarge
```

以下命令将创建一个名为 `postgresql-multi-az-db-cluster` 的 PostgreSQL 13.4 多可用区数据库集群。

Example

对于 Linux、macOS 或 Unix：

```
aws rds create-db-cluster \
    --db-cluster-identifier postgresql-multi-az-db-cluster \
    --engine postgres \
    --engine-version 13.4 \
    --manage-master-user-password \
    --master-username postgres \
    --port 5432 \
    --backup-retention-period 1  \
    --db-subnet-group-name default \
    --allocated-storage 4000 \
    --storage-type io1 \
    --iops 10000 \
    --db-cluster-instance-class db.m5d.xlarge
```

对于 Windows：

```
aws rds create-db-cluster ^
   --db-cluster-identifier postgresql-multi-az-db-cluster ^
   --engine postgres ^
   --engine-version 13.4 ^
   --manage-master-user-password ^
   --master-username postgres ^
   --port 5432 ^
   --backup-retention-period 1 ^
   --db-subnet-group-name default ^
   --allocated-storage 4000 ^
   --storage-type io1 ^
   --iops 10000 ^
   --db-cluster-instance-class db.m5d.xlarge
```

RDS API

在使用 RDS API 创建多可用区数据库集群之前，您必须满足所需的先决条件，例如创建 VPC 和 RDS 数据库子网组。有关更多信息，请参阅 数据库集群先决条件。

要使用 RDS API 创建多可用区数据库集群，请调用 CreateDBCluster 操作。请指定 `DBClusterIdentifier`。对于 `Engine` 参数，请指定 `mysql` 或 `postgresql`。

有关各选项的信息，请参阅 创建多可用区数据库集群的设置。

`CreateDBCluster` 操作将为您的数据库集群创建写入器数据库实例，以及两个读取器数据库实例。每个数据库实例都位于不同的可用区。

创建多可用区数据库集群的设置

有关创建多可用区数据库集群时所选设置的详细信息，请参阅下表。有关 Amazon CLI 选项的更多信息，请参阅 create-db-cluster。有关 RDS API 参数的更多信息，请参阅 CreateDBCluster。

控制台设置	设置说明	CLI 选项和 RDS API 参数
分配的存储空间	为数据库集群中的每个数据库实例分配的存储量（以 GiB 为单位）。有关更多信息，请参阅 Amazon RDS 数据库实例存储。	CLI 选项： `--allocated-storage` API 参数：

控制台设置	设置说明	CLI 选项和 RDS API 参数
		`AllocatedStorage`

自动次要版本升级	请选择 Enable auto minor version upgrade（启用自动次要版本升级），以使数据库集群能够在首选次要数据库引擎版本升级可用时自动接收这些升级。Amazon RDS 在维护时段内执行自动次要版本升级。	CLI 选项：
		`--auto-minor-version-upgrade`
		`--no-auto-minor-version-upgrade`
		API 参数：
		`AutoMinorVersionUpgrade`

备份保留期	您希望数据库集群自动备份的保留天数。对于多可用区数据库集群，此值必须设置为 **1** 或更大。
	有关更多信息，请参阅 备份简介。
		CLI 选项：
		`--backup-retention-period`
		API 参数：
		`BackupRetentionPeriod`

备份时段	Amazon RDS 自动备份数据库集群的时间段。除非您知道需要何时进行数据库备份，否则，请使用 No preference（无首选项）的默认设置。
	有关更多信息，请参阅“备份简介”。
		CLI 选项：
		`--preferred-backup-window`
		API 参数：
		`PreferredBackupWindow`

证书颁发机构	数据库集群使用的服务器证书的证书颁发机构（CA）。
	有关更多信息，请参阅 使用 SSL/TLS 加密与数据库实例或集群的连接。
		CLI 选项：
		`--ca-certificate-identifier`
		RDS API 参数：
		`CACertificateIdentifier`

控制台设置	设置说明	CLI 选项和 RDS API 参数

将标签复制到快照	在创建快照时，该选项将任何数据库集群标签复制到数据库快照中。
	
	有关更多信息，请参阅 为 Amazon RDS 资源添加标签。
		CLI 选项：
		
		`-copy-tags-to-snapshot`
		
		`-no-copy-tags-to-snapshot`
		
		RDS API 参数：
		
		`CopyTagsToSnapshot`

数据库身份验证	您想要使用的数据库身份验证选项。
	
	选择 Password authentication 以仅使用数据库密码验证数据库用户的身份。
	
	选择 Password and IAM DB authentication（密码和 IAM 数据库身份验证），以通过用户和角色的数据库密码和用户凭证验证数据库用户的身份。有关更多信息，请参阅 适用于 MariaDB、MySQL 和 PostgreSQL 的 IAM 数据库身份验证。
		CLI 选项：
		
		`--enable-iam-database-authentication`
		
		`--no-enable-iam-database-authentication`
		
		RDS API 参数：
		
		`EnableIAMDatabaseAuthentication`

数据库端口	要用于访问数据库集群的端口。显示的是默认端口。
	
	创建数据库集群后将无法更改端口。
	
	有些公司的防火墙将会阻止通往这些默认端口的连接。如果您的公司防火墙阻止默认端口，请为数据库集群输入其他端口。
		CLI 选项：
		
		`--port`
		
		RDS API 参数：
		
		`Port`

控制台设置	设置说明	CLI 选项和 RDS API 参数

数据库集群标识符	数据库集群的名称。请使用与命名本地部署服务器相同的方式命名数据库集群。数据库集群标识符最多可以包含 63 个字母数字字符，且必须在所选 Amazon 区域中对于您的账户是唯一的。	CLI 选项：
		`--db-cluster-identifier`
		RDS API 参数：
		`DBClusterIdentifier`

数据库实例类	多可用区数据库集群中各数据库实例（例如 `db.m5d.xlarge` ）的计算和内存容量。
	如有可能，请选择足够大的数据库实例类以便典型查询工作集能够保留在内存中。如果在内存中保留工作集，系统可以避免写入到磁盘，从而提高性能。
	有关支持的数据库实例类的列表，请参阅the section called “多可用区数据库集群的实例类可用性”。
		CLI 选项：
		`--db-cluster-instance-class`
		RDS API 参数：
		`DBClusterInstanceClass`

数据库集群参数组	要与数据库集群关联的数据库集群参数组。
	有关更多信息，请参阅 多可用区数据库集群的参数组。
		CLI 选项：
		`--db-cluster-parameter-group-name`
		RDS API 参数：
		`DBClusterParameterGroupName`

数据库引擎版本	要使用的数据库引擎的版本。	CLI 选项：
		`--engine-version`
		RDS API 参数：
		`EngineVersion`

控制台设置	设置说明	CLI 选项和 RDS API 参数
数据库集群参数组	要与数据库集群关联的数据库实例参数组。 有关更多信息，请参阅 多可用区数据库集群的参数组。	CLI 选项： `--db-cluster-parameter-group-name` RDS API 参数： **`DBClusterParameterGroupName`**

DB subnet group（数据库子网组）	要用于数据库集群的数据库子网组。选择 Choose existing（选择现有）以使用现有的数据库子网组。然后，从 Existing DB subnet groups（现有数据库子网组）下拉列表中选择所需的子网组。
	
	选择 Automatic setup（自动设置），让 RDS 选择兼容的数据库子网组。如果不存在任何子网组，RDS 会为您的集群创建一个新的子网组。
	
	有关更多信息，请参阅 使用数据库子网组。
		CLI 选项：
		
		`--db-subnet-group-name`
		
		RDS API 参数：
		
		`DBSubnetGroupName`

删除保护	启用删除保护以禁止删除数据库集群。如果使用控制台创建生产数据库集群，预设情况下将开启删除保护。
	
	有关更多信息，请参阅 删除数据库实例。
		CLI 选项：
		
		`--deletion-protection`
		
		`--no-deletion-protection`
		
		RDS API 参数：
		
		`DeletionProtection`

控制台设置	设置说明	CLI 选项和 RDS API 参数

加密	请选择 Enable Encryption（启用加密）以对该数据库集群开启静态加密。
	
	预设情况下，将对多可用区数据库集群开启加密。
	
	有关更多信息，请参阅 加密 Amazon RDS 资源。
		CLI 选项：
		
		`--kms-key-id`
		
		`--storage-encrypted`
		
		`--no-storage-encrypted`
		
		RDS API 参数：
		
		`KmsKeyId`
		
		`StorageEncrypted`

增强监控	选择 Enable enhanced monitoring（启用增强监控）可为您的数据库集群运行所在的操作系统开启实时指标收集。
	
	有关更多信息，请参阅 使用增强监控来监控操作系统指标。
		CLI 选项：
		
		`--monitoring-interval`
		
		`--monitoring-role-arn`
		
		RDS API 参数：
		
		`MonitoringInterval`
		
		`MonitoringRoleArn`

控制台设置	设置说明	CLI 选项和 RDS API 参数

Initial database name (初始数据库名称)	数据库集群的数据库名称。如果您未提供名称，Amazon RDS 不会在 MySQL 的数据库集群上创建数据库。但是，它确实会在 PostgreSQL 的数据库集群上创建数据库。名称不能是数据库引擎的保留字。它还有其他约束，具体取决于数据库引擎。
	
	MySQL：
	
	• 必须包含 1–64 个字母数字字符。
	
	PostgreSQL：
	
	• 必须包含 1–63 个字母数字字符。
	• 必须以字母或下划线开头。后续字符可以是字母、下划线或数字 (0-9)。
	• 初始数据库名称为 `postgres`。
		CLI 选项：
		
		`--database-name`
		
		RDS API 参数：
		
		`DatabaseName`

日志导出	要发布到 Amazon CloudWatch Logs 的数据库日志文件的类型。
	
	有关更多信息，请参阅“将数据库日志发布到 Amazon CloudWatch Logs”。
		CLI 选项：
		
		`-enable-cloudwatch-logs-exports`
		
		RDS API 参数：
		
		`EnableCloudwatchLogsExports`

控制台设置	设置说明	CLI 选项和 RDS API 参数

维护窗口	30 分钟时段，在这段时间内，将会应用对数据库集群的待处理修改。如果该时段无关紧要，请选择 No preference（无首选项）。
	
	有关更多信息，请参阅“Amazon RDS 维护时段”。
		CLI 选项：
		
		`--preferred-maintenance-window`
		
		RDS API 参数：
		
		`PreferredMaintenanceWindow`

在 Amazon Secrets Manager 中管理主凭证	选择在 Amazon Secrets Manager 中管理主凭证，以在 Secrets Manager 的密钥中管理主用户密码。
	
	（可选）选择用于保护密钥的 KMS 密钥。请从您的账户的 KMS 密钥中进行选择，或输入来自其他账户的密钥。
	
	有关更多信息，请参阅 使用 Amazon RDS 和 Amazon Secrets Manager 管理密码。
		CLI 选项：
		
		`--manage-master-user-password | --no-manage-master-user-password`
		
		`--master-user-secret-kms-key-id`
		
		RDS API 参数：
		
		`ManageMasterUserPassword`
		
		`MasterUserSecretKmsKeyId`

主密码	您的主用户账户密码。	CLI 选项：
		
		`--master-user-password`
		
		RDS API 参数：
		
		`MasterUserPassword`

控制台设置	设置说明	CLI 选项和 RDS API 参数

主用户名	用作登录数据库集群的主用户名，拥有所有数据库权限。
	
	• 它可以包含 1–16 个字母数字字符和下划线。
	• 它的第一个字符必须是字母。
	• 它不能是数据库引擎的保留字。
	
	您无法在创建多可用区数据库集群之后更改主用户名。
	
	有关授予主用户权限的更多信息，请参阅 主用户账户权限。
		CLI 选项：
		
		`--master-username`
		
		RDS API 参数：
		
		`MasterUsername`

控制台设置	设置说明	CLI 选项和 RDS API 参数

Performance Insights	请选择 Enable Performance Insights（启用性能详情）以监控数据库集群负载，以便您可以分析数据库性能和解决数据库性能问题。
	
	选择保留期以确定要保留的性能详情历史记录数量。免费套餐中的保留设置为 Default (7 days) [默认设置（7 天）]。要将性能数据保留更长时间，请指定 1–24 个月。有关保留期的更多信息，请参阅性能详情的定价和数据留存。
	
	选择一个 KMS 密钥以用来保护用于加密该数据库卷的密钥。请从您的账户的 KMS 密钥中进行选择，或输入来自其他账户的密钥。
	
	有关更多信息，请参阅 在 Amazon RDS 上使用性能详情监控数据库负载。
		CLI 选项：
		
		`--enable-performance-insights`
		
		`--no-enable-performance-insights`
		
		`--performance-insights-retention-period`
		
		`--performance-insights-kms-key-id`
		
		RDS API 参数：
		
		`EnablePerformanceInsights`
		
		`PerformanceInsightsRetentionPeriod`
		
		`PerformanceInsightsKMSKeyId`

预置 IOPS	最初为数据库集群分配的预置 IOPS（每秒输入/输出操作数）量。	CLI 选项：
		
		`--iops`
		
		RDS API 参数：
		
		`Iops`

控制台设置	设置说明	CLI 选项和 RDS API 参数

公有访问权限	请选择 Publicly accessible（可公开访问）以为数据库集群提供公有 IP 地址，这表示可以在 VPC 外部访问该实例。要可供公开访问，数据库集群还必须在 VPC 的公有子网中。
	
	请选择 Not publicly accessible（不可公开访问）以设置只能从 VPC 内部访问数据库集群。
	
	有关更多信息，请参阅 对互联网隐藏 VPC 中的数据库实例。
	
	要从其 VPC 外部连接到数据库集群，该数据库集群必须可公开访问。此外，必须使用数据库集群安全组的入站规则授予访问权限，并且必须满足其他要求。有关更多信息，请参阅 无法连接到 Amazon RDS 数据库实例。
	
	如果您的数据库集群不可公开访问，则您可以使用 Amazon Site-to-Site VPN 连接或 Amazon Direct Connect 连接从专用网络访问该实例。有关更多信息，请参阅 互联网络流量隐私。
		CLI 选项：
		
		`--publicly-accessible`
		
		`--no-publicly-accessible`
		
		RDS API 参数：
		
		`PubliclyAccessible`

控制台设置	设置说明	CLI 选项和 RDS API 参数

RDS 扩展支持	选择启用 RDS Extended Support，以允许受支持的主要引擎版本在 RDS 标准支持终止日期后继续运行。
	
	当您创建数据库集群时，Amazon RDS 默认使用 RDS 扩展支持。为了防止在 RDS 标准支持终止日期后创建新的数据库集群并避免支付 RDS 扩展支持费用，请禁用此设置。在 RDS 扩展支持定价开始日期之前，您的现有数据库集群不会产生费用。
	
	有关更多信息，请参阅 Amazon RDS 的 Amazon RDS 扩展支持。
		CLI 选项：
		
		`--engine-lifecycle-support`
		
		RDS API 参数：
		
		`EngineLifecycleSupport`

Storage throughput（存储吞吐量）	数据库集群的存储吞吐量值。仅在您为存储类型选择通用型 SSD（gp3）时，此设置才可见。 此设置不可配置，可根据您指定的 IOPS 自动设置。 有关更多信息，请参阅 gp3 存储（推荐）。	此值是自动计算的，并且没有 CLI 选项。
RDS Proxy（RDS 代理）	选择 Create an RDS Proxy（创建 RDS 代理），以便为您的数据库集群创建代理。Amazon RDS 会自动为代理创建 IAM 角色和 Secrets Manager 密钥。	创建数据库集群时不可用。

控制台设置	设置说明	CLI 选项和 RDS API 参数

存储类型	数据库集群的存储类型。
	
	仅支持通用型 SSD（gp3）、预调配 IOPS（io1）和预调配 IOPS SSD（io2）存储。
	
	有关更多信息，请参阅 Amazon RDS 存储类型。
		CLI 选项：
		
		`--storage-type`
		
		RDS API 参数：
		
		`StorageType`

Virtual Private Cloud (VPC)	基于要与此数据库集群关联的 Amazon VPC 服务的 VPC。 有关更多信息，请参阅 Amazon VPC 和 Amazon RDS。	对于 CLI 和 API，您可以指定 VPC 安全组 ID。

VPC security group (firewall) [VPC 安全组（防火墙）]	要与数据库集群关联的安全组。
	
	有关更多信息，请参阅 VPC 安全组概述。
		CLI 选项：
		
		`--vpc-security-group-ids`
		
		RDS API 参数：
		
		`VpcSecurityGroupIds`

创建多可用区数据库集群时不适用的设置

以下 Amazon CLI 命令 `create-db-cluster` 和 RDS API 操作 `CreateDBCluster` 中的设置不适用于多可用区数据库集群。

您也无法在控制台中为多可用区数据库集群指定这些设置。

Amazon CLI 设置	RDS API 设置
`--availability-zones`	`AvailabilityZones`
`--backtrack-window`	`BacktrackWindow`
`--character-set-name`	`CharacterSetName`

Amazon CLI 设置	RDS API 设置
--domain	Domain
--domain-iam-role-name	DomainIAMRoleName
--enable-global-write-forwarding | --no-enable-global-write-forwarding	EnableGlobalWriteForwarding
--enable-http-endpoint | --no-enable-http-endpoint	EnableHttpEndpoint
--global-cluster-identifier	GlobalClusterIdentifier
--option-group-name	OptionGroupName
--pre-signed-url	PreSignedUrl
--replication-source-identifier	ReplicationSourceIdentifier
--scaling-configuration	ScalingConfiguration

连接到 Amazon RDS 的多可用区数据库集群

多可用区数据库集群具有三个数据库实例，而不是单个数据库实例。每个连接均由特定的数据库实例处理。在连接到多可用区数据库集群时，您指定的主机名和端口将指向名为端点 的完全限定域名。多可用区数据库集群使用端点机制来提取这些连接，因此，您无需确切指定数据库集群中要连接的数据库实例。因此，当某些数据库实例不可用时，您不必对所有主机名进行硬编码或编写自己的逻辑来重新路由连接。

写入器端点将连接到数据库集群的写入器数据库实例，该实例同时支持读取和写入操作。读取器端点连接到两个读取器数据库实例之一，这两个实例仅支持读取操作。

通过使用端点，您可以根据使用案例将每个连接映射到相应的数据库实例或数据库实例组。例如，要执行 DDL 和 DML 语句，您可以连接到作为写入器数据库实例的任一数据库实例。要执行查询，您可以连接到读取器端点，并通过多可用区数据库集群自动在读取器数据库实例之间管理连接。对于诊断或优化，您可以连接到特定数据库实例端点以检查有关特定数据库实例的详细信息。

有关连接到数据库实例的信息，请参阅 连接到 Amazon RDS 数据库实例。

有关连接到多可用区数据库集群的更多信息，请参阅以下主题。

主题

- 集群端点
- 读取器终端节点
- 实例端点
- 高可用性连接
- 使用 Amazon 驱动程序连接到 Amazon RDS 的多可用区数据库集群

多可用区数据库集群端点的类型

端点由包含主机地址的唯一标识符表示。可从多可用区数据库集群使用以下类型的端点。

集群端点

多可用区数据库集群的集群端点（或写入器端点）连接到该数据库集群的当前写入器数据库实例。此端点是唯一可以执行写操作（如 DDL 和 DML 语句）的端点。此端点还可以执行读取操作。

每个多可用区数据库集群均有一个集群端点和一个写入器数据库实例。

对数据库集群上的所有写入操作使用集群端点，这些操作包括插入、更新、删除和 DDL 更改。您还可以对读取操作（如查询）使用集群端点。

如果数据库集群的当前写入器数据库实例失败，多可用区数据库集群将自动故障转移到新的写入器数据库实例。在故障转移期间，数据库集群将继续为从新的写入器数据库实例到集群端点的请求提供服务，对服务造成的中断最少。

以下示例介绍多可用区数据库集群中的集群端点。

```
mydbcluster.cluster-123456789012.us-east-1.rds.amazonaws.com
```

有关连接到集群端点的更多信息，请参阅集群端点。

读取器端点

多可用区数据库集群的读取器端点为数据库集群的只读连接提供支持。对读取操作（如 SELECT 查询）使用读取器端点。通过在读取器数据库实例上处理这些语句，此端点可减少写入器数据库实例上的开销。它还可以帮助集群扩展容量以同时 SELECT 查询。每个多可用区数据库集群均有一个读取器端点。

读取器端点向读取器数据库实例之一发送每个连接请求。在会话中使用读取器端点时，您只能执行只读语句，例如该会话中的 SELECT。

以下示例介绍多可用区数据库集群的读取器端点。读取器端点的只读意图由集群端点名称中的 `-ro` 指示。

```
mydbcluster.cluster-ro-123456789012.us-east-1.rds.amazonaws.com
```

有关连接到读取器端点的更多信息，请参阅读取器终端节点。

实例端点

实例端点会连接到多可用区数据库集群中的特定数据库实例。数据库集群中的每个数据库实例具有自己的唯一实例端点。因此，数据库集群的当前写入器数据库实例具有一个实例端点，并且数据库集群中的每个读取器数据库实例都具有一个实例端点。

实例端点提供对数据库集群连接的直接控制。此控制可以帮助您解决可能不适合使用集群端点或读取器端点的场景。例如，客户端应用程序可能根据工作负载类型需要更精细的负载均衡。在这种情况下，您可以配置多个客户端以连接到数据库集群中的不同读取器数据库实例，以便分配读取工作负载。

以下示例介绍多可用区数据库集群中数据库实例的实例端点。

```
mydbinstance.123456789012.us-east-1.rds.amazonaws.com
```

有关连接到实例端点的更多信息，请参阅实例端点。

查看端点

使用控制台、Amazon CLI 或 Amazon RDS API 查看集群端点、读取器端点和实例端点。

Console

在 Amazon Web Services Management Console 中，您可以在每个多可用区数据库集群的详细信息页面中查看集群端点和读取器端点。您可在每个数据库实例的详细信息页面中查看实例端点。

Amazon CLI

利用 Amazon CLI，您会在 describe-db-clusters 命令的输出中看到写入器和读取器端点。例如，以下命令显示当前 Amazon 区域中所有集群的端点属性。

```
aws rds describe-db-cluster-endpoints
```

Amazon RDS API

利用 Amazon RDS API，您可以通过调用 DescribeDBClusterEndpoints 操作来检索端点。输出还将显示 Amazon Aurora 数据库集群端点（如果存在）。

集群端点

每个多可用区数据库集群都有一个内置读取器端点，其名称和其他属性均由 Amazon RDS 管理。您无法创建、删除或修改此类端点。

在管理数据库集群、执行提取、转换、加载 (ETL) 操作或开发和测试应用程序时，可以使用集群端点。集群端点将连接到集群的写入器数据库实例。写入器数据库实例是您可以在其中创建表和索引、运行 `INSERT` 语句以及执行其他 DDL 和 DML 操作的唯一数据库实例。

当故障转移机制将新数据库实例提升为集群的写入器数据库实例时，集群端点指向的物理 IP 地址会发生更改。如果您使用任意形式的连接池或其他多路复用，请准备刷新或减少任何缓存的 DNS 信息的生存时间。这样做可确保您不会尝试与在故障转移后不可用或现在为只读的数据库实例建立读/写连接。

读取器终端节点

您将读取器端点用于多可用区数据库集群的只读连接。此端点可帮助数据库集群处理查询密集型工作负载。读取器端点是您向在集群上执行报告或其他只读操作的应用程序提供的端点。读取器端点向多可用区数据库数据库集群中的可用读取器数据库实例发送连接。

每个多可用区集群都有一个内置读取器端点，其名称和其他属性由 Amazon RDS 管理。您无法创建、删除或修改此类端点。

实例端点

多可用区数据库集群中的每个数据库实例均有自己的内置实例端点，其名称和其他属性由 Amazon RDS 管理。您无法创建、删除或修改此类端点。使用多可用区数据库集群时，与实例端点相比，您通常会更频繁地使用写入器和读取器端点。

在日常操作中，使用实例端点的主要方式是诊断影响多可用区数据库集群中某个特定数据库实例的容量或性能问题。在连接到特定数据库实例时，您可以检查其状态变量、指标等。这样做可以帮助您确定该数据库实例与集群中其他数据库实例的不同之处。

高可用性连接

对于高可用性非常重要的多可用区数据库集群，请使用写入器端点进行读/写或通用型连接，并使用读取器端点进行只读连接。写入器和读取器端点比实例端点更好地管理数据库实例故障转移。与实例端点不同，如果集群中的数据库实例变得不可用，写入器和读取器端点会自动更改其连接到的数据库实例。

如果数据库集群的写入器数据库实例失败，Amazon RDS 将自动故障转移到新的写入器数据库实例。它通过将读取器数据库实例提升为新的写入器数据库实例来实现这一目标。如果发生了故障转移，您可以使用写入器端点重新连接到新提升的写入器数据库实例。或者，您可以使用读取器端点重新连接到数据库集群中的读取器数据库实例之一。在故障转移期间，在将读取器数据据可实例提升为新的写入器数据库实例之后，读取器端点可能会在很短的时间内将连接定向到数据库集群的新写入器数据库实例。如果您设计自己的应用程序逻辑来管理实例端点连接，则可以手动或以编程方式搜索数据库集群中生成的可用数据库实例集。

使用 Amazon 驱动程序连接到 Amazon RDS 的多可用区数据库集群

借助 Amazon 驱动程序套件，可显著缩短切换和故障转移时间，并支持使用 Amazon Secrets Manager、Amazon Identity and Access Management（IAM）和联合身份进行身份验证。Amazon 驱动程序依靠监控数据库集群状态和了解集群拓扑，来确定新的写入器。这种方法将切换和故障转移时间缩短到几秒钟，而开源驱动程序的切换和故障转移时间则为几十秒。

随着新服务功能的推出，使用 Amazon 驱动程序套件可为这些服务功能提供内置支持。

使用 Amazon Web Services（Amazon）JDBC 驱动程序连接到多可用区数据库集群

Amazon Web Services（Amazon）JDBC 驱动程序设计为高级 JDBC 包装器，可帮助应用程序利用集群数据库的功能。此包装器是对现有 JDBC 驱动程序各项功能的补充和扩展。该驱动程序与以下社区驱动程序兼容：

- MySQL Connector/J
- MariaDB Connector/J
- pgJDBC

要安装 Amazon JDBC 驱动程序，请附加 Amazon JDBC 驱动程序.jar 文件（位于应用程序 `CLASSPATH` 中），并保留对相应社区驱动程序的引用。按如下方式更新相应的连接 URL 前缀：

- `jdbc:mysql://` 到 `jdbc:aws-wrapper:mysql://`
- `jdbc:mariadb://` 到 `jdbc:aws-wrapper:mariadb://`
- `jdbc:postgresql://` 到 `jdbc:aws-wrapper:postgresql://`

有关 Amazon JDBC 驱动程序的更多信息及其完整使用说明，请参阅 Amazon Web Services (Amazon) JDBC Driver GitHub repository。

使用 Amazon Web Services（Amazon）Python 驱动程序连接到多可用区数据库集群

Amazon Web Services（Amazon）驱动程序设计为高级 Python 包装器。这款包装器是对开源 Psycopg 驱动程序的各项功能的补充和扩展。Amazon Python 驱动程序支持 Python 3.8 及更高版本。您可以使用 `pip` 命令和 `psycopg` 开源软件包安装 `aws-advanced-python-wrapper` 程序包。

有关 Amazon JDBC 驱动程序的更多信息及其完整使用说明，请参阅 Amazon Web Services（Amazon）Python 驱动程序 GitHub 存储库。

自动连接 Amazon RDS 的 Amazon 计算资源和多可用区数据库集群

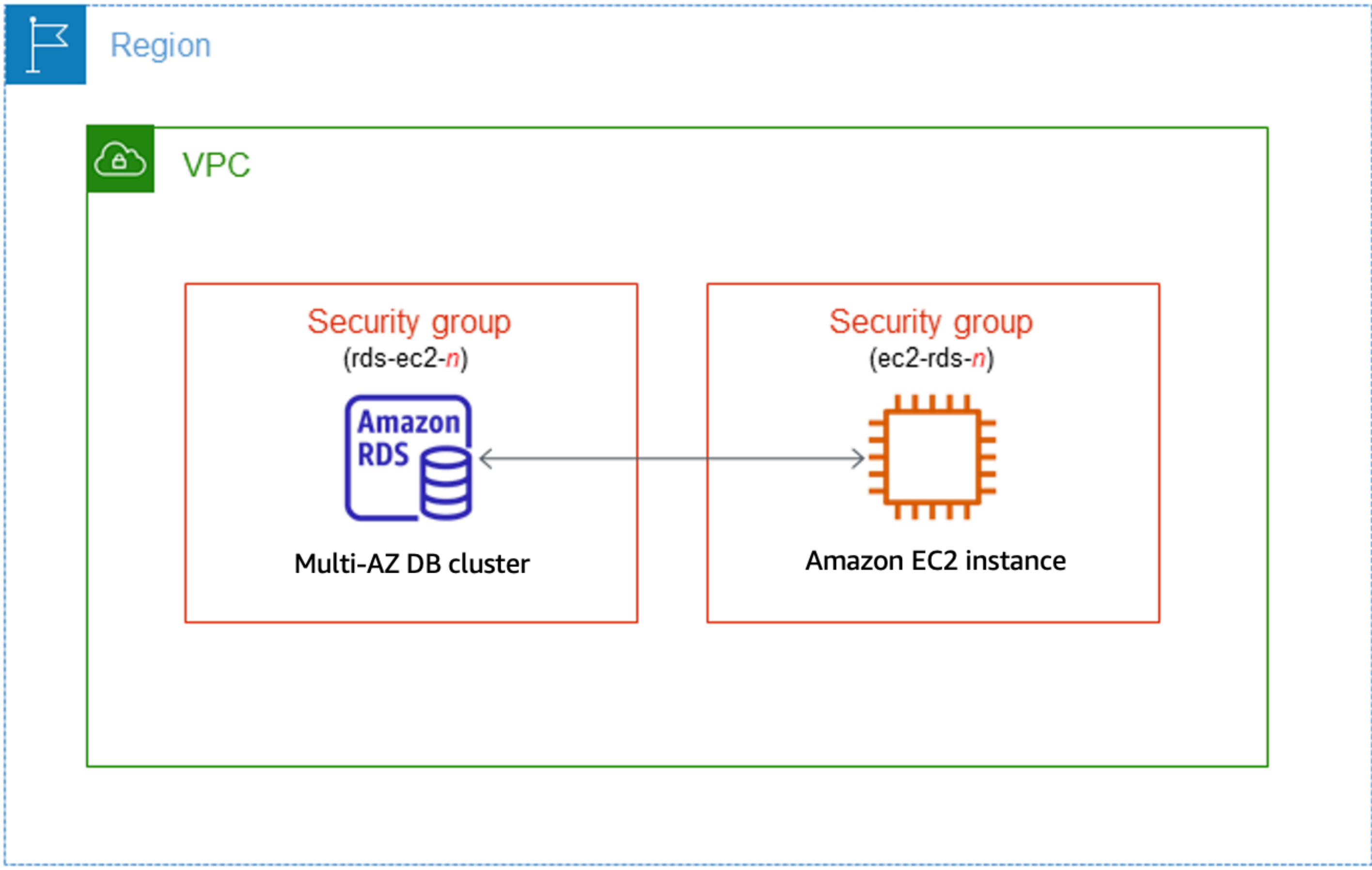
您可以自动连接多可用区数据库集群和 Amazon 计算资源，例如 Amazon Elastic Compute Cloud（Amazon EC2）实例和 Amazon Lambda 函数。

以下主题详细说明了如何配置网络设置、安全组和连接参数，以便在多可用区数据库集群部署中与 Amazon RDS 数据库实例建立可靠的连接。这些主题专注于优化与多可用区数据库集群交互的应用程序的网络连接和性能，从而确保安全、高效的数据操作。

主题

自动连接 EC2 实例和多可用区数据库集群

您可以使用 RDS 控制台来简化在 Amazon Elastic Compute Cloud（Amazon EC2）实例和多可用区数据库集群之间设置连接的过程。通常，多可用区数据库集群位于私有子网中，而 EC2 实例位于 VPC 内的公有子网中。您可以在 EC2 实例上使用 SQL 客户端来连接到多可用区数据库集群。EC2 实例还可以运行用于访问私有多可用区数据库集群的 Web 服务器或应用程序。



如果您想连接到与多可用区数据库集群不在同一 VPC 中的 EC2 实例，请参阅the section called "在 VPC 中访问数据库实例的场景"中的相应方案。

主题

- 与 EC2 实例的自动连接概述
- 自动连接 EC2 实例和多可用区数据库集群
- 查看连接的计算资源

与 EC2 实例的自动连接概述

当您自动在 EC2 实例和多可用区数据库集群之间设置连接时，Amazon RDS 会为您的 EC2 实例和数据库集群配置 VPC 安全组。

以下是将 EC2 实例与多可用区数据库集群连接的要求：

- EC2 实例必须与多可用区数据库集群存在于同一 VPC 中。

  如果同一 VPC 中不存在任何 EC2 实例，控制台将提供创建一个此类实例的链接。

- 设置连接的用户必须具有执行以下 EC2 操作的权限：
  - `ec2:AuthorizeSecurityGroupEgress`
  - `ec2:AuthorizeSecurityGroupIngress`
  - `ec2:CreateSecurityGroup`
  - `ec2:DescribeInstances`
  - `ec2:DescribeNetworkInterfaces`
  - `ec2:DescribeSecurityGroups`
  - `ec2:ModifyNetworkInterfaceAttribute`
  - `ec2:RevokeSecurityGroupEgress`

当您设置与 EC2 实例的连接时，Amazon RDS 会根据与多可用区数据库集群和 EC2 实例关联的安全组的当前配置采取操作，如下表所述。

当前 RDS 安全组配置	当前 EC2 安全组配置	RDS 操作
有一个或多个安全组与多可用区数据库集群 [其名称与模式 `rds-ec2-n`（其中 *n* 是数字）相匹配] 关联。尚未修改与此模式匹配的安全组。该安全组只具有一条以 EC2 实例的 VPC 安全组作为源的入站规则。	有一个或多个安全组与 EC2 实例关联，此实例的名称与模式 `rds-ec2-n`（其中 *n* 是数字）相匹配。尚未修改与此模式匹配的安全组。该安全组只具有一条以多可用区数据库集群的 VPC 安全组作为源的出站规则。	Amazon RDS 不执行任何操作。 已在 EC2 实例和多可用区数据库集群之间自动配置了连接。由于 EC2 实例和 RDS 数据库之间已经存在连接，因此不会修改安全组。
以下任一条件适用： • 没有与多可用区数据库集群（其名称与模式 `rds-ec2-n` 匹配）关联的安全组。 • 有一个或多个安全组与多可用区数据库集群（其名称与模式 `rds-ec2-n` 相匹配）	以下任一条件适用： • 没有与 EC2 实例（其名称与模式 `ec2-rds-n` 匹配）关联的安全组。 • 有一个或多个安全组与 EC2 实例关联，此实例的名称与模式 `ec2-rds-n` 相匹配。	[RDS action: create new security groups](#)

当前 RDS 安全组配置	当前 EC2 安全组配置	RDS 操作
关联。但是，这些安全组都不能用于与 EC2 实例的连接。如果安全组没有一条以 EC2 实例的 VPC 安全组作为源的入站规则，则无法使用该安全组。如果已修改了安全组，也不能使用它。修改示例包括添加规则或更改现有规则的端口。	但是，这些安全组都不能用于与多可用区数据库集群的连接。如果安全组没有一条以多可用区数据库集群的 VPC 安全组作为源的出站规则，则无法使用该安全组。如果已修改了安全组，也不能使用它。	
有一个或多个安全组与多可用区数据库集群（其名称与模式 `rds-ec2-`*n* 相匹配）关联。尚未修改与此模式匹配的安全组。该安全组只具有一条以 EC2 实例的 VPC 安全组作为源的入站规则。	有一个或多个安全组与 EC2 实例关联，此实例的名称与模式 `ec2-rds-`*n* 相匹配。但是，这些安全组都不能用于与多可用区数据库集群的连接。如果安全组没有一条以多可用区数据库集群的 VPC 安全组作为源的出站规则，则无法使用该安全组。如果已修改了安全组，也不能使用它。	RDS action: create new security groups
有一个或多个安全组与多可用区数据库集群（其名称与模式 `rds-ec2-`*n* 相匹配）关联。尚未修改与此模式匹配的安全组。该安全组只具有一条以 EC2 实例的 VPC 安全组作为源的入站规则。	存在用于连接的有效 EC2 安全组，但它与 EC2 实例不关联。此安全组的名称与模式 `rds-ec2-`*n* 相匹配。尚未修改它。它只具有一条以多可用区数据库集群的 VPC 安全组作为源的出站规则。	RDS action: associate EC2 security group

当前 RDS 安全组配置	当前 EC2 安全组配置	RDS 操作
以下任一条件适用： • 没有与多可用区数据库集群（其名称与模式 `rds-ec2-n` 匹配）关联的安全组。 • 有一个或多个安全组与多可用区数据库集群（其名称与模式 `rds-ec2-n` 相匹配）关联。但是，这些安全组都不能用于与 EC2 实例的连接。如果安全组没有一条以 EC2 实例的 VPC 安全组作为源的入站规则，则无法使用该安全组。如果已修改了安全组，也不能使用它。	有一个或多个安全组与 EC2 实例关联，此实例的名称与模式 `rds-ec2-n` 相匹配。尚未修改与此模式匹配的安全组。该安全组只具有一条以多可用区数据库集群的 VPC 安全组作为源的出站规则。	[RDS action: create new security groups](#)

RDS 操作：创建新的安全组

Amazon RDS 执行以下操作：

- 创建与模式 `rds-ec2-n` 匹配的新安全组。该安全组具有一条以 EC2 实例的 VPC 安全组作为源的入站规则。该安全组与多可用区数据库集群关联，并允许 EC2 实例访问多可用区数据库集群。
- 创建与模式 `ec2-rds-n` 匹配的新安全组。该安全组具有一条以多可用区数据库集群的 VPC 安全组作为源的出站规则。该安全组与 EC2 实例相关联，并允许 EC2 实例向多可用区数据库集群发送流量。

RDS 操作：关联 EC2 安全组

Amazon RDS 将有效的现有 EC2 安全组与 EC2 实例关联。该安全组允许 EC2 实例向多可用区数据库集群发送流量。

自动连接 EC2 实例和多可用区数据库集群

在 EC2 实例与 RDS 数据库之间设置连接之前，请确保满足与 EC2 实例的自动连接概述中所述的要求。

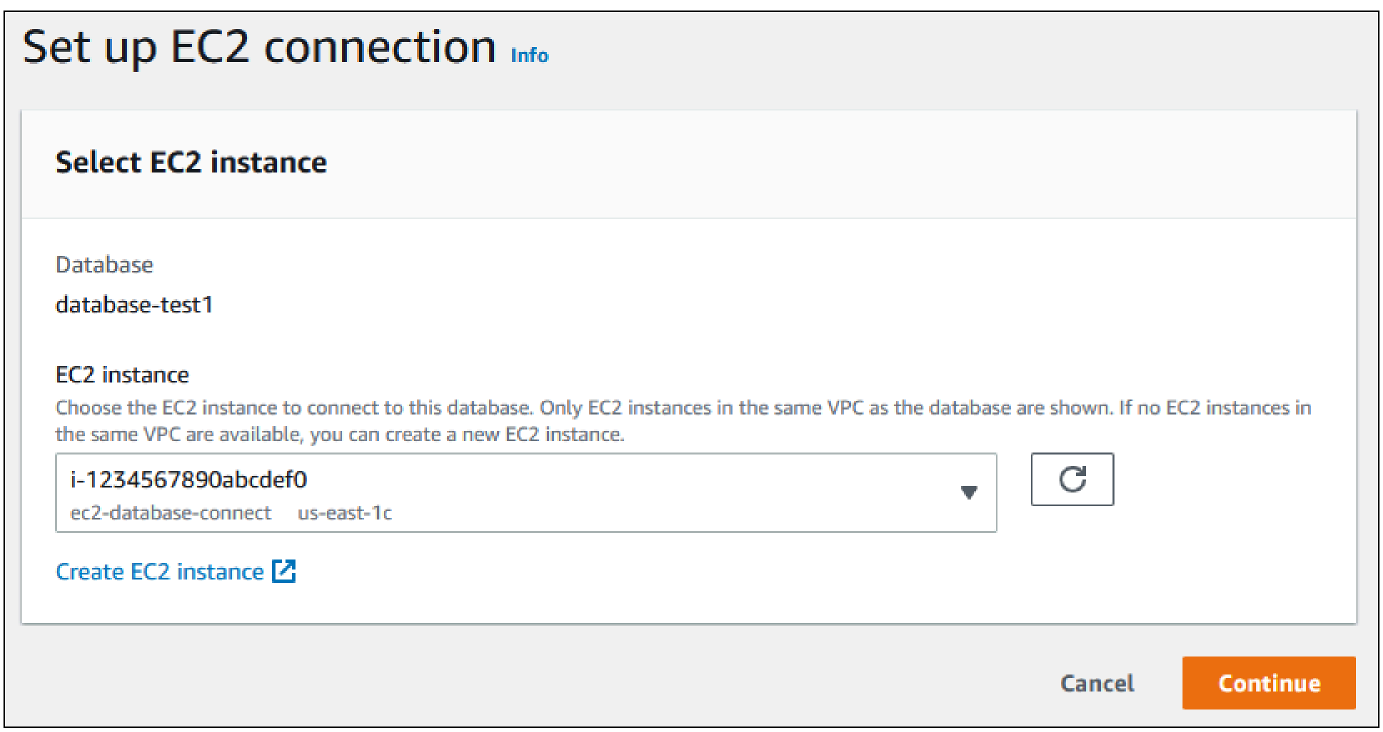
如果您在配置连接后更改安全组，则这些更改可能会影响 EC2 实例与 RDS 数据库之间的连接。

> **Note**
>
> 您只能使用 Amazon Web Services Management Console自动在 EC2 实例与 RDS 数据库之间设置连接。您无法使用 Amazon CLI 或 RDS API 自动设置连接。

自动连接 EC2 实例与 RDS 数据库

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases（数据库），然后选择 RDS 数据库。
3. 从操作中，选择设置 EC2 连接。

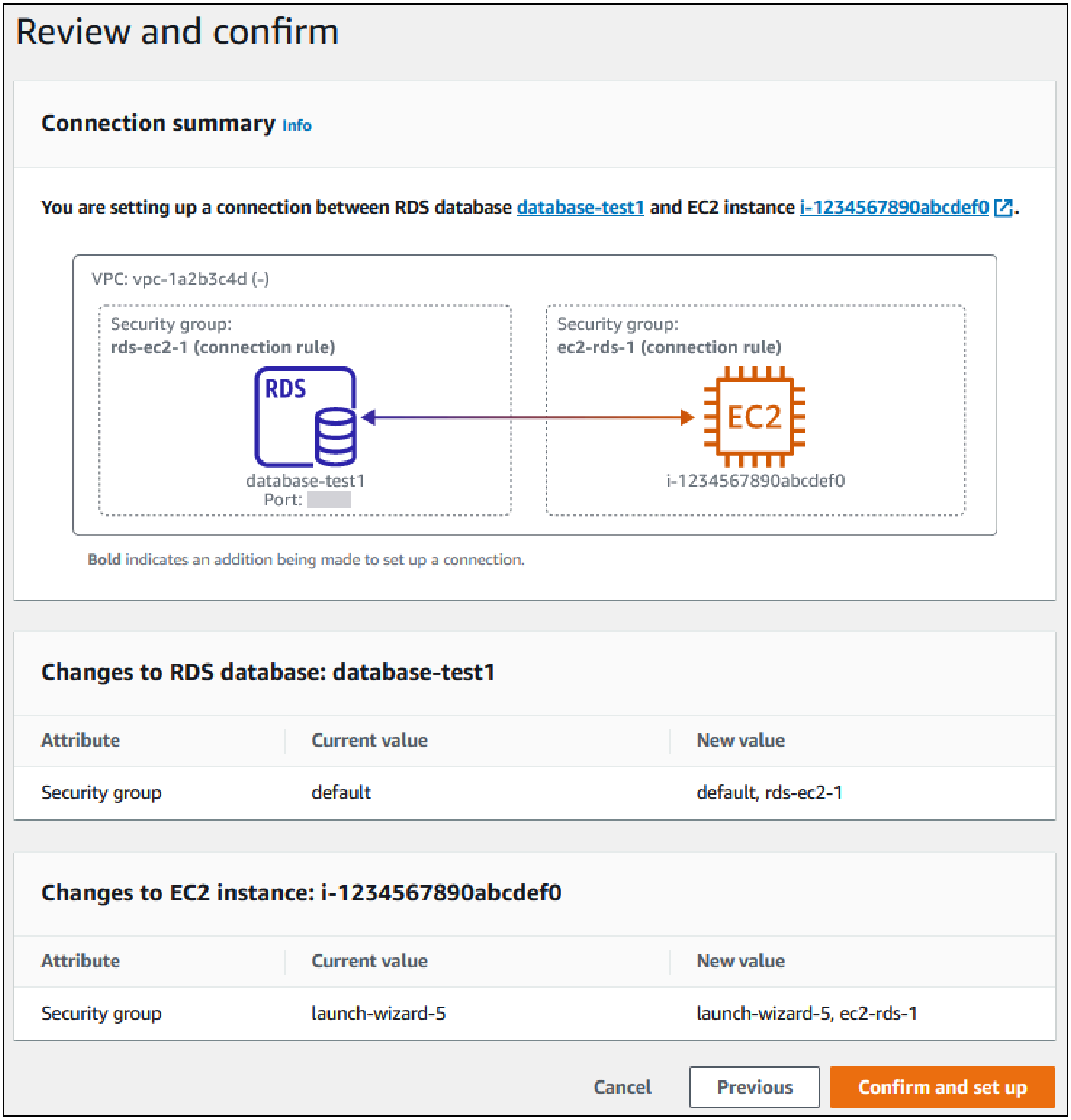
   将出现 Set up EC2 connection（设置 EC2 连接）页面。
4. 在 Set up EC2 connection（设置 EC2 连接）页上，选择 EC2 实例。



如果同一 VPC 中不存在任何 EC2 实例，请选择 Create EC2 instance（创建 EC2 实例）来创建一个此类实例。在这种情况下，请确保新的 EC2 实例与 RDS 数据库位于同一 VPC 中。

5. 选择继续。

将出现 Review and confirm（检查并确认）页面。



6. 在 Review and confirm（检查并确认）页面上，检查 RDS 为设置与 EC2 实例的连接而将进行的更改。

   如果更改正确，请选择确认并设置。

   如果更改不正确，请选择 Previous（上一步）或 Cancel（取消）。

查看连接的计算资源

您可以使用 Amazon Web Services Management Console查看连接到 RDS 数据库的计算资源。显示的资源包括自动设置的计算资源连接。您可以通过以下方式自动设置与计算资源的连接：

- 您可以在创建数据库时选择计算资源。

  有关更多信息，请参阅创建 Amazon RDS 数据库实例 和创建 Amazon RDS 的多可用区数据库集群。
- 您可以在现有数据库和计算资源之间设置连接。

  有关更多信息，请参阅 自动连接 EC2 实例和 RDS 数据库。

列出的计算资源不包括手动连接到数据库的计算资源。例如，您可以通过向与数据库关联的 VPC 安全组添加规则来允许计算资源手动访问数据库。

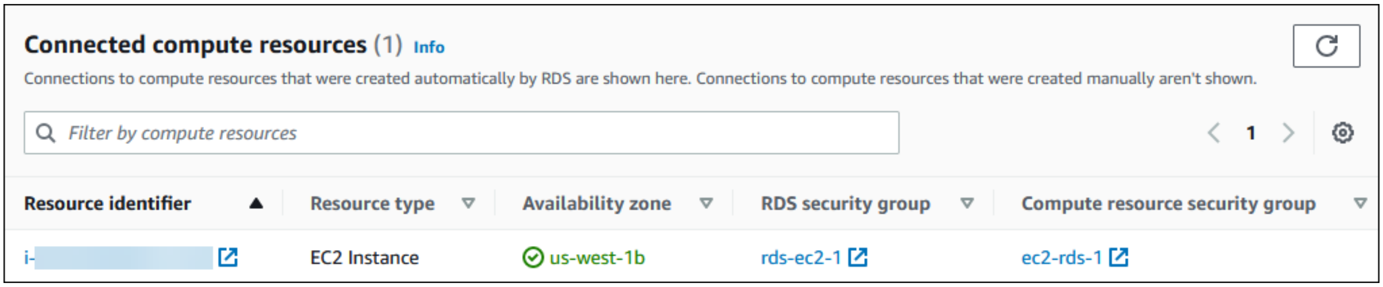
要列出计算资源，必须满足以下条件：

- 与计算资源关联的安全组的名称与模式 `ec2-rds-n` 相匹配（其中 *n* 是数字）。
- 与计算资源关联的安全组具有出站规则，其端口范围设置为 RDS 数据库使用的端口。
- 与计算资源关联的安全组具有出站规则，源设置为与 RDS 数据库关联的安全组。
- 与 RDS 数据库关联的安全组的名称与模式 `rds-ec2-n`（其中 *n* 是数字）相匹配。
- 与 RDS 数据库关联的安全组具有入站规则，其端口范围设置为 RDS 数据库使用的端口。
- 与 RDS 数据库关联的安全组有一条入站规则，其源设置为与计算资源关联的安全组。

查看连接到 RDS 数据库的计算资源

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases（数据库），然后选择 RDS 数据库的名称。

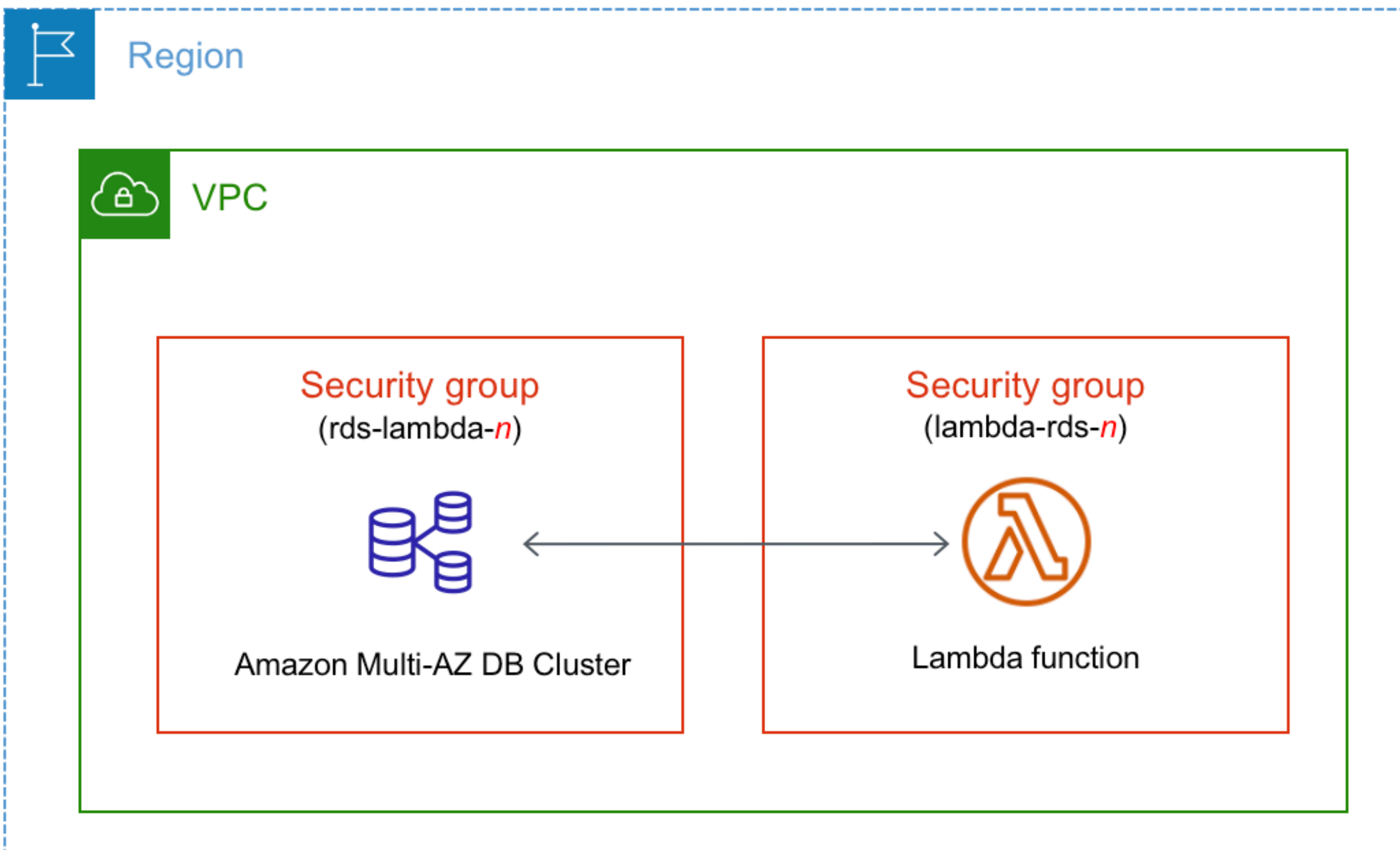
3. 在 Connectivity & security（连接和安全）选项卡上，在 Connected compute resources（连接的计算资源）中查看计算资源。



自动连接 Lambda 函数和多可用区数据库集群

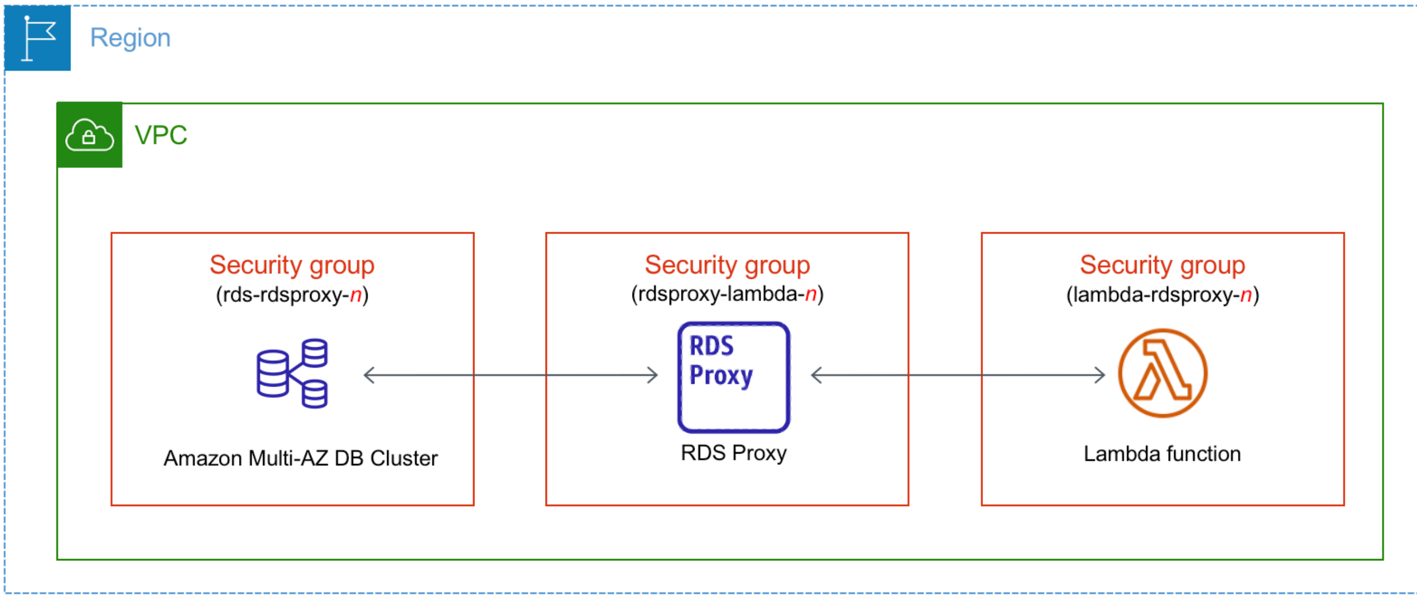
您可以使用 RDS 控制台来简化在 Lambda 函数和多可用区数据库集群之间设置连接的过程。您可以使用 RDS 控制台来简化在 Lambda 函数和多可用区数据库集群之间设置连接的过程。通常，多可用区数据库集群位于 VPC 内的私有子网中。应用程序可以使用 Lambda 函数访问您的私有多可用区数据库集群。

下图显示了您的多可用区数据库集群与 Lambda 函数之间的直接连接。



您可以通过 RDS 代理在 Lambda 函数和数据库之间设置连接，以提高数据库性能和弹性。通常，Lambda 函数会进行频繁的短期数据库连接，此类连接受益于 RDS 代理提供的连接池。您可以利用已为 Lambda 函数提供的任何 IAM 身份验证，而不是在 Lambda 应用程序代码中管理数据库凭证。有关更多信息，请参阅 [的 Amazon RDS 代理]。

您可以使用控制台为您的连接自动创建代理。您也可以选择现有代理。控制台更新代理安全组，以允许来自您的数据库和 Lambda 函数的连接。您可以输入数据库凭证或选择访问数据库所需的 Secrets Manager 密钥。



主题

- 与 Lambda 函数的自动连接概述
- 自动连接 Lambda 函数和多可用区数据库集群
- 查看连接的计算资源

与 Lambda 函数的自动连接概述

当您自动在 Lambda 函数和多可用区数据库集群之间设置连接时，Amazon RDS 会为您的 Lambda 函数和数据库集群配置 VPC 安全组。

以下是将 Lambda 函数与多可用区数据库集群连接的要求：

- Lambda 函数必须与多可用区数据库集群存在于同一 VPC 中。

  如果同一 VPC 中不存在任何 Lambda 函数，控制台将提供创建一个此类函数的链接。

- 设置连接的用户必须具有执行以下 Amazon RDS、Amazon EC2、Lambda、Secrets Manager 和 IAM 操作的权限：
  - Amazon RDS
    - `rds:CreateDBProxies`
    - `rds:DescribeDBInstances`
    - `rds:DescribeDBProxies`
    - `rds:ModifyDBInstance`
    - `rds:ModifyDBProxy`
    - `rds:RegisterProxyTargets`
  - Amazon EC2
    - `ec2:AuthorizeSecurityGroupEgress`
    - `ec2:AuthorizeSecurityGroupIngress`
    - `ec2:CreateSecurityGroup`
    - `ec2:DeleteSecurityGroup`
    - `ec2:DescribeSecurityGroups`
    - `ec2:RevokeSecurityGroupEgress`
    - `ec2:RevokeSecurityGroupIngress`
  - Lambda
    - `lambda:CreateFunctions`
    - `lambda:ListFunctions`
    - `lambda:UpdateFunctionConfiguration`
  - Secrets Manager
    - `sercetsmanager:CreateSecret`
    - `secretsmanager:DescribeSecret`
  - IAM
    - `iam:AttachPolicy`
    - `iam:CreateRole`
    - `iam:CreatePolicy`
  - Amazon KMS
    - `kms:describeKey`

当您在 Lambda 函数和多可用区数据库集群之间设置连接时，Amazon RDS 会为您的函数和多可用区数据库集群配置 VPC 安全组。如果您使用 RDS 代理，则 Amazon RDS 还会为代理配置 VPC 安全组。Amazon RDS 根据与多可用区数据库集群、Lambda 函数和代理关联的安全组的当前配置执行操作，如下表所述。

当前 RDS 安全组配置	当前 Lambda 安全组配置	当前代理安全组配置	RDS 操作
Amazon RDS 不采取任何措施，因为所有资源的安全组都遵循正确的命名模式并具有正确的入站和出站规则。	有一个或多个安全组与多可用区数据库集群 [其名称与模式 `rds-lambda-` *n*（其中 *n* 是数字）相匹配，或关联代理的 `TargetHealth` 是 `AVAILABLE`] 关联。 尚未修改与此模式匹配的安全组。该安全组只具有一条以 Lambda 函数或代理的 VPC 安全组作为源的入站规则。	有一个或多个安全组与 Lambda 函数关联，此函数的名称与模式 `lambda-rds-` *n* 或 `lambda-rdsproxy-` *n*（其中 *n* 是数字）相匹配。 尚未修改与此模式匹配的安全组。此安全组只有一条出站规则，此规则以多可用区数据库集群或代理的 VPC 安全组作为目标。	有一个或多个安全组与代理关联，此代理的名称与模式 `rdsproxy-lambda-` *n*（其中 *n* 是数字）相匹配。 尚未修改与此模式匹配的安全组。此安全组具有入站和出站规则，这些规则具有 Lambda 函数和多可用区数据库集群的 VPC 安全组。
以下任一条件适用： • 没有与多可用区数据库集群（名称与模式 `rds-lambda-` *n* 匹配或关联代理的 `TargetHealth` 为 `AVAILABLE` ）相关联的安全组。 •	以下任一条件适用： • 没有与 Lambda 函数（名称与模式 `lambda-rds-` *n* 或 `lambda-rdsproxy-` *n* 匹配）关联的安全组。 • 有一个或多个安全组与 Lambda 函数关联，此函	以下任一条件适用： • 没有与代理（名称与模式 `rdsproxy-lambda-` *n* 匹配）关联的安全组。 • 有一个或多个安全组与代理关联，此代理的名称与 `rdsproxy-lambda-` *n* 相匹	[RDS action: create new security groups](#)

当前 RDS 安全组配置	当前 Lambda 安全组配置	当前代理安全组配置	RDS 操作
有一个或多个与多可用区数据库集群（名称与模式 `rds-lambda-` *n* 匹配或关联代理的 `TargetHealth` 为 `AVAILABLE` ）相关联的安全组。但是，Amazon RDS 不能使用这些安全组中的任何一个来连接 Lambda 函数。 如果安全组没有一条以 Lambda 函数或代理的 VPC 安全组作为源的入站规则，则 Amazon RDS 无法使用该安全组。Amazon RDS 也无法使用经过修改的安全组。修改示例包括添加规则或更改现有规则的端口。	数的名称与模式 `lambda-rds-` *n* 或 `lambda-rdsproxy-` *n* 相匹配。但是，Amazon RDS 不能使用这些安全组中的任何一个来连接多可用区数据库集群。 如果安全组没有一条以多可用区数据库集群或代理的 VPC 安全组作为源的出站规则，则 Amazon RDS 无法使用该安全组。Amazon RDS 也无法使用经过修改的安全组。	配。但是，Amazon RDS 不能使用这些安全组中的任何一个来连接多可用区数据库集群或 Lambda 函数。 如果安全组没有入站和出站规则（这些规则具有多可用区数据库集群和 Lambda 函数的 VPC 安全组），则 Amazon RDS 无法使用该安全组。Amazon RDS 也无法使用经过修改的安全组。	

当前 RDS 安全组配置	当前 Lambda 安全组配置	当前代理安全组配置	RDS 操作
有一个或多个与多可用区数据库集群（名称与模式 `rds-lambda-` *n* 匹配或关联代理的 `TargetHealth` 为 `AVAILABLE`）相关联的安全组。 尚未修改与此模式匹配的安全组。该安全组只具有一条以 Lambda 函数或代理的 VPC 安全组作为源的入站规则。	有一个或多个安全组与 Lambda 函数关联，此函数的名称与模式 `lambda-rds-` *n* 或 `lambda-rdsproxy-` *n* 相匹配。 但是，Amazon RDS 不能使用这些安全组中的任何一个来连接多可用区数据库集群。如果安全组没有一条以多可用区数据库集群或代理的 VPC 安全组作为目标的出站规则，则 Amazon RDS 无法使用该安全组。Amazon RDS 也无法使用经过修改的安全组。	有一个或多个安全组与代理关联，此代理的名称与模式 `rdsproxy-lambda-` *n* 相匹配。 但是，Amazon RDS 不能使用这些安全组中的任何一个来连接多可用区数据库集群或 Lambda 函数。如果安全组没有入站和出站规则（这些规则具有多可用区数据库集群和 Lambda 函数的 VPC 安全组），则 Amazon RDS 无法使用该安全组。Amazon RDS 也无法使用经过修改的安全组。	RDS action: create new security groups

当前 RDS 安全组配置	当前 Lambda 安全组配置	当前代理安全组配置	RDS 操作
有一个或多个与多可用区数据库集群（名称与模式 `rds-lambda-` *n* 匹配或关联代理的 `TargetHealth` 为 `AVAILABLE` ）相关联的安全组。 尚未修改与此模式匹配的安全组。该安全组只具有一条以 Lambda 函数或代理的 VPC 安全组作为源的入站规则。	存在用于连接的有效 Lambda 安全组，但它与 Lambda 函数不关联。此安全组的名称与模式 `lambda-rds-` *n* 或 `lambda-rdsproxy-` *n* 相匹配。尚未修改它。它只有一条出站规则，此规则以多可用区数据库集群或代理的 VPC 安全组作为目标。	存在用于连接的有效代理安全组，但它与代理不关联。此安全组的名称与模式 `rdsproxy-lambda-` *n* 相匹配。尚未修改它。它具有入站和出站规则，这些规则具有多可用区数据库集群和 Lambda 函数的 VPC 安全组。	[RDS action: associate Lambda security group](#)

当前 RDS 安全组配置	当前 Lambda 安全组配置	当前代理安全组配置	RDS 操作
以下任一条件适用： • 没有与多可用区数据库集群（名称与模式 `rds-lambda-` *n* 匹配或关联代理的 `TargetHealth` 为 `AVAILABLE` ）相关联的安全组。 • 有一个或多个与多可用区数据库集群（名称与模式 `rds-lambda-` *n* 匹配或关联代理的 `TargetHealth` 为 `AVAILABLE` ）相关联的安全组。但是，Amazon RDS 不能使用这些安全组中的任何一个来连接 Lambda 函数或代理。 如果安全组没有一条以 Lambda 函数或代理的 VPC 安全组作为源的入站规则，则 Amazon RDS 无法使用该安全组。Amazon	有一个或多个安全组与 Lambda 函数关联，此函数的名称与模式 `lambda-rds-` *n* 或 `lambda-rdsproxy-` *n* 相匹配。 尚未修改与此模式匹配的安全组。此安全组只有一条出站规则，此规则以多可用区数据库集群或代理的 VPC 安全组作为目标。	有一个或多个安全组与代理关联，此代理的名称与模式 `rdsproxy-lambda-` *n* 相匹配。 尚未修改与此模式匹配的安全组。此安全组具有入站和出站规则，这些规则具有多可用区数据库集群和 Lambda 函数的 VPC 安全组。	[RDS action: create new security groups](#)

当前 RDS 安全组配置	当前 Lambda 安全组配置	当前代理安全组配置	RDS 操作
RDS 也无法使用经过修改的安全组。			

当前 RDS 安全组配置	当前 Lambda 安全组配置	当前代理安全组配置	RDS 操作
有一个或多个安全组与多可用区数据库集群 [其名称与模式 `rds-rdsproxy-` *n*（其中 *n* 是数字）相匹配] 关联。	以下任一条件适用： • 没有与 Lambda 函数（名称与模式 `lambda-rds-` *n* 或 `lambda-rdsproxy-` *n* 匹配）关联的安全组。 • 有一个或多个安全组与 Lambda 函数关联，此函数的名称与模式 `lambda-rds-` *n* 或 `lambda-rdsproxy-` *n* 相匹配。但是，Amazon RDS 不能使用这些安全组中的任何一个来连接多可用区数据库集群。 如果安全组没有一条以多可用区数据库集群或代理的 VPC 安全组作为目标的出站规则，则 Amazon RDS 无法使用该安全组。Amazon RDS 也无法使用经过修改的安全组。	以下任一条件适用： • 没有与代理（名称与模式 `rdsproxy-lambda-` *n* 匹配）关联的安全组。 • 有一个或多个安全组与代理关联，此代理的名称与 `rdsproxy-lambda-` *n* 相匹配。但是，Amazon RDS 不能使用这些安全组中的任何一个来连接多可用区数据库集群或 Lambda 函数。 如果安全组没有入站和出站规则（这些规则具有多可用区数据库集群和 Lambda 函数的 VPC 安全组），则 Amazon RDS 无法使用该安全组。Amazon RDS 也无法使用经过修改的安全组。	[RDS action: create new security groups](#)

RDS 操作：创建新的安全组

Amazon RDS 执行以下操作：

- 创建与模式 `rds-lambda-n` 匹配的新安全组。该安全组具有一条以 Lambda 函数或代理的 VPC 安全组作为源的入站规则。该安全组与多可用区数据库集群关联，并允许函数或代理访问多可用区数据库集群。
- 创建与模式 `lambda-rds-n` 匹配的新安全组。此安全组具有一条出站规则，此规则以多可用区数据库集群或代理的 VPC 安全组作为目标。此安全组与 Lambda 函数相关联，并允许 Lambda 函数将流量发送到多可用区数据库集群或者通过代理发送流量。
- 创建与模式 `rdsproxy-lambda-n` 匹配的新安全组。此安全组具有入站和出站规则，这些规则具有多可用区数据库集群和 Lambda 函数的 VPC 安全组。

RDS 操作：关联 Lambda 安全组

Amazon RDS 将有效的现有的 Lambda 安全组与 Lambda 函数关联。此安全组允许该函数向多可用区数据库集群发送流量或通过代理发送流量。

自动连接 Lambda 函数和多可用区数据库集群

您可以使用 Amazon RDS 控制台将 Lambda 函数自动连接到您的多可用区数据库集群。这简化了在这些资源之间设置连接的过程。

您也可以使用 RDS 代理在连接中包含代理。Lambda 函数会进行频繁的短期数据库连接，此类连接受益于 RDS 代理提供的连接池。您还可以使用已经为 Lambda 函数设置的任何 IAM 身份验证，而不是在 Lambda 应用程序代码中管理数据库凭证。

您可以使用设置 Lambda 连接页面将现有的多可用区数据库集群连接到新的和现有 Lambda 函数。设置过程会自动为您设置所需的安全组。

在 Lambda 函数和多可用区数据库集群之间设置连接之前，请确保：

- 您的 Lambda 函数和多可用区数据库集群位于同一 VPC 中。
- 您拥有用户账户的相应权限。有关要求的更多信息，请参阅与 Lambda 函数的自动连接概述。

如果您在配置连接后更改安全组，则这些更改可能会影响 Lambda 函数与多可用区数据库集群之间的连接。

> **Note**
>
> 您只能在 Amazon Web Services Management Console中自动设置多可用区数据库集群和 Lambda 函数之间的连接。要连接 Lambda 函数，多可用区数据库集群中的所有实例都必须处于可用状态。

自动连接 Lambda 函数和多可用区数据库集群

<result>

确认设置后，Amazon RDS 开始连接您的 Lambda 函数、RDS 代理（如果您使用了代理）和多可用区数据库集群的过程。控制台显示连接详细信息对话框，其中列出了允许在您的资源之间进行连接的安全组更改。

</result>

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择数据库，然后选择要连接到 Lambda 函数的多可用区数据库集群。
3. 对于操作，选择设置 Lambda 连接。
4. 在设置 Lambda 连接页面上，在选择 Lambda 函数下方，执行以下任一操作：
   - 如果您在与多可用区数据库集群相同的 VPC 中有一个现有的 Lambda 函数，请选择选择现有函数，然后选择该函数。
   - 如果您在同一 VPC 中没有 Lambda 函数，请选择创建新函数，然后输入函数名称。原定设置运行时系统设置为 Nodejs.18。完成连接设置后，您可以在 Lambda 控制台中修改新 Lambda 函数的设置。
5. （可选）在 RDS 代理下，选择使用 RDS 代理进行连接，然后执行以下任一操作：
   - 如果您有要使用的现有代理，请选择选择现有代理，然后选择此代理。
   - 如果您没有代理，并且希望 Amazon RDS 自动为您创建代理，请选择创建新代理。然后，对于数据库凭证，执行以下任一操作：

     a. 选择数据库用户名和密码，然后输入多可用区数据库集群的用户名和密码。

     b. 选择 Secrets Manager 密钥。然后，对于选择密钥，选择一个 Amazon Secrets Manager 密钥。如果您没有 Secrets Manager 密钥，请选择创建新的 Secrets Manager 密钥以创建新密钥。创建密钥后，对于选择密钥，选择新密钥。

创建新代理后，选择选择现有代理，然后选择该代理。请注意，您的代理可能需要一些时间才能用于连接。

6. （可选）扩展连接摘要并验证您的资源的突出显示更新。
7. 选择 Set up (设置)。

查看连接的计算资源

您可以使用 Amazon Web Services Management Console查看连接到多可用区数据库集群的计算资源。显示的资源包括 Amazon RDS 自动设置的计算资源连接。

列出的计算资源不包括那些手动连接到多可用区数据库集群的计算资源。例如，您可以通过向与多可用区数据库集群关联的 VPC 安全组添加规则，来允许计算资源手动访问该集群。

要使控制台列出 Lambda 函数，必须满足以下条件：

- 与计算资源关联的安全组的名称与模式 `lambda-rds-n` 或 `lambda-rdsproxy-n`（其中 *n* 是数字）相匹配。
- 与计算资源关联的安全组具有出站规则，其端口范围设置为多可用区数据库集群或关联代理的端口。出站规则的目标必须设置为与多可用区数据库集群或关联代理关联的安全组。
- 连接到与您数据库关联的代理的安全组名称与模式 `rds-rdsproxy-n`（其中 *n* 是一个数字）相匹配。
- 与该函数关联的安全组有一条出站规则，其端口设置为多可用区数据库集群或关联代理使用的端口。目标必须设置为与多可用区数据库集群或关联代理关联的安全组。

查看自动连接到多可用区数据库集群的计算资源

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择数据库，然后选择多可用区数据库集群。
3. 在连接和安全选项卡上，在连接的计算资源下查看计算资源。

修改 Amazon RDS 的多可用区数据库集群

多可用区数据库集群在三个独立可用区中有一个写入器数据库实例和两个读取器数据库实例。与多可用区部署相比，多可用区数据库集群可提供高可用性、增加读取工作负载容量以及更低的延迟。有关多可用区数据库集群的更多信息，请参阅 Amazon RDS 的多可用区数据库集群部署。

您可以修改多可用区数据库集群以更改其设置。您还可以对多可用区数据库集群执行操作，例如获取其快照。

> **Important**
>
> 您无法修改多可用区数据库集群中的数据库实例。所有修改都必须在数据库集群级别完成。您可以对多可用区数据库集群中的数据库实例执行的唯一操作是重启它。

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 修改多可用区数据库集群。

控制台

要修改多可用区数据库集群

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases（数据库），然后选择要修改的多可用区数据库集群。
3. 选择修改。此时会显示修改数据库集群页面。
4. 根据需要更改任意设置。有关每项设置的信息，请参阅 修改多可用区数据库集群的设置。
5. 当所有更改都达到您的要求时，选择继续并查看修改摘要。
6. （可选）选择立即应用以立即应用更改。选择此选项在某些情况下可能导致停机。有关更多信息，请参阅 立即应用更改。
7. 在确认页面上，检查您的更改。如果更改正确无误，请选择 Modify DB cluster（修改数据库集群）以保存更改。

   也可以选择 Back (返回) 编辑您的更改，或选择 Cancel (取消) 取消更改。

Amazon CLI

要使用 Amazon CLI 修改多可用区数据库集群，请调用 modify-db-cluster 命令。指定数据库集群标识符以及要修改的选项值。有关各选项的信息，请参阅修改多可用区数据库集群的设置。

Example

以下代码修改 `my-multi-az-dbcluster`，将备份保留期设置为 1 周 (7 天)。该代码使用 `--deletion-protection` 启用删除保护。要关闭删除保护，请使用 `--no-deletion-protection`。将在下一维护时段使用 `--no-apply-immediately` 应用这些更改。使用 `--apply-immediately` 可立即应用更改。有关更多信息，请参阅 立即应用更改。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-cluster \
    --db-cluster-identifier my-multi-az-dbcluster \
    --backup-retention-period 7 \
    --deletion-protection \
    --no-apply-immediately
```

对于 Windows：

```
aws rds modify-db-cluster ^
    --db-cluster-identifier my-multi-az-dbcluster ^
    --backup-retention-period 7 ^
    --deletion-protection ^
    --no-apply-immediately
```

RDS API

要使用 Amazon RDS API 修改多可用区数据库集群，请调用 ModifyDBCluster 操作。指定数据库集群标识符，以及您要修改的设置参数。有关每个参数的信息，请参阅修改多可用区数据库集群的设置。

立即应用更改

修改多可用区数据库集群后，您可立即应用更改。要立即应用更改，请选中 Amazon Web Services Management Console中的立即应用选项。或者，您可以在调用 Amazon CLI 时使用 `--apply-immediately` 选项，或在使用 Amazon RDS API 时将 `ApplyImmediately` 参数设置为 `true`。

如果您没有选择立即应用更改，更改将被放置在等待修改队列中。队列中的所有等待更改都将在下一维护时段应用。如果您选择立即应用更改，将应用您的新更改以及等待修改队列中的所有更改。

> **Important**
>
> 如果任何待处理修改需要数据库集群暂时不可用（停机），则选择立即应用选项可能会导致意外停机。
>
> 当您选择立即应用更改时，任何待定修改也将立即应用，而不是在下一个维护时段内应用。
>
> 如果您不想将待定更改应用到下一个维护时段，您可以修改数据库实例来恢复更改。您可以通过使用 Amazon CLI 和指定 `--apply-immediately` 选项来执行此操作。

一些数据库设置的更改会立即应用，即使您选择延迟更改也是如此。要查看不同数据库的设置如何与“立即应用”设置进行交互，请参阅修改多可用区数据库集群的设置。

修改多可用区数据库集群的设置

有关可用于修改多可用区数据库集群的设置的详细信息，请参见下表。有关 Amazon CLI 选项的更多信息，请参阅 modify-db-cluster。有关 RDS API 参数的更多信息，请参阅 ModifyDBCluster。

控制台设置	设置说明	CLI 选项和 RDS API 参数	发生更改时	停机说明

分配的存储空间	为数据库集群中的每个数据库实例分配的存储量（以 GiB 为单位）。有关更多信息，请参阅 Amazon RDS 数据库实例存储。	CLI 选项：
		
		`--allocated-storage`
		
		RDS API 参数：
		
		`AllocatedStorage`
			如果您选择立即应用更改，则更改立即生效。
			
			如果您不选择立即应用更改，则更改会在下一个维护时段中生效。
				在此更改期间，不会出现停机。

自动次要版本升级	请选择 Enable auto minor version upgrade（启用自动次要版本升级），以使数据库集群能够在首选次要数据库引擎版本升级可用	CLI 选项：
		
		`--auto-minor-version-upgrade`
		
		`--no-auto-minor-version-upgrade`
			更改立即发生。此设置忽略立即应用设置。	在此更改期间发生停机。

控制台设置	设置说明	CLI 选项和 RDS API 参数	发生更改时	停机说明
	时自动接收这些升级。Amazon RDS 在维护时段内执行自动次要版本升级。	RDS API 参数： `AutoMinorVersionUpgrade`		

备份保留期	保留数据库集群的自动备份的天数。该值必须大于零。
	
	有关更多信息，请参阅 备份简介。
		CLI 选项：
		
		`--backup-retention-period`
		
		RDS API 参数：
		
		`BackupRetentionPeriod`
			如果您选择立即应用更改，则更改立即生效。
			
			如果您不选择立即应用更改，则更改会在下一个维护时段中生效。
				在此更改期间，不会出现停机。

备份时段	Amazon RDS 自动备份数据库集群的时间段。除非您知道需要何时进行数据库备份，否则，请使用 No preference（无首选项）的默认设置。
	
	有关更多信息，请参阅“备份简介”。
		CLI 选项：
		
		`--preferred-backup-window`
		
		RDS API 参数：
		
		`PreferredBackupWindow`
			更改会尽快异步应用。	在此更改期间，不会停机。

控制台设置	设置说明	CLI 选项和 RDS API 参数	发生更改时	停机说明

证书颁发机构	数据库集群使用的服务器证书的证书颁发机构（CA）。
	
	有关更多信息，请参阅 使用 SSL/TLS 加密与数据库实例或集群的连接。
		CLI 选项：
		
		`--ca-certificate-identifier`
		
		RDS API 参数：
		
		`CACertificateIdentifier`
			如果您选择立即应用更改，则更改立即生效。
			
			如果您不选择立即应用更改，则更改会在下一个维护时段中生效。
				仅当数据库引擎不支持在不重新启动的情况下轮换时，才会发生停机。您可以使用 describe-db-engine-versions Amazon CLI 命令来确定数据库引擎是否支持在不重新启动的情况下轮换。

将标签复制到快照	在创建快照时，该选项将任何数据库集群标签复制到数据库快照中。
	
	有关更多信息，请参阅 为 Amazon RDS 资源添加标签。
		CLI 选项：
		
		`-copy-tags-to-snapshot`
		
		`-no-copy-tags-to-snapshot`
		
		RDS API 参数：
		
		`CopyTagsToSnapshot`
			更改立即发生。此设置忽略“立即应用”设置。	在此更改期间，不会停机。

数据库身份验证	对于多可用区数据库集群，仅支持 Password authentication（密码身份验证）。	无，因为密码身份验证是原定设置。	如果您选择立即应用更改，则更改立即生效。 如果您不选择立即应用更改，则更改会在下一个维护时段中生效。	在此更改期间，不会出现停机。

控制台设置	设置说明	CLI 选项和 RDS API 参数	发生更改时	停机说明

数据库集群标识符	数据库集群标识符。此值以一个小写字符串存储。
	
	更改数据库集群标识符后，数据库集群端点发生变化。数据库集群中的数据库实例的标识符和端点也会发生变化。新的数据库集群名称必须是唯一的。最大长度为 63 个字符。
	
	数据库集群中的数据库实例的名称将更改为与数据库集群的新名称相对应。新的数据库实例名称不能与现有数据库实例的名称相同。例如，如果您将数据库集群名称更改为 `maz`，则数据库实例名称可能会更改为 `maz-instance-1` 。在这种情况下，不能存在名为 `maz-instance-1` 的现有数据库实例。
		CLI 选项：
		
		`--new-db-cluster-identifier`
		
		RDS API 参数：
		
		`NewDBClusterIdentifier`
			如果您选择立即应用更改，则更改立即生效。
			
			如果您不选择立即应用更改，则更改会在下一个维护时段中生效。
				在此更改期间，不会出现停机。

控制台设置	设置说明	CLI 选项和 RDS API 参数	发生更改时	停机说明
	有关更多信息，请参阅 重命名 Amazon RDS 的多可用区数据库集群。			

DB cluster instance class（数据库集群实例类）	多可用区数据库集群中各数据库实例（例如 `db.r6gd.xlarge` ）的计算和内存容量。
	
	如有可能，请选择足够大的数据库实例类以便典型查询工作集能够保留在内存中。如果在内存中保留工作集，系统可以避免写入到磁盘，从而提高性能。
	
	有关更多信息，请参阅 the section called “多可用区数据库集群的实例类可用性”。
		CLI 选项：
		
		`--db-cluster-instance-class`
		
		RDS API 参数：
		
		`DBClusterInstanceClass`
			如果您选择立即应用更改，则更改立即生效。
			
			如果您不选择立即应用更改，则更改会在下一个维护时段中生效。
				在此更改期间发生停机。

控制台设置	设置说明	CLI 选项和 RDS API 参数	发生更改时	停机说明

数据库集群参数组	要与数据库集群关联的数据库集群参数组。
	
	有关更多信息，请参阅 多可用区数据库集群的参数组。
		CLI 选项：
		
		`--db-cluster-parameter-group-name`
		
		RDS API 参数：
		
		`DBClusterParameterGroupName`
			参数组更改会立即发生。	在此更改期间，不会停机。当您更改参数组时，对某些参数的更改将立即应用于多可用区数据库集群中的数据库实例，而无需重新启动。而对其他一些参数进行的更改，只有在重新启动数据库实例之后才会应用。

数据库引擎版本	要使用的数据库引擎的版本。	CLI 选项：
		
		`--engine-version`
		
		RDS API 参数：
		
		`EngineVersion`
			如果您选择立即应用更改，则更改立即生效。
			
			如果您不选择立即应用更改，则更改会在下一个维护时段中生效。
				在此更改期间发生停机。

删除保护	启用删除保护以禁止删除数据库集群。
	
	有关更多信息，请参阅 删除数据库实例。
		CLI 选项：
		
		`--deletion-protection`
		
		`--no-deletion-protection`
		
		RDS API 参数：
		
		`DeletionProtection`
			更改立即发生。此设置忽略“立即应用”设置。	在此更改期间，不会停机。

控制台设置	设置说明	CLI 选项和 RDS API 参数	发生更改时	停机说明

维护窗口	30 分钟时段，在这段时间内，将会应用对数据库集群的待处理修改。如果该时段无关紧要，请选择 No preference（无首选项）。
	
	有关更多信息，请参阅“Amazon RDS 维护时段”。
		CLI 选项：
		
		`--preferred-maintenance-window`
		
		RDS API 参数：
		
		`PreferredMaintenanceWindow`
			更改立即发生。此设置忽略“立即应用”设置。	如果有一个或多个待处理的操作导致停机，并且维护时段经过更改，加入了当前时间，则立即应用这些待处理的操作并会出现停机。

控制台设置	设置说明	CLI 选项和 RDS API 参数	发生更改时	停机说明

在 Amazon Secrets Manager 中管理主凭证	选择在 Amazon Secrets Manager 中管理主凭证，以在 Secrets Manager 的密钥中管理主用户密码。
	
	（可选）选择用于保护密钥的 KMS 密钥。请从您的账户的 KMS 密钥中进行选择，或输入来自其他账户的密钥。
	
	如果 RDS 已在管理数据库集群的主用户密码，则可以通过选择 Rotate secret immediately（立即轮换密钥）来轮换主用户密码。
	
	有关更多信息，请参阅 [使用 Amazon RDS 和 Amazon Secrets Manager 管理密码](#)。
		CLI 选项：
		
		`--manage-master-user-password | --no-manage-master-user-password`
		
		`--master-user-secret-kms-key-id`
		
		`--rotate-master-user-password | --no-rotate-master-user-password`
		
		RDS API 参数：
		
		`ManageMasterUserPassword`
		
		`MasterUserSecretKmsKeyId`
		
		`RotateMasterUserPassword`
			如果您开启或关闭自动主用户密码管理，则更改会立即发生。此更改忽略立即应用设置。
			
			如果要轮换主用户密码，则必须指定立即应用更改。
				在此更改期间，不会停机。

控制台设置	设置说明	CLI 选项和 RDS API 参数	发生更改时	停机说明

新建主密码	您的主用户账户密码。	CLI 选项：
		`--master-user-password`
		RDS API 参数：
		`MasterUserPassword`
			更改会尽快异步应用。此设置忽略“立即应用”设置。	在此更改期间，不会停机。

预置 IOPS	最初为数据库集群分配的预置 IOPS（每秒输入/输出操作数）量。	CLI 选项：
		`--iops`
		RDS API 参数：
		`Iops`
			如果您选择立即应用更改，则更改立即生效。
			如果您不选择立即应用更改，则更改会在下一个维护时段中生效。
				在此更改期间，不会出现停机。

控制台设置	设置说明	CLI 选项和 RDS API 参数	发生更改时	停机说明
公有访问权限	请选择 Publicly accessible（可公开访问）以为数据库集群提供公有 IP 地址，这表示可以在 Virtual Private Cloud (VPC) 外部访问该实例。要可供公开访问，数据库集群还必须在 VPC 的公有子网中。 请选择 Not publicly accessible（不可公开访问）以设置只能从 VPC 内部访问数据库集群。 有关更多信息，请参阅 [对互联网隐藏 VPC 中的数据库实例]()。 要从其 VPC 外部连接到数据库集群，该数据库集群必须可公开访问。此外，必须使用数据库集群安全组的入站规则授予访问权限，并且必须满足其他要求。	修改数据库集群时不可用。	更改立即发生。此设置忽略“立即应用”设置。	在此更改期间，不会停机。

控制台设置	设置说明	CLI 选项和 RDS API 参数	发生更改时	停机说明
	有关更多信息，请参阅 无法连接到 Amazon RDS 数据库实例。 如果您的数据库集群不可公开访问，则您可以使用 Amazon Site-to-Site VPN 连接或 Amazon Direct Connect 连接从专用网络访问该实例。有关更多信息，请参阅 互联网络流量隐私。			

存储类型	数据库集群的存储类型。
	
	仅支持通用型 SSD（gp3）、预调配 IOPS（io1）和预调配 IOPS SSD（io2）存储。
	
	有关更多信息，请参阅 Amazon RDS 存储类型。
		CLI 选项：
		
		`--storage-type`
		
		RDS API 参数：
		
		`StorageType`
			如果您选择立即应用更改，则更改立即生效。
			
			如果您不选择立即应用更改，则更改会在下一个维护时段中生效。
				在此更改期间，不会出现停机。

控制台设置	设置说明	CLI 选项和 RDS API 参数	发生更改时	停机说明

VPC 安全组	要与数据库集群关联的安全组。
	
	有关更多信息，请参阅 VPC 安全组概述。
		CLI 选项：
		
		`--vpc-security-group-ids`
		
		RDS API 参数：
		
		`VpcSecurityGroupIds`
			更改会尽快异步应用。此设置忽略“立即应用”设置。	在此更改期间，不会停机。

修改多可用区数据库集群时不适用的设置

Amazon CLI 命令 modify-db-cluster 和 RDS API 操作 ModifyDBCluster 中的以下设置不适用于多可用区数据库集群。

您也无法在控制台中为多可用区数据库集群修改这些设置。

Amazon CLI 设置	RDS API 设置
`--backtrack-window`	`BacktrackWindow`
`--cloudwatch-logs-export-configuration`	`CloudwatchLogsExportConfiguration`
`--copy-tags-to-snapshot | --no-copy-tags-to-snapshot`	`CopyTagsToSnapshot`
`--db-instance-parameter-group-name`	`DBInstanceParameterGroupName`
`--domain`	`Domain`
`--domain-iam-role-name`	`DomainIAMRoleName`

Amazon CLI 设置	RDS API 设置
`--enable-global-write-forwarding | --no-enable-global-write-forwarding`	`EnableGlobalWriteForwarding`
`--enable-http-endpoint | --no-enable-http-endpoint`	`EnableHttpEndpoint`
`--option-group-name`	`OptionGroupName`
`--port`	`Port`
`--scaling-configuration`	`ScalingConfiguration`
`--storage-type`	`StorageType`

升级 Amazon RDS 的多可用区数据库集群的引擎版本

Amazon RDS 提供每个支持的数据库引擎的更高版本，因此，您可以使多可用区数据库集群保持最新。本主题说明将多可用区数据库集群升级到更高版本的过程。

升级多可用区数据库集群需要选择新的兼容引擎版本并规划潜在的停机时间。该过程通过利用多可用区架构的失效转移功能，确保将中断降至最低。在升级期间，首先更新主实例，然后失效转移到备用实例以保持可用性。最佳实践包括在低流量期间执行升级、在非生产环境中进行测试以及验证应用程序与新版本的兼容性。

主题

次要版本升级

次要版本升级 仅包含与现有应用程序向后兼容的更改。当您启动次要版本升级时，Amazon RDS 首先一次升级一个读取器数据库实例。然后，其中一个读取器数据库实例将切换为新的写入器数据库实例。然后，Amazon RDS 升级旧的写入器实例（现在是读取器实例）。

升级期间的停机时间限于其中一个读取器数据库实例变为新写入器数据库实例所花费的时间。这一停机时间就像自动失效转移一样。有关更多信息，请参阅 the section called “对多可用区数据库集群执行失效转移”。请注意，多可用区数据库集群的副本滞后可能会影响停机时间。有关更多信息，请参阅 the section called “副本滞后和多可用区数据库集群”。

对于 RDS for PostgreSQL 多可用区数据库集群只读副本，Amazon RDS 一次升级一个集群成员实例。升级期间，读取器和写入器集群角色不会切换。因此，在 Amazon RDS 升级集群写入器实例时，数据库集群可能会出现停机。

> **Note**
>
> 多可用区数据库集群次要版本升级的停机时间通常为 35 秒。与 RDS 代理一起使用时，您可以将停机时间进一步缩短到一秒或更短。有关更多信息，请参阅 Amazon RDS 代理。或者，您可以使用开源数据库代理，例如 ProxySQL、PgBouncer 或 Amazon Advanced JDBC Wrapper Driver。

主要版本升级

主版本升级可能引入与现有应用程序不兼容的更改。

当您启动 RDS for PostgreSQL 多可用区数据库集群的主要版本升级时，Amazon RDS 会同时升级读取器和写入器实例。因此，在升级完成之前，数据库集群可能不可用。

当您启动 RDS for MySQL 多可用区数据库集群的主要版本升级时，Amazon RDS 将一次升级一个集群成员实例，因此，将从较低的引擎版本复制到更高的引擎版本。在主要版本升级期间，必须确保您的工作负载与源引擎和目标引擎版本兼容，因为引擎版本在语法和功能上可能有所不同。

> **Note**
>
> 与次要版本升级一样，RDS for MySQL 主要版本升级期间的停机时间通常为 35 秒。与 RDS 代理一起使用时，您可以将停机时间进一步缩短到一秒或更短。有关更多信息，请参阅 Amazon RDS 代理。

升级多可用区数据库集群

升级多可用区数据库集群的引擎版本的过程与升级数据库实例引擎版本的过程相同。有关说明，请参阅 the section called “升级引擎版本”。唯一的区别是，当使用 Amazon Command Line Interface（Amazon CLI）时，您使用 modify-db-cluster 命令并指定 `--db-cluster-identifier` 参数（以及 `--allow-major-version-upgrade` 参数）。

有关主要版本升级和次要版本升级的更多信息，请参阅数据库引擎的以下文档：

- the section called “升级 PostgreSQL 数据库引擎”
- the section called “升级 MySQL 数据库引擎”

升级多可用区数据库集群只读副本

Amazon RDS 不会自动升级多可用区数据库集群只读副本。对于次要版本升级，必须先手动升级所有只读副本，然后升级集群。否则，升级将被阻止。当您执行集群的主要版本升级时，所有只读副本的复制状态将更改为终止。升级完成后，您必须删除并重新创建只读副本。有关更多信息，请参阅 the section called “监控只读复制”。

重命名 Amazon RDS 的多可用区数据库集群

您可以使用 Amazon Web Services Management Console、Amazon CLI `modify-db-cluster` 命令或 Amazon RDS API `ModifyDBCluster` 操作重命名多可用区数据库集群。重命名多可用区数据库集群可能产生显著影响。以下是您在重命名多可用区数据库集群之前需了解的注意事项列表。

- 当您重命名多可用区数据库集群时，多可用区数据库集群的集群端点会发生变化。这些端点之所以更改，是因为它们包括您分配给多可用区数据库集群的名称。您可以将流量从旧端点重定向到新端点。有关多可用区数据库集群端点的更多信息，请参阅 连接到 Amazon RDS 的多可用区数据库集群。
- 当您重命名多可用区数据库集群时，多可用区数据库集群使用的旧 DNS 名称会被删除，尽管它可能会在缓存中保留几分钟。重命名的多可用区数据库集群的新 DNS 名称在大约两分钟内生效。重命名的多可用区数据库集群在新名称生效之前不可用。
- 重命名集群时，您不能使用现有的多可用区数据库集群名称。
- 如果您重用数据库实例的名称，则与多可用区数据库集群的名称关联的指标和事件保持不变。
- 无论重命名如何，多可用区数据库集群标签都会保留在多可用区数据库集群中。
- 对于重命名的多可用区数据库集群，数据库集群快照也会保留下来。

> **Note**
>
> 多可用区数据库集群是在云中运行的独立数据库环境。多可用区数据库集群可承载多个数据库。有关更改数据库名称的信息，请参阅数据库引擎文档。

重命名以替换现有的多可用区数据库集群

重命名多可用区数据库集群的最常见场景包括从数据库集群快照还原数据或执行时间点故障恢复（PITR）。通过重命名多可用区数据库集群，您可以在不更改任何引用多可用区数据库集群的应用程序代码的情况下替换多可用区数据库集群。在这些情况下，完成以下步骤：

1. 停止流向多可用区数据库集群的所有流量。您可以重定向流量，使其无法访问多可用区数据库集群上的数据库，或者选择其他方式阻止流量访问多可用区数据库集群中的数据库。
2. 重命名现有的多可用区数据库集群。
3. 通过从数据库集群快照进行还原或恢复到某个时间点，创建新的多可用区数据库集群。然后，为新的多可用区数据库集群提供先前多可用区数据库集群的名称。

如果您删除旧的多可用区数据库集群，则您负责删除旧的多可用区数据库集群的任何不需要的数据库集群快照。

控制台

重命名多可用区数据库集群

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases（数据库）。
3. 选择要重命名的多可用区数据库集群。
4. 选择修改。
5. 在 Settings（设置）中，为 DB cluster identifier（数据库集群标识符）输入新名称。
6. 选择 Continue (继续)。
7. 要立即应用更改，请选择立即应用。选择此选项在某些情况下可能导致中断。有关更多信息，请参阅 立即应用更改。
8. 在确认页面上，检查您的更改。如果更改正确无误，请选择修改集群以保存更改。

   或者，选择 Back（返回）以编辑您的更改，或者选择 Cancel（取消）以放弃更改。

Amazon CLI

要重命名多可用区数据库集群，请使用 Amazon CLI 命令 modify-db-cluster。提供当前 `--db-cluster-identifier` 值并为 `--new-db-cluster-identifier` 参数提供多可用区数据库集群的新名称。

Example

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-cluster \
    --db-cluster-identifier DBClusterIdentifier \
    --new-db-cluster-identifier NewDBClusterIdentifier
```

对于 Windows：

```
aws rds modify-db-cluster ^
    --db-cluster-identifier DBClusterIdentifier ^
```

```
    --new-db-cluster-identifier NewDBClusterIdentifier
```

RDS API

要重命名多可用区数据库集群，请使用以下参数调用 Amazon RDS API 操作 ModifyDBCluster：

- `DBClusterIdentifier` – 数据库集群的现有名称。
- `NewDBClusterIdentifier` – 数据库集群的新名称。

重启 Amazon RDS 的多可用区数据库集群和读取器数据库实例

您可能需要重启多可用区数据库集群，通常是出于维护目的。例如，如果要进行某些修改或更改与数据库集群关联的数据库集群参数组，则重启该数据库集群。进行此操作以使更改生效。

如果数据库集群未使用对其关联的数据库集群参数组所做的最新更改，则 Amazon Web Services Management Console 将显示状态为 pending-reboot 的数据库集群参数组。pending-reboot 参数组状态不会在下一个维护时段期间导致自动重启。要向该数据库集群应用最新的参数更改，请手动重启数据库集群。有关参数组的更多信息，请参阅 多可用区数据库集群的参数组。

重启数据库集群会重新启动数据库引擎服务。重启数据库集群将导致短暂中断，在此期间，数据库集群状态将被设为 rebooting（正在重启）。

如果数据库集群未处于 Available（可用）状态，则无法重启。您的数据库可能会由于几个原因而不可用，例如，正在进行备份、以前请求的修改或维护时段操作。

重启数据库集群所需的时间取决于崩溃恢复过程、重启时的数据库活动以及特定数据库集群的行为。为优化重新启动的时间，建议在重启过程中尽可能减少数据库活动。减少数据库活动可减少中转事务的回滚活动。

> **Important**
>
> 多可用区数据库集群不支持通过故障转移重启。当您重启多可用区数据库集群的写入器实例时，它不会影响该数据库集群中的读取器数据库实例，也不会发生故障转移。在重启读取器数据库实例时，不会发生故障转移。要对多可用区数据库集群进行故障转移，请在控制台中选择 Failover（故障转移），调用 Amazon CLI 命令 `failover-db-cluster`，或调用 API 操作 `FailoverDBCluster`。

控制台

要重启数据库集群

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，请选择 Databases（数据库），然后选择要重启的多可用区数据库集群。
3. 对于操作，选择重启。

   此时将显示 Reboot DB cluster（重启数据库集群）页面。

4. 请选择 Reboot（重启）以重启您的数据库集群。

   或者选择 Cancel（取消）。

Amazon CLI

要使用 Amazon CLI 重启多可用区数据库集群，请调用 reboot-db-cluster 命令。

```
aws rds reboot-db-cluster --db-cluster-identifier mymultiazdbcluster
```

RDS API

要使用 Amazon RDS API 重启多可用区数据库集群，请调用 RebootDBCluster 操作。

对 Amazon RDS 的多可用区数据库集群执行失效转移

如果多可用区数据库集群中的写入器数据库实例发生计划内或计划外的中断，Amazon RDS 会自动故障转移到另一个可用区中的读取器数据库实例。这可最大限度地减少中断，从而确保高可用性。失效转移可能发生在硬件故障、网络问题或手动请求期间。该主题概述了故障自动检测、失效转移期间的事件序列及其对读取和写入操作的影响。它还提供了监控失效转移和尽可能减少失效转移时间的最佳实践。

完成故障转移所用的时间取决于在写入器数据库实例变为不可用时的数据库活动和其他条件。故障转移时间通常低于 35 秒。当两个读取器数据库实例都应用了来自失败写入器的未完成事务后，故障转移即完成。完成故障转移后，RDS 控制台还需要一段时间才能反映新的可用区。

主题

自动故障转移

Amazon RDS 会自动处理故障转移，因此，您可以尽快恢复数据库操作而无需管理干预。要进行故障转移，写入器数据库实例将自动切换到读取器数据库实例。

手动对多可用区数据库集群执行故障转移

如果您手动对多可用区数据库集群执行失效转移，RDS 将首先终止主数据库实例。然后，内部监控系统检测到主数据库实例运行状况不佳，并提升可读副本数据库实例。故障转移时间通常低于 35 秒。

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 手动对多可用区数据库集群执行故障转移。

控制台

要手动对多可用区数据库集群执行故障转移

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases（数据库）。
3. 选择要进行故障转移的多可用区数据库集群。

4. 对于 Actions（操作），请选择 Failover（故障转移）。

   此时将会显示失效转移数据库集群页面。

5. 请选择 Failover（故障转移）以确认手动进行故障转移。

Amazon CLI

要手动对多可用区数据库集群执行故障转移，请使用 Amazon CLI 命令 failover-db-cluster。

Example

```
aws rds failover-db-cluster --db-cluster-identifier mymultiazdbcluster
```

RDS API

要手动对多可用区数据库集群执行故障转移，请调用 Amazon RDS API FailoverDBCluster 然后指定 `DBClusterIdentifier`。

确定多可用区数据库集群是否已完成故障转移

要确定多可用区数据库集群是否发生故障转移，您可以执行以下操作：

- 将数据库事件订阅设置为在故障转移启动时向您发送电子邮件或 SMS 通知。有关事件的更多信息，请参阅 使用 Amazon RDS 事件通知。
- 使用 Amazon RDS 控制台或 API 操作查看数据库事件。
- 使用 Amazon RDS 控制台、Amazon CLI 以及 RDS API 查看多可用区数据库集群的当前状态。

有关如何响应故障转移、缩短恢复时间以及 Amazon RDS 的其他最佳实践的信息，请参阅 Amazon RDS 的最佳实践。

设置 DNS 名称查找的 JVM TTL

故障转移机制自动更改数据库实例的域名系统 (DNS) 记录，使其指向读取器数据库实例。因此，您需要重新建立与数据库实例之间的所有现有连接。在 Java 虚拟机 (JVM) 环境中，由于 Java DNS 缓存机制的工作原理，您可能需要重新配置 JVM 设置。

JVM 缓存 DNS 名称查找。当 JVM 将主机名解析为 IP 地址时，它会在指定时间段内 (称为存活时间 (TTL)) 缓存 IP 地址。

由于 Amazon 资源使用偶尔变更的 DNS 名称条目，因此建议您为 JVM 配置的 TTL 值不超过 60 秒。这样做可确保在资源的 IP 地址发生更改时，您的应用程序可以通过重新查询 DNS 来接收和使用资源的新 IP 地址。

对于一些 Java 配置，将设置 JVM 默认 TTL，以便在重新启动 JVM 之前绝不刷新 DNS 条目。因此，如果 Amazon 资源的 IP 地址在应用程序仍在运行时发生更改，则在您手动重新启动 JVM 并刷新缓存的 IP 信息之前，将无法使用该资源。在此情况下，设置 JVM 的 TTL，以便定期刷新其缓存的 IP 信息是极为重要的。

> **Note**
>
> 默认 TTL 是变化的，具体取决于 JVM 的版本以及是否安装安全管理器。许多 JVM 提供的默认 TTL 小于 60 秒。如果您使用此类 JVM 并且未使用安全管理器，则您可以忽略本主题的剩余内容。有关 Oracle 中安全管理器的更多信息，请参阅 Oracle 文档中的安全管理器。

要修改 JVM 的 TTL，请设置 `networkaddress.cache.ttl` 属性值。根据您的需求，使用下列方法之一：

- 要为使用 JVM 的所有应用程序全局设置属性值，请在 `networkaddress.cache.ttl` 文件中设置 `$JAVA_HOME/jre/lib/security/java.security`。

```
networkaddress.cache.ttl=60
```

- 要仅在本地为应用程序设置属性，请在建立任何网络连接之前，在应用程序的初始化代码中设置 `networkaddress.cache.ttl`。

```
java.security.Security.setProperty("networkaddress.cache.ttl" , "60");
```

使用多可用区数据库集群为 Amazon RDS 设置 PostgreSQL 逻辑复制

通过将 PostgreSQL 逻辑复制与多可用区数据库集群结合使用，您可以复制和同步各个表，而不是整个数据库实例。逻辑复制使用发布和订阅模型将更改从源复制到一个或多个接收者。它的工作原理是使用 PostgreSQL 预写日志（WAL）中的更改记录。有关更多信息，请参阅 the section called “执行逻辑复制”。

当您在多可用区数据库集群的写入器数据库实例上创建新的逻辑复制插槽时，该插槽会异步复制到集群中的每个读取器数据库实例上。读取器数据库实例上的插槽会持续与写入器数据库实例上的插槽同步。

运行 RDS for PostgreSQL 版本 14.8-R2 及更高版本以及 15.3-R2 及更高版本的多可用区数据库集群支持逻辑复制。

> **Note**
>
> 除了原生 PostgreSQL 逻辑复制功能外，运行 RDS for PostgreSQL 的多可用区数据库集群还支持 `pglogical` 扩展。

有关 PostgreSQL 逻辑复制的更多信息，请参阅 PostgreSQL 文档中的逻辑复制。

主题

- 先决条件
- 设置逻辑复制
- 限制和建议

先决条件

要为多可用区数据库集群配置 PostgreSQL 逻辑复制，必须满足以下先决条件。

- 您的用户账户必须是 `rds_superuser` 组的成员并具有 `rds_superuser` 权限。有关更多信息，请参阅 the section called “了解 PostgreSQL 角色和权限”。
- 您的多可用区数据库集群必须与自定义数据库集群参数组相关联，您才能配置以下过程中所述的参数值。有关更多信息，请参阅 the section called “数据库集群参数组”。

设置逻辑复制

要为多可用区数据库集群设置逻辑复制，请在关联的数据库集群参数组中启用特定参数，然后创建逻辑复制插槽。

> **Note**
>
> 从 PostgreSQL 版本 16 开始，您可以使用多可用区数据库集群的读取器数据库实例来进行逻辑复制。

为 RDS for PostgreSQL 多可用区数据库集群设置逻辑复制

1. 打开与您的 RDS for PostgreSQL 多可用区数据库集群关联的自定义数据库集群参数组。
2. 在参数搜索字段中，找到 `rds.logical_replication` 静态参数并将其值设置为 `1`。此参数更改会增加 WAL 生成，因此只有在使用逻辑插槽时才启用它。
3. 在此更改过程中，请配置以下数据库集群参数。

    - `max_wal_senders`
    - `max_replication_slots`
    - `max_connections`

    根据您的预期使用情况，您可能还需要更改以下参数的值。但是，在许多情况下，原定设置值就足够了。

    - `max_logical_replication_workers`
    - `max_sync_workers_per_subscription`
4. 重启多可用区数据库集群以使参数值生效。有关说明，请参阅 the section called “重启多可用区数据库集群”。
5. 如the section called “使用逻辑复制槽”中所述，在多可用区数据库集群的写入器数据库实例上创建逻辑复制插槽。该过程需要您指定解码插件。目前，RDS for PostgreSQL 支持 PostgreSQL 随附的 `test_decoding`、`wal2json` 和 `pgoutput` 插件。

    该插槽会异步复制到集群中的每个读取器数据库实例。
6. 验证多可用区数据库集群所有读取器数据库实例上的插槽状态。为此，请检查所有读取器数据库实例上的 `pg_replication_slots` 视图，并确保在应用程序积极使用逻辑更改时，状态 `confirmed_flush_lsn` 推进。

以下命令演示了如何检查读取器数据库实例上的复制状态。

```
% psql -h test-postgres-instance-2.abcdefabcdef.us-west-2.rds.amazonaws.com

postgres=> select slot_name, slot_type, confirmed_flush_lsn from
 pg_replication_slots;
  slot_name    | slot_type | confirmed_flush_lsn
--------------+-----------+---------------------
 logical_slot | logical   | 32/D0001700
(1 row)

postgres=> select slot_name, slot_type, confirmed_flush_lsn from
 pg_replication_slots;
  slot_name    | slot_type | confirmed_flush_lsn
--------------+-----------+---------------------
 logical_slot | logical   | 32/D8003628
(1 row)

% psql -h test-postgres-instance-3.abcdefabcdef.us-west-2.rds.amazonaws.com

postgres=> select slot_name, slot_type, confirmed_flush_lsn from
 pg_replication_slots;
  slot_name    | slot_type | confirmed_flush_lsn
--------------+-----------+---------------------
 logical_slot | logical   | 32/D0001700
(1 row)

postgres=> select slot_name, slot_type, confirmed_flush_lsn from
 pg_replication_slots;
  slot_name    | slot_type | confirmed_flush_lsn
--------------+-----------+---------------------
 logical_slot | logical   | 32/D8003628
(1 row)
```

完成复制任务后，停止复制过程，删除复制插槽并关闭逻辑复制。要关闭逻辑复制，请修改您的数据库集群参数组并将 `rds.logical_replication` 的值设回 `0`。重启集群以使参数更改生效。

限制和建议

在使用运行 PostgreSQL 16 的多可用区数据库集群进行逻辑复制时，以下限制和建议适用：

- 您只能使用写入器数据库实例来创建或删除逻辑复制槽。例如，CREATE SUBSCRIPTION 命令必须在主机连接字符串中使用集群写入器端点。
- 在任何表同步或再同步期间，您都必须使用集群写入器端点。例如，您可以使用以下命令来再同步新添加的表。

```
Postgres=>ALTER SUBSCRIPTION subscription-name CONNECTION host=writer-endpoint
Postgres=>ALTER SUBSCRIPTION subscription-name REFRESH PUBLICATION
```

- 在使用读取器数据库实例进行逻辑复制之前，必须等待表完成同步。您可以使用 [pg_subscription_rel](https://www.postgresql.org/docs/current/catalog-pg-subscription-rel.html) 目录表来监控表同步。当 srsubstate 列设置为就绪（r）时，表示表同步已完成。
- 建议您在初始表同步完成后使用实例端点进行逻辑复制连接。以下命令通过将复制载荷分流到一个读取器数据库实例，来减少写入器数据库实例的载荷:

```
Postgres=>ALTER SUBSCRIPTION subscription-name CONNECTION host=reader-instance-endpoint
```

  您不能同时在多个数据库实例上使用相同的复制槽。当两个或更多应用程序从集群中的不同数据库实例复制逻辑更改时，某些更改可能由于集群失效转移或网络问题而丢失。在此类情况下，您可以使用实例端点在主机连接字符串中进行逻辑复制。使用相同配置的另一个应用程序将显示以下错误消息：

```
replication slot slot_name is already active for PID x providing immediate feedback.
```

- 使用 pglogical 扩展时，您只能使用集群写入器端点。该扩展具有已知的局限性，可能会在表同步期间创建未使用的逻辑复制槽。过时的复制槽会保留预写日志（WAL）文件，并可能导致磁盘空间问题。

使用 Amazon RDS 的多可用区数据库集群只读副本

数据库集群只读副本是从源数据库实例创建的一种特殊类型的集群。在创建只读副本之后，对主数据库实例的任何更新都将异步复制到多可用区数据库集群只读副本。您可以将从应用程序发出的读取查询路由到只读副本，以减轻主数据库实例上的负载。利用只读副本，可以实现弹性扩展并超越单个数据库实例的容量限制，以处理高读取量的数据库工作负载。

还可以从多可用区数据库集群创建一个或多个数据库实例只读副本。数据库实例只读副本可让您将多余的读取流量引导到只读副本，从而扩展到源多可用区数据库集群的计算容量或 I/O 容量之外。目前，您无法从现有的多可用区数据库集群创建多可用区数据库集群只读副本。

在使用只读副本迁移到多可用区数据库集群或从多可用区数据库集群中创建数据库实例只读副本之间进行选择时，请考虑您的特定使用案例和性能要求。

使用只读副本迁移到多可用区数据库集群

当您需要增强数据库的可用性和耐久性，同时最大限度地减少停机时间时，这种方法是理想之选。通过使用只读副本转换到多可用区数据库集群，可以确保持续运行和数据一致性。这种方法对于生产环境特别有用，在这类环境中，保持可用性和减少对实时工作负载的影响至关重要。

从多可用区数据库集群创建数据库实例只读副本

当您想要扩展读取操作或从主数据库实例卸载读取流量时，此方法很适合。通过从现有的多可用区数据库集群创建只读副本，可以在不影响主实例稳定性的情况下，分配读取密集型工作负载并提高性能。

选择正确的方法取决于优先事项是确保高可用性和耐久性，还是扩展读取性能。评估工作负载特征和操作要求来做出明智的决策。

主题

使用只读副本迁移到多可用区数据库集群

要将单可用区部署或多可用区数据库实例部署迁移到停机时间更少的多可用区数据库集群部署，您可以创建多可用区数据库集群只读副本。对于源，您可以在单可用区部署中指定数据库实例，或在多可用区数据库实例部署中指定主数据库实例。数据库实例可以在迁移到多可用区数据库集群期间处理写入事务。

在创建多可用区数据库集群只读副本之前，请考虑以下事项：

- 源数据库实例的版本必须支持多可用区数据库集群。有关更多信息，请参阅 支持 Amazon RDS 中多可用区数据库集群的区域和数据库引擎。
- 多可用区数据库集群只读副本的主版本必须与其源的版本相同，次要版本必须相同或更高。
- 您必须先通过将备份保留期设置为一个非零值来在源数据库实例上开启自动备份。
- 源数据库实例的分配存储空间必须为 100GiB 或更高。
- 对于 RDS for MySQL，源数据库实例的 `gtid-mode` 和 `enforce_gtid_consistency` 参数都必须设置为 `ON`。您必须使用自定义参数组，而不是原定设置参数组。有关更多信息，请参阅 the section called “数据库参数组”。
- 长时间运行的活动事务会减慢创建只读副本的过程。我们建议您等待长时间运行的事务完成，然后再创建只读副本。
- 如果您删除多可用区数据库集群只读副本的源数据库实例，则只读副本会被提升为独立的多可用区数据库集群。

创建和提升多可用区数据库集群只读副本

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 创建和提升多可用区数据库集群只读副本。

> **Note**
>
> 我们强烈建议您基于源数据库实例的 Amazon VPC，在相同的虚拟私有云（VPC）中创建所有只读副本。
> 如果您在与源数据库实例不同的 VPC 中创建只读副本，则副本和 Amazon RDS 系统之间的无类别域间路由（CIDR）范围可能会重叠。CIDR 重叠会使副本不稳定，从而对连接到副本的应用程序产生负面影响。如果您在创建只读副本时收到错误，请选择其他目标数据库子网组。有关更多信息，请参阅 在 VPC 中使用数据库实例。

控制台

要使用只读副本将单可用区部署或多可用区数据库实例部署迁移到多可用区数据库集群，请使用 Amazon Web Services Management Console 完成以下步骤。

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。

2. 创建多可用区数据库集群只读副本。

   a. 在导航窗格中，选择 Databases (数据库)。

   b. 选择要作为只读副本源的数据库实例。

   c. 对于 Actions (操作)，选择 Create read replica (创建只读副本)。

   d. 对于 Availability and durability（可用性与持久性），选择 Multi-AZ DB cluster（多可用区数据库集群）。

   e. 对于数据库实例标识符，输入该只读副本的名称。

   f. 对于其余部分，请指定数据库集群设置。有关设置的信息，请参阅 创建多可用区数据库集群的设置。

   g. 选择 Create read replica (创建只读副本)。

3. 准备就绪后，将只读副本提升为独立的多可用区数据库集群：

   a. 停止任何事务向源数据库实例写入，然后等待对只读副本完成所有更新。

      在主数据库实例上完成数据库更新后，只读副本进行数据库更新。这种复制滞后可能会有很大差异。使用 `ReplicaLag` 指标确定只读副本完成所有更新的时间。有关复制滞后的更多信息，请参阅 监控只读复制。

   b. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。

   c. 在 Amazon RDS 控制台中，选择数据库。

      随后会显示数据库窗格。每个只读副本在角色列中显示副本。

   d. 选择要提升的多可用区数据库集群只读副本。

   e. 对于操作，请选择提升。

   f. 在 Promote read replica（提升只读副本）页面上，输入新提升的多可用区数据库集群的备份保留期和备份时段。

   g. 根据需要完成设置后，选择 Promote read replica（提升只读副本）。

   h. 等待所提升的多可用区数据库集群的状态变为 `Available`。

   i. 指示您的应用程序使用提升的多可用区数据库集群。

（可选）如果不再需要单可用区部署或多可用区数据库实例部署，则将其删除。有关说明，请参阅 删除数据库实例。

Amazon CLI

要使用只读副本将单可用区部署或多可用区数据库实例部署迁移到多可用区数据库集群，请使用 Amazon CLI 完成以下步骤。

1. 创建多可用区数据库集群只读副本。

   要从源数据库实例创建只读副本，请使用 Amazon CLI 命令 `create-db-cluster`。对于 `--replication-source-identifier`，指定源数据库实例的 Amazon 资源名称（ARN）。

   对于 Linux、macOS 或 Unix：

```
aws rds create-db-cluster \
  --db-cluster-identifier mymultiazdbcluster \
  --replication-source-identifier arn:aws:rds:us-
east-2:123456789012:db:mydbinstance
  --engine postgres \
  --db-cluster-instance-class db.m5d.large \
  --storage-type io1 \
  --iops 1000 \
  --db-subnet-group-name defaultvpc \
  --backup-retention-period 1
```

   对于 Windows：

```
aws rds create-db-cluster ^
  --db-cluster-identifier mymultiazdbcluster ^
  --replication-source-identifier arn:aws:rds:us-
east-2:123456789012:db:mydbinstance
  --engine postgres ^
  --db-cluster-instance-class db.m5d.large ^
  --storage-type io1 ^
  --iops 1000 ^
  --db-subnet-group-name defaultvpc ^
  --backup-retention-period 1
```

2. 停止任何事务向源数据库实例写入，然后等待对只读副本完成所有更新。

   在主数据库实例上完成数据库更新后，只读副本进行数据库更新。这种复制滞后可能会有很大差异。使用 `Replica Lag` 指标确定只读副本完成所有更新的时间。有关复制滞后的更多信息，请参阅 监控只读复制。

3. 准备就绪后，将只读副本提升为独立的多可用区数据库集群。

要提升多可用区数据库集群只读副本，请使用 Amazon CLI 命令 [promote-read-replica-db-cluster]。对于 `--db-cluster-identifier`，指定多可用区数据库集群只读副本的标识符。

```
aws rds promote-read-replica-db-cluster --db-cluster-identifier mymultiazdbcluster
```

4. 等待所提升的多可用区数据库集群的状态变为 `Available`。
5. 指示您的应用程序使用提升的多可用区数据库集群。

（可选）如果不再需要单可用区部署或多可用区数据库实例部署，则将其删除。有关说明，请参阅 删除数据库实例。

RDS API

要使用只读副本将单可用区部署或多可用区数据库实例部署迁移到多可用区数据库集群，请使用 RDS API 完成以下步骤。

1. 创建多可用区数据库集群只读副本。

   要创建多可用区数据库集群只读副本，请使用带有所需参数 `DBClusterIdentifier` 的 `CreateDBCluster` 操作。对于 `ReplicationSourceIdentifier`，指定源数据库实例的 Amazon 资源名称（ARN）。
2. 停止任何事务向源数据库实例写入，然后等待对只读副本完成所有更新。

   在主数据库实例上完成数据库更新后，只读副本进行数据库更新。这种复制滞后可能会有很大差异。使用 `Replica Lag` 指标确定只读副本完成所有更新的时间。有关复制滞后的更多信息，请参阅 监控只读复制。
3. 准备就绪后，将只读副本提升为独立的多可用区数据库集群。

   要提升多可用区数据库集群只读副本，请使用带有所需参数 `DBClusterIdentifier` 的 `PromoteReadReplicaDBCluster` 操作。指定多可用区数据库集群只读副本的标识符。
4. 等待所提升的多可用区数据库集群的状态变为 `Available`。
5. 指示您的应用程序使用提升的多可用区数据库集群。

（可选）如果不再需要单可用区部署或多可用区数据库实例部署，则将其删除。有关说明，请参阅 删除数据库实例。

创建多可用区数据库集群只读副本的限制

以下限制适用于从单可用区部署或多可用区数据库实例部署创建多可用区数据库集群部署只读副本。

- 您无法在与拥有源数据库实例的 Amazon Web Services 账户不同的 Amazon Web Services 账户中创建多可用区数据库集群只读副本。
- 您无法在与源数据库实例不同的 Amazon Web Services 区域 中创建多可用区数据库集群只读副本。
- 您无法将多可用区数据库集群只读副本还原到某个时间点。
- 存储加密在源数据库实例和多可用区数据库集群上必须具有相同的设置。
- 如果源数据库实例已加密，则必须使用相同的 KMS 密钥对多可用区数据库集群只读副本进行加密。
- 如果源数据库实例使用通用型 SSD（gp3）存储且分配的存储空间低于 400 GiB，则无法修改多可用区数据库集群只读副本的预调配 IOPS。
- 要对源数据库实例执行次要版本升级，必须先对多可用区数据库集群只读副本执行次要版本升级。
- 在对 RDS for PostgreSQL 多可用区数据库集群只读副本执行次要版本升级时，在升级之后，读取器数据库实例不会切换到写入器数据库实例。因此，在 Amazon RDS 升级写入器实例时，数据库集群可能会遇到停机。
- 您无法对多可用区数据库集群只读副本执行主要版本升级。
- 您可以对多可用区数据库集群只读副本的源数据库实例执行主要版本升级，但复制到只读副本的操作会停止且无法重新启动。
- 多可用区数据库集群只读副本不支持级联只读副本。
- 对于 RDS for PostgreSQL，多可用区数据库集群只读副本无法进行失效转移。

从多可用区数据库集群创建数据库实例只读副本

您可以从多可用区数据库集群创建一个数据库实例只读副本，以便扩展到集群的计算容量或 I/O 容量之外，以应对读取量繁重的数据库工作负载。可以将这一过多的读取流量引导到一个或多个数据库实例只读副本。还可以使用只读副本从多可用区数据库集群迁移到数据库实例。

要创建只读副本，请将多可用区数据库集群指定为复制源。多可用区数据库集群的读取器实例之一始终是复制源，而并非写入器实例。此条件可确保副本始终与源集群同步，即使在失效转移的情况下也是如此。

主题

- [比较读取器数据库实例和数据库实例只读副本]
- [注意事项]

- 创建数据库实例只读副本
- 提升数据库实例只读副本
- 从多可用区数据库集群创建数据库实例只读副本的限制

比较读取器数据库实例和数据库实例只读副本

多可用区数据库集群的数据库实例只读副本在以下各个方面不同于多可用区数据库集群的读取器数据库实例：

- 读取器数据库实例充当自动失效转移目标，而数据库实例只读副本则不是。
- 读取器数据库实例必须先确认写入器数据库实例中的更改，然后才能提交更改。但是，对于数据库实例只读副本，更新将异步复制到只读副本，而无需确认。
- 读取器数据库实例始终与多可用区数据库集群的写入器数据库实例共享相同的实例类、存储类型和引擎版本。但是，数据库实例只读副本不一定必须与源集群共享相同的配置。
- 可以将数据库实例只读副本提升为单独的数据库实例。您无法将多可用区数据库集群的读取器数据库实例提升为独立实例。
- 读取器端点仅将请求路由到多可用区数据库集群的读取器数据库实例。该端点从不将请求路由到数据库实例只读副本。

有关读取器和写入器数据库实例的更多信息，请参阅the section called “多可用区数据库集群架构”。

注意事项

在从多可用区数据库集群创建数据库实例只读副本之前，请考虑以下事项：

- 创建数据库实例只读副本时，此只读副本必须与其源集群属于相同的主版本以及相同或更高的次要版本。创建只读副本后，您可以选择将此只读副本升级到比源集群更高的次要版本。
- 创建数据库实例只读副本时，分配的存储空间必须与源多可用区数据库集群的已分配存储空间相同。创建只读副本后，您可以更改分配的存储空间。
- 对于 RDS for MySQL，源多可用区数据库集群的 `gtid-mode` 参数必须设置为 `ON`。有关更多信息，请参阅 the section called “数据库集群参数组”。
- 长时间运行的活动事务会减慢创建只读副本的过程。我们建议您等待长时间运行的事务完成，然后再创建只读副本。
- 如果您删除数据库实例只读副本的源多可用区数据库集群，则该集群正在写入的任何只读副本都将提升为独立的数据库实例。

创建数据库实例只读副本

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 从多可用区数据库集群创建数据库实例只读副本。

> **Note**
>
> 我们强烈建议您基于源多可用区数据库集群的 Amazon VPC，在相同的虚拟私有云（VPC）中创建所有只读副本。
> 如果您在与源多可用区数据库集群不同的 VPC 中创建只读副本，则副本和 RDS 系统之间的无类别域间路由（CIDR）范围可能会重叠。CIDR 重叠会使副本不稳定，从而对连接到副本的应用程序产生负面影响。如果您在创建只读副本时收到错误，请选择其他目标数据库子网组。有关更多信息，请参阅 the section called “在 VPC 中使用数据库实例”。

控制台

要从多可用区数据库集群创建数据库实例只读副本，请使用 Amazon Web Services Management Console完成以下步骤。

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases（数据库）。
3. 选择要作为只读副本的源的多可用区数据库集群。
4. 对于 Actions (操作)，选择 Create read replica (创建只读副本)。
5. 对于副本源，请确保选择了正确的多可用区数据库集群。
6. 对于数据库标识符，输入只读副本的名称。
7. 对于其余部分，请指定数据库实例设置。有关设置的信息，请参阅 the section called “可用的设置”。

   > **Note**
   >
   > 为数据库实例只读副本分配的存储空间必须与为源多可用区数据库集群分配的存储空间相同。

8. 选择 Create read replica (创建只读副本)。

Amazon CLI

要从多可用区数据库集群创建数据库实例只读副本，请使用 Amazon CLI 命令 [create-db-instance-read-replica]。对于 `--source-db-cluster-identifier`，指定多可用区数据库集群的标识符。

对于 Linux、macOS 或 Unix：

```
aws rds create-db-instance-read-replica \
  --db-instance-identifier myreadreplica \
  --source-db-cluster-identifier mymultiazdbcluster
```

对于 Windows：

```
aws rds create-db-instance-read-replica ^
  --db-instance-identifier myreadreplica ^
  --source-db-cluster-identifier mymultiazdbcluster
```

RDS API

要从多可用区数据库集群创建数据库实例只读副本，请使用 [CreateDBInstanceReadReplica] 操作。

提升数据库实例只读副本

如果您不再需要数据库实例只读副本，则可以将该只读副本提升为独立的数据库实例。提升只读副本时，数据库实例将在重启后变得可用。有关说明，请参阅 the section called “提升只读副本”。

如果您使用只读副本将多可用区数据库集群部署迁移到单可用区或多可用区数据库实例部署，请确保停止正在向源数据库集群写入的任何事务。然后，等待对只读副本完成所有更新。在多可用区数据库集群的其中一个读取器数据库实例上发生数据库更新后，只读副本会发生数据库更新。这种复制滞后可能会有很大差异。使用 `ReplicaLag` 指标确定只读副本完成所有更新的时间。有关复制滞后的更多信息，请参阅 the section called “监控只读复制”。

提升只读副本后，请等待已提升的数据库实例的状态变为 `Available`，然后再引导应用程序使用提升的数据库实例。或者，如果您不再需要多可用区数据库集群部署，请将其删除。有关说明，请参阅 the section called “删除多可用区数据库集群”。

从多可用区数据库集群创建数据库实例只读副本的限制

以下限制适用于从多可用区数据库集群部署创建数据库实例只读副本。

- 您无法在与拥有源多可用区数据库集群的 Amazon Web Services 账户不同的 Amazon Web Services 账户中创建数据库实例只读副本。
- 您无法在与源多可用区数据库集群不同的 Amazon Web Services 区域中创建数据库实例只读副本。
- 您无法将数据库实例只读副本还原到某个时间点。
- 存储加密在源多可用区数据库集群和数据库实例只读副本上必须具有相同的设置。
- 如果源多可用区数据库集群已加密，则必须使用相同的 KMS 密钥对数据库实例只读副本进行加密。
- 要对源多可用区数据库集群执行次要版本升级，必须先对数据库实例只读副本执行次要版本升级。
- 数据库实例只读副本不支持级联只读副本。
- 对于 RDS for PostgreSQL，源多可用区数据库集群必须运行 PostgreSQL 版本 13.11、14.8 或 15.2.R2 或更高版本，才能创建数据库实例只读副本。
- 您可以对数据库实例只读副本的源多可用区数据库集群执行主要版本升级，但复制到只读副本的操作会停止且无法重新启动。

设置从 Amazon RDS 的多可用区数据库集群进行外部复制

可以在多可用区数据库集群和 Amazon RDS 外部的数据库之间设置复制。

外部复制支持多可用区数据库集群在 RDS 数据库实例和外部数据库之间复制数据，无论是在本地还是在其它云环境中。外部复制有利于灾难恢复、数据迁移和维护不同位置的系统之间的一致性。本节涵盖了设置复制的先决条件，如何配置复制过程，以及诸如复制延迟、带宽及与不同数据库引擎的兼容性等关键注意事项。

RDS for MySQL

要为 RDS for MySQL 多可用区数据库集群设置外部复制，必须在集群内的数据库实例上将二进制日志文件保留足够长的时间，来确保在 Amazon RDS 删除二进制日志文件之前将更改应用于副本。为此，请通过调用 `mysql.rds_set_configuration` 存储过程并指定 `binlog retention hours` 参数来配置二进制日志保留期。有关更多信息，请参阅 the section called “二进制日志保留小时数”。

`binlog retention hours` 的默认值为 `NULL`，这意味着不保留二进制日志（0 小时）。如果要为多可用区数据库集群设置外部复制，则必须将参数设置为 `NULL` 以外的值。

只能从多可用区数据库集群的写入器数据库实例中配置二进制日志保留，并且该设置会异步传播到所有读取器数据库实例。

此外，我们强烈建议在外部副本上启用基于 GTID 的复制。然后，如果其中一个数据库实例出现故障，则可以从集群中另一个运行状况正常的数据库实例继续复制。有关更多信息，请参阅 MySQL 文档中的 Replication with Global Transaction Identifiers。

RDS for PostgreSQL

要为 RDS for PostgreSQL 多可用区数据库集群设置外部复制，必须启用逻辑复制。有关说明，请参阅 the section called “使用多可用区数据库集群的 PostgreSQL 逻辑复制”。

删除 Amazon RDS 的多可用区数据库集群

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 删除数据库多可用区数据库集群。

删除多可用区数据库集群所需的时间可能因以下因素而异：

- 备份保留期（即，要删除的备份数）。
- 删除的数据量。
- 是否拍摄了最终快照。

必须先在多可用区数据库集群上禁用删除保护功能，然后才能将其删除。有关更多信息，请参阅 the section called “删除数据库实例的先决条件”。您可以通过修改多可用区数据库集群来禁用删除保护。有关更多信息，请参阅 the section called “修改多可用区数据库集群”。

控制台

要删除多可用区数据库集群

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，请选择 Databases（数据库），然后选择要删除的多可用区数据库集群。
3. 对于操作，选择删除。
4. 请选择 Create final snapshot（是否创建最终快照？）来为多可用区数据库集群创建最终数据库快照。

   如果创建最终快照，请为 Final snapshot name（最终快照名称）输入名称。
5. 请选择 Retain automated backups（保留自动备份）以保留自动备份。
6. 在框中输入 **`delete me`**。
7. 选择删除。

Amazon CLI

要使用 Amazon CLI 删除多可用区数据库集群，请调用 delete-db-cluster 命令，并采用以下选项：

- `--db-cluster-identifier`
- `--final-db-snapshot-identifier` 或 `--skip-final-snapshot`

Example 拥有最终快照

对于 Linux、macOS 或 Unix：

```
aws rds delete-db-cluster \
    --db-cluster-identifier mymultiazdbcluster \
    --final-db-snapshot-identifier mymultiazdbclusterfinalsnapshot
```

对于 Windows：

```
aws rds delete-db-cluster ^
    --db-cluster-identifier mymultiazdbcluster ^
    --final-db-snapshot-identifier mymultiazdbclusterfinalsnapshot
```

Example 没有最终快照

对于 Linux、macOS 或 Unix：

```
aws rds delete-db-cluster \
    --db-cluster-identifier mymultiazdbcluster \
    --skip-final-snapshot
```

对于 Windows：

```
aws rds delete-db-cluster ^
    --db-cluster-identifier mymultiazdbcluster ^
    --skip-final-snapshot
```

RDS API

要使用 Amazon RDS API 删除多可用区数据库集群，请调用 DeleteDBCluster 操作，并采用以下参数：

- `DBClusterIdentifier`
- `FinalDBSnapshotIdentifier` 或者 `SkipFinalSnapshot`

Amazon RDS 的多可用区数据库集群的限制

多可用区数据库集群在三个独立可用区中有一个写入器数据库实例和两个读取器数据库实例。与多可用区部署相比，多可用区数据库集群可提供高可用性、增加读取工作负载容量以及更低的延迟。有关多可用区数据库集群的更多信息，请参阅 Amazon RDS 的多可用区数据库集群部署。

以下限制适用于多可用区数据库集群。

- 多可用区数据库集群不支持以下特征：
  - IPv6 连接（双堆栈模式）
  - 跨区域自动备份
  - Kerberos 身份验证
  - 修改端口。作为替代方法，您可以将多可用区数据库集群还原到某个时间点并指定其他端口。
  - 选项组
  - 已删除的集群的时间点故障恢复（PITR）
  - 通过设置最大分配的存储空间来执行存储自动扩缩。作为替代方法，您可以手动扩展存储。
  - 停止并启动多可用区数据库集群
  - 复制多可用区数据库集群的快照
  - 加密未加密的多可用区数据库集群
- RDS for MySQL 多可用区数据库集群仅支持以下系统存储过程：
  - `mysql.rds_rotate_general_log`
  - `mysql.rds_rotate_slow_log`
  - `mysql.rds_show_configuration`
  - `mysql.rds_set_external_master_with_auto_position`
  - `mysql.rds_set_configuration`
- RDS for PostgreSQL 多可用区数据库集群不支持以下扩展：`aws_s3` 和 `pg_transport`。
- RDS for PostgreSQL 多可用区数据库集群不支持使用自定义 DNS 服务器进行出站网络访问。

Amazon RDS 的 Amazon RDS 扩展支持

借助 Amazon RDS 扩展支持，您可以在 RDS 标准支持终止日期后，继续在主要引擎版本上运行数据库，但需要额外付费。在 RDS 标准支持终止日期之后，如果您在创建或还原数据库实例期间没有禁用 RDS 扩展支持，则 Amazon RDS 会自动将它们注册到 RDS 扩展支持。自动注册到 RDS 扩展支持不会更改数据库引擎，也不会影响数据库实例的正常运行时间或性能。

RDS 扩展支持提供以下更新和技术支持：

- 数据库实例或数据库集群（包括数据库引擎）的 critical and high CVEs 的安全更新
- 针对关键问题的错误修复和补丁
- 能够在标准 Amazon RDS 服务等级协议范围内建立支持案例并获得故障排除帮助

这项付费服务让您有更多时间升级到支持的主要引擎版本。例如，RDS For MySQL 版本 5.7 的 RDS 标准支持终止日期为 2024 年 2 月 29 日。但是，在该日期之前，您还没有准备好手动升级到 RDS for MySQL 版本 8.0。在这一情况下，Amazon RDS 会在 2024 年 2 月 29 日自动将您的数据库注册到 RDS 扩展支持，并且您可以继续运行 RDS for MySQL 版本 5.7。从 2024 年 3 月 1 日起，Amazon RDS 将自动向您收取 RDS 扩展支持费用。

对于主要引擎版本，在 RDS 标准支持终止日期后，RDS 扩展支持可提供长达 3 年的服务。此时间之后，如果您尚未将主要引擎版本升级到支持的版本，Amazon RDS 将自动升级您的主要引擎版本。建议您尽快升级到支持的主要引擎版本。

有关 RDS 标准支持终止日期和 RDS 扩展支持终止日期的更多信息，请参阅 Amazon RDS 上支持的 MySQL 主要版本和 Amazon RDS for PostgreSQL 的发布日历。

主题

Amazon RDS 扩展支持概述

在 RDS 标准支持终止日期之后，如果您在创建或还原数据库实例期间没有禁用 RDS 扩展支持，则 Amazon RDS 会自动将它们注册到 RDS 扩展支持。如果您尚未运行在 RDS 标准支持终止日期之前发布的最后一个次要版本，Amazon RDS 会自动将您的数据库实例升级到该次要版本。直至主要引擎版本的 RDS 标准支持终止日期之后，Amazon RDS 才会升级您的次要版本。

您可以使用已达到 RDS 标准支持终止日期的主要引擎版本创建新的数据库。RDS 会自动在 RDS 扩展支持中注册这些新数据库，并向您收取此服务的费用。

如果您在 RDS 标准支持终止日期之前 升级到仍处于 RDS 标准支持之下的引擎，Amazon RDS 不会将您的引擎注册到 RDS 扩展支持。

如果您试图还原与超过 RDS 标准支持终止日期但未注册到 RDS 扩展支持的引擎兼容的数据库快照，则 Amazon RDS 将尝试升级快照，使其与仍处于 RDS 标准支持之下的最新引擎版本兼容。如果还原失败，则 Amazon RDS 将使用与快照兼容的版本，自动将您的引擎注册到 RDS 扩展支持。

您可以随时终止 RDS 扩展支持的注册。要终止注册，请将每个已注册的引擎升级到仍处于 RDS 标准支持之下的更高引擎版本。RDS 扩展支持注册的终止将在您完成升级到仍处于 RDS 标准支持之下的更高引擎版本的当天生效。

有关 RDS 标准支持终止日期和 RDS 扩展支持终止日期的更多信息，请参阅 Amazon RDS 上支持的 MySQL 主要版本和 Amazon RDS for PostgreSQL 的发布日历。

Amazon RDS 扩展支持费用

从 RDS 标准支持终止日期的第二天开始，您将为在 RDS 扩展支持中注册的所有引擎付费。有关 RDS 标准支持终止日期，请参阅主要版本和 Release calendar for Amazon RDS for PostgreSQL。RDS 扩展支持收费适用于多可用区部署中的备用实例。

当您执行以下操作之一时，RDS 扩展支持的额外费用将自动停止：

- 升级到标准支持涵盖范围内的引擎版本。
- 删除在 RDS 标准支持终止日期之后运行主要版本的数据库。

如果您的目标引擎版本将来进入 RDS 扩展支持，将重新开始收费。

例如 RDS for PostgreSQL 11 将于 2024 年 3 月 1 日起进入扩展支持，但要等到 2024 年 4 月 1 日才会开始收费。您在 2024 年 4 月 30 日将 RDS for PostgreSQL 11 数据库升级到 RDS for PostgreSQL 12。您将只需在 RDS for PostgreSQL 11 上支付 30 天的扩展支持费用。在 RDS 标准支持终止日期

2025 年 2 月 28 日之后，您继续在此数据库实例上运行 RDS for PostgreSQL 12。从 2025 年 3 月 1 日起，将再次向您的数据库收取 RDS 扩展支持费用。

有关更多信息，请参阅 Amazon RDS for MySQL 定价和 Amazon RDS for PostgreSQL 定价。

避免支付 Amazon RDS 扩展支持费用

您可以通过阻止 RDS 在 RDS 标准支持终止日期后创建或还原数据库实例或多可用区数据库集群，来避免支付 RDS 扩展支持费用。为此，请使用 Amazon CLI 或 RDS API。

在 Amazon CLI 中，为 `--engine-lifecycle-support` 选项指定 `open-source-rds-extended-support-disabled`。在 RDS API 中，为 `LifeCycleSupport` 参数指定 `open-source-rds-extended-support-disabled`。有关更多信息，请参阅创建数据库集群或多可用区数据库集群或还原数据库实例或多可用区集群。

提供 Amazon RDS Extended Support 版本

RDS Extended Support 仅适用于主要版本。此功能不适用于次要版本。

RDS 扩展支持可用于 RDS for MySQL 和 RDS for PostgreSQL。有关更多信息，请参阅《Amazon RDS for PostgreSQL 发行说明》中的 主要版本 和 Amazon RDS for PostgreSQL 发布日历。

您还可以使用 Amazon CLI 或 RDS API 查看有关引擎版本支持日期的信息。有关更多信息，请参阅 在 Amazon RDS 扩展支持中查看引擎版本的支持日期。

Amazon RDS Extended Support 版本命名

Amazon RDS 将在 RDS Extended Support 中发布新的次要版本，其中包含针对引擎的修复程序和 CVE 补丁。有关更多信息，请参阅《Amazon RDS for PostgreSQL 发行说明》中的 适用于 RDS for MySQL 的 Amazon RDS 扩展支持版本和面向 RDS for PostgreSQL 的 Amazon RDS Extended Support 更新。

这些次要版本的名称将采用以下形式 major.minor-RDS.YYYYMMDD.patch.YYYYMMDD：例如，5.7.44-RDS.20240208.R2.20240210（适用于 RDS for MySQL）或 11.22-RDS.20240208.R2.20240210（适用于 RDS for PostgreSQL）。

major

对于 MySQL，主要版本号是版本号的整数和第一个小数点后的部分，例如 8.0。主要版本升级会增加版本号的主要版本部分。例如，从 5.7.44 升级到 8.0.33 是主要版本升级，其中 5.7 和 8.0 为主要版本号。

对于 PostgreSQL，主要版本号是整数，例如 11。

minor-RDS.YYYYMMDD

对于 MySQL，次要版本号是版本号的第三部分，例如 5.7.44-RDS.20240208 中的 44-RDS.20240208。

对于 PostgeSQL，次要版本号是版本号的第二部分，例如 11.22-RDS.20240208 中的 22-RDS.20240208。

该日期是 Amazon RDS 创建 Amazon RDS 次要版本的日期。

patch

补丁版本是在 Amazon RDS 创建 Amazon RDS 次要版本（例如 5.7.44-RDS.20240208.R2 或 11.22-RDS.20240208.R2 中的 R2）的日期之后的版本。

Amazon RDS 补丁版本包括在发布后添加到 Amazon RDS 次要版本中的重要错误修复。

YYYYMMDD

日期是 Amazon RDS 创建补丁版本的日期，例如 5.7.44-RDS.20240208.R2.20240210 或 11.22-RDS.20240208.R2.20240210 中的 20240210。

Amazon RDS 日期版本是一个安全补丁，它包括在发布后添加到次要版本中的重要安全修复程序。它不包括任何可能更改引擎行为的修复程序。

Amazon RDS 和客户对于 Amazon RDS 扩展支持的责任

以下内容描述了 Amazon RDS 对于 RDS 扩展支持的责任以及您对于 RDS 扩展支持的责任。

主题

Amazon RDS 的责任

在 RDS 标准支持终止日期之后，Amazon RDS 将为注册了 RDS 扩展支持的引擎提供补丁、错误修复和升级。这将持续长达 3 年，或者直到您停止使用引擎为止，以先发生者为准。

这些补丁将用于根据美国国家漏洞数据库（NVD）的 CVSS 严重性评级定义的严重和高 CVE。有关更多信息，请参阅漏洞指标。

您的责任

您负责为注册了 RDS 扩展支持的数据库实例或多可用区数据库集群应用补丁、错误修复和升级。Amazon RDS 保留随时更改、替换或撤回此类补丁、错误修复和升级的权利。如果需要某个补丁来解决安全或关键稳定性问题，Amazon RDS 将保留使用该补丁更新您的数据库实例、多可用区数据库集群或要求您安装该补丁的权利。

您还负责在 RDS 扩展支持终止日期之前，将引擎升级到更高的引擎版本。RDS 的扩展支持终止日期通常是 RDS 标准支持日期之后 3 年。有关数据库主要引擎版本的 RDS 扩展支持终止日期，请参阅主要版本和 Release calendar for Amazon RDS for PostgreSQL。

如果您不升级引擎，则在 RDS 扩展支持终止日期之后，Amazon RDS 会尝试将您的引擎升级到 RDS 标准支持所支持的更新引擎版本。如果升级失败，Amazon RDS 将保留以下权利：删除在 RDS 标准支持终止日期之后正在运行该引擎的数据库实例或多可用区数据库集群。但是，在这样做之前，Amazon RDS 将保留您来自该引擎的数据。

使用 Amazon RDS 扩展支持创建数据库实例或多可用区数据库集群

在创建数据库实例或多可用区数据库集群时，请在控制台中选择启用 RDS 扩展支持，或者使用 Amazon CLI 中的扩展支持选项或 RDS API 中的参数。当您在 Amazon RDS 扩展支持中注册数据库实例或多可用区数据库集群时，它将在数据库实例或多可用区数据库集群的生命周期内永久注册 RDS 扩展支持。

如果您使用控制台，则必须选择启用 RDS 扩展支持。默认情况下未选择该设置。

如果您使用 Amazon CLI 或 RDS API，但未指定 RDS 扩展支持设置，则 Amazon RDS 默认为启用 RDS 扩展支持。当您使用 Amazon CloudFormation 或其它服务自动执行操作时，此默认行为会在 RDS 标准支持终止日期之后保持数据库的可用性。

可以使用 Amazon CLI 或 RDS API 来创建数据库实例或多可用区数据库集群，来防止注册 RDS 扩展支持。

主题

- 使用 RDS 扩展支持创建数据库实例或多可用区数据库集群

RDS 扩展支持行为

下表汇总了当主引擎版本达到 RDS 标准支持终止时会发生的情况。

RDS 扩展支持状态*	行为
已启用	Amazon RDS 将向您收取 RDS 扩展支持费用。
已禁用	Amazon RDS 将您的数据库实例或多可用区数据库集群升级到支持的引擎版本。此升级在 RDS 标准支持终止日期或之后不久进行。

* 在 RDS 控制台中，RDS 扩展支持状态显示为“是”或“否”。在 Amazon CLI 或 RDS API 中，RDS 扩展支持状态显示为 `open-source-rds-extended-support` 或 `open-source-rds-extended-support-disabled`。

使用 RDS 扩展支持时的注意事项

在创建数据库集群或多可用区数据库集群之前，请考虑以下事项：

- 在 RDS 标准支持终止日期过去之后，您可以阻止创建新的数据库实例或新的多可用区数据库集群，并避免支付 RDS 扩展支持费用。为此，请使用 Amazon CLI 或 RDS API。在 Amazon CLI 中，为 `--engine-lifecycle-support` 选项指定 `open-source-rds-extended-support-disabled`。在 RDS API 中，为 `LifeCycleSupport` 参数指定 `open-source-rds-extended-support-disabled`。如果您指定 `open-source-rds-extended-support-disabled` 且 RDS 标准支持终止日期已过，则创建数据库实例或多可用区数据库集群始终会失败。
- RDS 扩展支持在集群级别进行设置。集群成员在 RDS 控制台、Amazon CLI 中的 `--engine-lifecycle-support`，RDS API 中的 `EngineLifecycleSupport`，始终具有相同的 RDS 扩展支持设置。

有关更多信息，请参阅MySQL 版本和 Amazon RDS for PostgreSQL 发行日历。

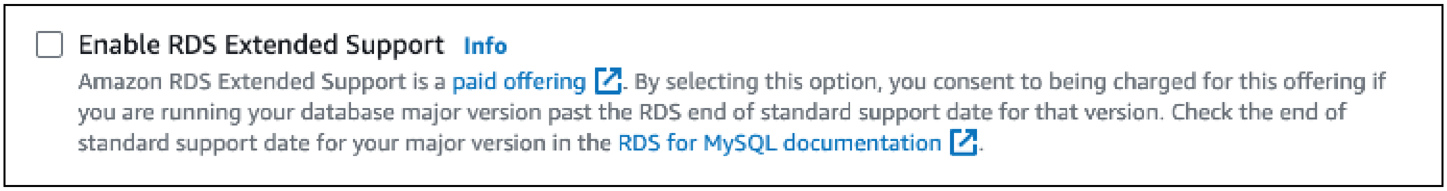
使用 RDS 扩展支持创建数据库实例或多可用区数据库集群

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 创建数据库实例或多可用区数据库集群

控制台

在创建数据库实例或多可用区数据库集群时，请在引擎选项部分中选择启用 RDS 扩展支持。默认情况下未选择该设置。

下图显示了启用 RDS 扩展支持设置：



Amazon CLI

在您运行 create-db-instance 或 create-db-cluster（多可用区数据库集群）Amazon CLI 命令时，请通过为 `--engine-lifecycle-support` 选项指定 `open-source-rds-extended-support` 来选择 RDS 扩展支持。默认情况下，此选项设置为 `open-source-rds-extended-support`。

要防止在 RDS 标准支持终止日期之后创建新的数据库实例或多可用区数据库集群，请为 `--engine-lifecycle-support` 选项指定 `open-source-rds-extended-support-disabled`。这样，您就可以避免支付任何关联的 RDS 扩展支持费用。

RDS API

当您使用 CreateDBInstance 或 CreateDBCluster（多可用区数据库集群）Amazon RDS API 操作时，请通过将 `EngineLifecycleSupport` 参数设置为 `open-source-rds-extended-support` 来选择 RDS 扩展支持。默认情况下，此参数设置为 `open-source-rds-extended-support`。

要防止在 RDS 标准支持终止日期之后创建新的数据库实例或多可用区数据库集群，请为 `EngineLifecycleSupport` 参数指定 `open-source-rds-extended-support-disabled`。这样，您就可以避免支付任何关联的 RDS 扩展支持费用。

有关更多信息，请参阅以下主题：

- 要创建数据库实例，请按照创建 Amazon RDS 数据库实例中用于您的数据库引擎的说明操作。
- 要创建多可用区数据库集群，请按 创建 Amazon RDS 的多可用区数据库集群 中数据库引擎的说明操作。

查看数据库实例或多可用区数据库集群在 Amazon RDS 扩展支持中的注册情况

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 查看数据库实例或多可用区数据库集群在 RDS 扩展支持中的注册情况。

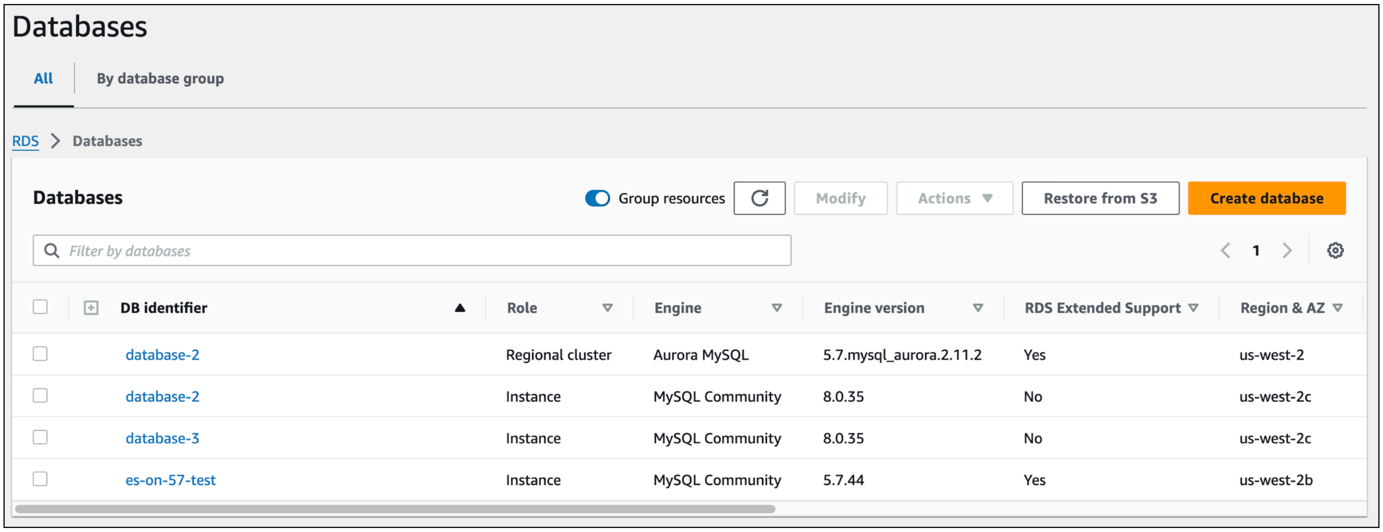
> **Note**
>
> 控制台中的 RDS 扩展支持列、Amazon CLI 中的 `--engine-lifecycle-support` 选项和 RDS API 中的 `EngineLifecycleSupport` 参数仅表示在 RDS 扩展支持中进行注册。RDS 扩展支持仅在数据库引擎版本已达到 RDS 标准支持终止时才开始收费。有关更多信息，请参阅《Amazon RDS for PostgreSQL 发行说明》中的 [主要版本](#) 和 [Amazon RDS for PostgreSQL 发布日历](#)。
>
> 例如，您有一个已注册了 RDS 扩展支持的 RDS for MySQL 5.7 数据库。2024 年 3 月 1 日，Amazon RDS 开始向您收取此数据库的 RDS 扩展支持费用。2024 年 7 月 31 日，您将此数据库升级到 RDS for MySQL 8.0。此数据库的 RDS 扩展支持状态保持为已启用。但是，对该数据库的 RDS 扩展支持收费已停止，因为 RDS for MySQL 8.0 尚未达到 RDS 标准支持终止。在 2026 年 8 月 1 日之前，Amazon RDS 都不会向您收取该数据库的 RDS 扩展支持费用，该日期是对 RDS for MySQL 8.0 的 RDS 标准支持终止的时间。

控制台

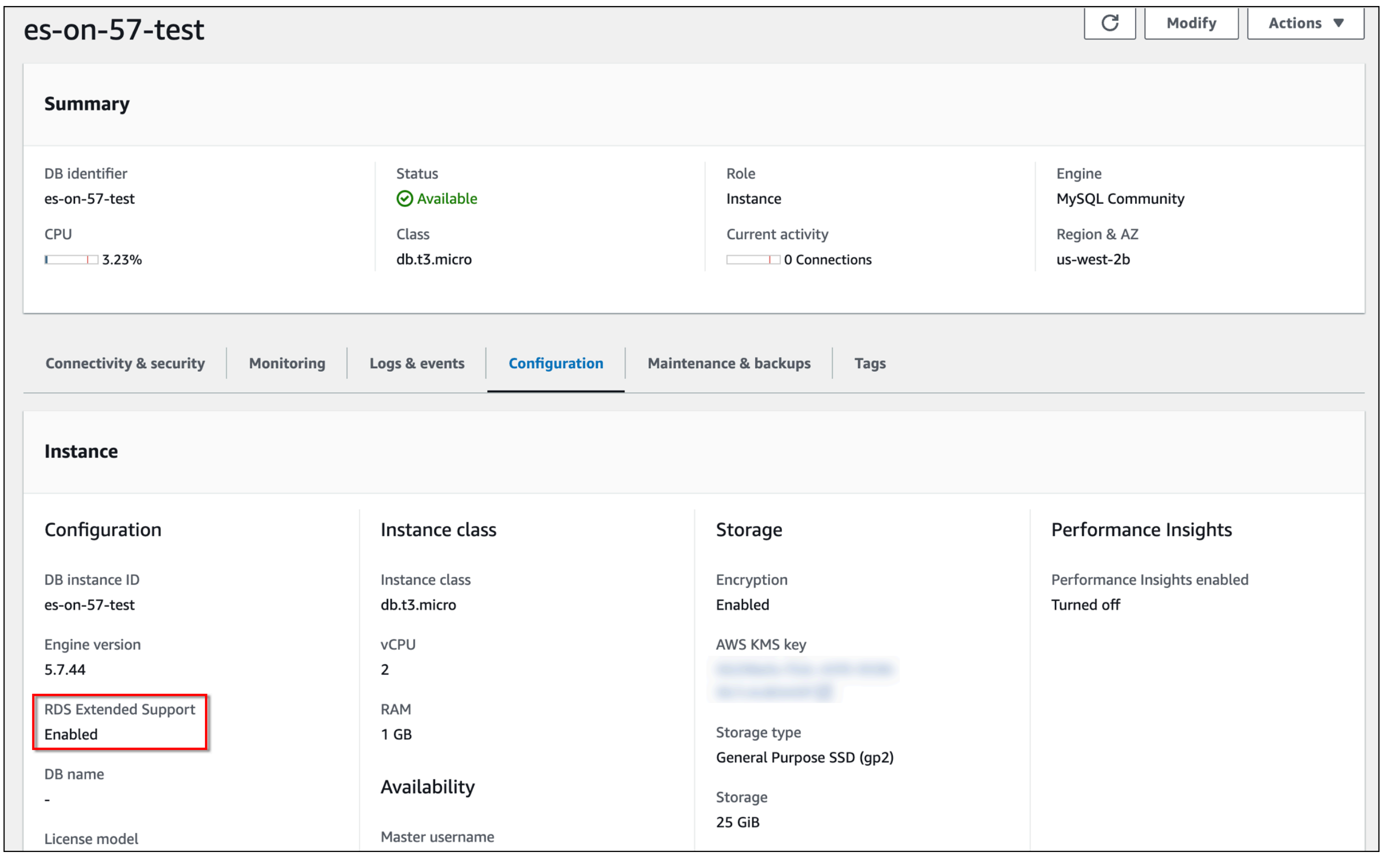
查看数据库实例或多可用区数据库集群在 RDS 扩展支持中的注册情况

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases（数据库）。RDS 扩展支持下的值指示数据库实例或多可用区数据库集群是否已在 RDS 扩展支持中注册。如果没有显示任何值，则 RDS 扩展支持不适用于您的数据库。

   > **Tip**
   >
   > 如果未出现 RDS 扩展支持列，请选择首选项图标，然后打开 RDS 扩展支持。



3. 您还可以在配置选项卡上查看每个数据库的注册情况。在数据库标识符下选择一个数据库。在配置选项卡上，在扩展支持下查看数据库是否已注册。



Amazon CLI

要使用 Amazon CLI 查看您的数据库在 RDS 扩展支持中的注册情况，请运行 describe-db-instances 或 describe-db-clusters（多可用区数据库集群）命令。

如果 RDS 扩展支持对数据库可用，则响应中包含参数 `EngineLifecycleSupport`。值 `open-source-rds-extended-support` 指示数据库实例或多可用区数据库集群已在 RDS 扩展支持中注册。值 `open-source-rds-extended-support-disabled` 指示已禁用在 RDS 扩展支持中注册数据库实例或多可用区数据库集群。

示例

以下命令会返回所有数据库实例的信息：

```
aws rds describe-db-instances
```

以下响应显示在数据库实例 `database-1` 上运行的 PostgreSQL 引擎已注册到 RDS 扩展支持：

```
{
    "DBInstanceIdentifier": "database-1",
    "DBInstanceClass": "db.t3.large",
    "Engine": "postgres",
    ...
    "EngineLifecycleSupport": "open-source-rds-extended-support"
}
```

RDS API

要使用 Amazon RDS API 查看您的数据库在 RDS 扩展支持中的注册情况，请使用 DescribeDBInstances 或 DescribeDBClusters 操作。

如果 RDS 扩展支持对数据库可用，则响应中包含参数 `EngineLifecycleSupport`。值 `open-source-rds-extended-support` 指示数据库实例或多可用区数据库集群已在 RDS 扩展支持中注册。值 `open-source-rds-extended-support-disabled` 指示已禁用在 RDS 扩展支持中注册数据库实例或多可用区数据库集群。

在 Amazon RDS 扩展支持中查看引擎版本的支持日期

您可以使用 Amazon CLI 或 RDS API 在 Amazon RDS 扩展支持中查看有关数据库实例或多可用区数据库集群的引擎版本的支持日期的信息。利用此信息，您可以更好地规划升级。

Amazon CLI 命令和 RDS API 操作返回 RDS 标准支持和 RDS 扩展支持的开始日期和结束日期。这些日期也可以在主要引擎版本表中找到。有关更多信息，请参阅《Amazon RDS for PostgreSQL 发行说明》中的 Amazon RDS 上支持的 MySQL 主要版本和 Amazon RDS for PostgreSQL 发布日历。

Amazon CLI

要使用 Amazon CLI 查看主要引擎版本的 RDS 标准支持和 RDS 扩展支持的开始日期和结束日期，请运行 describe-db-major-engine-versions 命令。

此命令将返回开源引擎（MariaDB、MySQL 和 PostgreSQL）的以下相关参数。它不会为商用引擎（Db2、SQL Server 和 Oracle）返回这些参数。

- `SupportedEngineLifecycles` – 此参数是一个对象数组，其中包括 `LifecycleSupportName`、`LifecycleSupportStartDate` 和 `LifecycleSupportEndDate`。
- `LifecycleSupportName` – 此参数表示引擎版本所处的支持类型：RDS 标准支持（`open-source-rds-standard-support`）或 RDS 扩展支持（`open-source-rds-extended-support`）。对于 MariaDB，此参数仅返回 RDS 标准支持（`open-source-rds-standard-support`）。
- `LifecycleSupportStartDate` – 此参数根据 `LifecycleSupportName` 的值，列出主要引擎版本的 RDS 标准支持或 RDS 扩展支持的开始日期。
- `LifecycleSupportEndDate` – 此参数根据 `LifecycleSupportName` 的值，列出主要引擎版本的 RDS 标准支持或 RDS 扩展支持的结束日期。

示例

响应示例显示以下信息：

- MySQL 5.7 的支持的引擎生命周期 `open-source-rds-standard-support` 和 `open-source-rds-extended-support` 的开始日期和结束日期。现已提供 MySQL 5.7 的 RDS 扩展支持。
- MariaDB 10.6 的支持的引擎生命周期 `open-source-rds-standard-support` 的开始日期和结束日期。尚未提供 MariaDB 10.6 的 RDS 扩展支持。
- 未提供有关 SQL Server Enterprise Edition 13 的支持的引擎生命周期的信息，因为 SQL Server 是商用引擎。

```
{
    "DBMajorEngineVersions": [
```

```
        {
            "Engine": "mysql",
            "MajorEngineVersion": "5.7",
            "SupportedEngineLifecycles": [
                {
                    "LifecycleSupportName": "open-source-rds-standard-support",
                    "LifecycleSupportStartDate": "2016-02-22T00:00:00+00:00",
                    "LifecycleSupportEndDate": "2024-02-29T23:59:59.999000+00:00"
                },
                {
                    "LifecycleSupportName": "open-source-rds-extended-support",
                    "LifecycleSupportStartDate": "2024-03-01T00:00:00+00:00",
                    "LifecycleSupportEndDate": "2027-02-28T23:59:59.999000+00:00"
                }
            ]
        },
        {
            "Engine": "mariadb",
            "MajorEngineVersion": "10.6",
            "SupportedEngineLifecycles": [
                {
                    "LifecycleSupportName": "open-source-rds-standard-support",
                    "LifecycleSupportStartDate": "2022-02-03T00:00:00+00:00",
                    "LifecycleSupportEndDate": "2026-07-31T23:59:59.999000+00:00"
                }
            ]
        },
        {
            "Engine": "sqlserver-ee",
            "MajorEngineVersion": "13.00"
        },
        ...
    ]
}
```

RDS API

要使用 RDS API 查看主要引擎版本的 RDS 标准支持和 RDS 扩展支持的开始日期和结束日期，请使用 DescribeDBMajorEngineVersions 操作。

如果已为某个引擎版本提供 RDS 扩展支持，则响应将包含参数 `SupportedEngineLifeCycles` 作为带两个对象的数组。一个对象包含 RDS 标准支持的开始日期和结束日期。第二个对象包含 RDS 扩展支持的开始日期和结束日期。

如果未提供开源引擎版本（MariaDB、MySQL 和 PostgreSQL）的 RDS 扩展支持，则响应仅包含参数 `SupportedEngineLifeCycles` 作为带单个对象的数组。该对象包含 RDS 标准支持的开始日期和结束日期。

如果引擎版本适用于商用引擎（Db2、SQL Server 和 Oracle），则响应不包含参数 `SupportedEngineLifeCycles`。

使用 Amazon RDS 扩展支持还原数据库实例或多可用区集群

在还原数据库实例或多可用区集群时，请在控制台中选择启用 RDS 扩展支持，或者使用 Amazon CLI 中的扩展支持选项或使用 RDS API 中的参数。当您在 RDS 扩展支持中注册数据库实例或多可用区数据库集群时，它将在数据库实例或多可用区数据库集群的生命周期内永久注册 RDS 扩展支持。

RDS 扩展支持设置的默认值取决于您是使用控制台、Amazon CLI 还是 RDS API 来还原数据库。如果您使用控制台，但未选择启用 RDS 扩展支持，并且您要还原的主引擎版本已超过 RDS 标准支持终止，则 Amazon RDS 会自动将您的数据库实例升级到更高的引擎版本。如果您使用 Amazon CLI 或 RDS API，但未指定 RDS 扩展支持设置，则 Amazon RDS 默认为启用 RDS 扩展支持。当您使用 Amazon CloudFormation 或其他服务自动执行操作时，此默认行为会在 RDS 标准支持终止日期之后保持数据库的可用性。可以使用 Amazon CLI 或 RDS API 来禁用 RDS 扩展支持。

主题

RDS 扩展支持行为

下表汇总了当要还原的数据库实例或多可用区数据库集群的主引擎版本已达到 RDS 标准支持终止时会发生的情况。

RDS 扩展支持状态*	行为
已启用	Amazon RDS 将向您收取 RDS 扩展支持费用。
已禁用**	还原完成后，Amazon RDS 会自动将数据库实例或多可用区数据库集群升级到更高的引擎版本（在将来的维护时段中）。

* 在 RDS 控制台中，RDS 扩展支持状态显示为“是”或“否”。在 Amazon CLI 或 RDS API 中，RDS 扩展支持状态显示为 `open-source-rds-extended-support` 或 `open-source-rds-extended-support-disabled`。

** 此选项仅在还原运行 PostgreSQL 12 及更高版本或 MySQL 8 及更高版本的数据库实例或多可用区数据库集群时才可用。

使用 RDS 扩展支持时的注意事项

在还原数据库实例或多可用区集群之前，请考虑以下事项：

- 在 RDS 标准支持终止日期过去之后，如果您想从 Amazon S3 还原数据库实例或多可用区集群，您只能使用 Amazon CLI 或 RDS API 执行还原操作。使用 restore-db-cluster-from-s3 Amazon CLI 命令中的 `--engine-lifecycle-support` 选项，或 RestoreDBClusterFromS3 RDS API 操作中的 `EngineLifecycleSupport` 参数。
- 如果您想阻止 RDS 将您的数据库还原到 RDS 扩展支持版本，请在 Amazon CLI 或 RDS API 中指定 `open-source-rds-extended-support-disabled`。这样，您就可以避免支付任何关联的 RDS 扩展支持费用。

  如果您指定此设置，Amazon RDS 会自动将您还原的数据库升级到新的、受支持的主要版本。如果升级未通过升级前检查，Amazon RDS 将安全地回滚到 RDS 扩展支持引擎版本。该数据库将保持 RDS 扩展支持模式，并且在您手动升级数据库之前，Amazon RDS 将向您收取 RDS 扩展支持费用。

  例如，如果您在不使用 RDS 扩展支持的情况下还原 MySQL 5.7 快照，Amazon RDS 将尝试自动将您的数据库升级到 MySQL 8.0。如果此升级由于存在您需要解决的问题而失败，Amazon RDS 会将数据库回滚到 MySQL 5.7。在您可以修复该问题之前，Amazon RDS 会将数据库保留在 RDS 扩展支持上。例如，升级可能由于存储空间不足而失败。修复问题后，必须启动升级。首次尝试升级数据库后，Amazon RDS 将不再尝试对其进行升级。
- RDS 扩展支持在集群级别进行设置。集群成员在 RDS 控制台、Amazon CLI 中的 `--engine-lifecycle-support`，RDS API 中的 `EngineLifecycleSupport`，始终具有相同的 RDS 扩展支持设置。

有关更多信息，请参阅MySQL 版本和 Amazon RDS for PostgreSQL 发行日历。

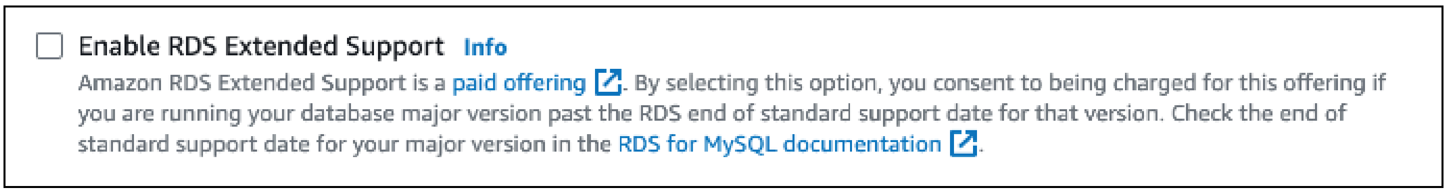
使用 RDS 扩展支持还原数据库实例或多可用区数据库集群

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API，借助 RDS 扩展支持版本还原数据库实例或多可用区集群

控制台

在还原数据库实例或多可用区数据库集群时，请在引擎选项部分选择启用 RDS 扩展支持。如果您未选择此设置，并且要还原的主引擎版本已超过 RDS 标准支持终止，则 Amazon RDS 会自动将您的数据库实例或多可用区数据库集群升级到处于 RDS 标准支持下的版本。

下图显示了启用 RDS 扩展支持设置：



Amazon CLI

在您运行 restore-db-instance-from-db-snapshot 或 restore-db-cluster-from-snapshot Amazon CLI 命令时，请通过为 `--engine-lifecycle-support` 选项指定 `open-source-rds-extended-support` 来选择 RDS 扩展支持。

如果您想避免支付与 RDS 扩展支持关联的费用，请将 `--engine-lifecycle-support` 选项设置为 `open-source-rds-extended-support-disabled`。默认情况下，此选项设置为 `open-source-rds-extended-support`。

您还可以使用以下 Amazon CLI 命令指定该值：

- restore-db-cluster-from-s3
- restore-db-cluster-to-point-in-time
- restore-db-instance-from-s3
- restore-db-instance-to-point-in-time

RDS API

在使用 RestoreDBInstanceFromDBSnapshot 或 RestoreDBClusterFromSnapshot Amazon RDS API 操作时，请通过将 `EngineLifecycleSupport` 参数设置为 `open-source-rds-extended-support` 来选择 RDS 扩展支持。

如果您想避免支付与 RDS 扩展支持关联的费用，请将 `EngineLifecycleSupport` 参数设置为 `open-source-rds-extended-support-disabled`。默认情况下，此参数设置为 `open-source-rds-extended-support`。

您还可以使用以下 API 操作指定该值：

- RestoreDBClusterFromS3
- RestoreDBClusterToPointInTime
- RestoreDBInstanceFromS3
- RestoreDBInstanceToPointInTime

有关还原数据库实例或多可用区数据库集群的更多信息，请按照还原到数据库实例中数据库引擎的相关说明操作。

使用 Amazon RDS 蓝绿部署进行数据库更新

蓝绿部署将生产数据库环境复制到单独的同步暂存环境。通过使用 Amazon RDS 蓝绿部署，您可以在不影响生产环境的情况下对暂存环境中的数据库进行更改。例如，您可以升级主数据库引擎版本或次要数据库引擎版本，或在暂存环境中更改数据库参数。准备就绪后，可以将暂存环境提升为新的生产数据库环境，在此环境中，停机时间通常不到一分钟。

> **Note**
>
> 目前，仅 RDS for MariaDB、RDS for MySQL 和 RDS for PostgreSQL 支持蓝绿部署。有关 Amazon Aurora 可用性，请参阅《Amazon Aurora 用户指南》中的使用 Amazon Aurora 蓝绿部署进行数据库更新。

主题

Amazon RDS 蓝绿部署概述

通过使用 Amazon RDS 蓝绿部署，您可以进行数据库更改并测试，然后再在生产环境中实施这些更改。蓝绿部署 会创建一个复制生产环境的暂存环境。在蓝绿部署中，蓝色环境 是当前的生产环境。绿色环境 是暂存环境，与当前生产环境保持同步。

您可以在绿色环境中更改 RDS 数据库实例，而不会影响生产工作负载。例如，您可以升级主要或次要数据库引擎版本、升级基础文件系统配置或在暂存环境中更改数据库参数。您可以彻底测试绿色环境中的变化。准备就绪后，您可以切换环境，以将绿色环境转换为新的生产环境。切换通常需要不到一分钟，不会丢失数据，也无需更改应用程序。

由于绿色环境是生产环境拓扑的副本，因此绿色环境包括数据库实例使用的功能。这些功能包括只读副本、存储配置、数据库快照、自动备份、性能详情和增强监控。如果蓝色数据库实例为多可用区数据库实例部署，则绿色数据库实例也是多可用区数据库实例部署。

> **Note**
>
> 目前，仅支持 RDS for MariaDB、RDS for MySQL 和 RDS for PostgreSQL 蓝绿部署。有关 Amazon Aurora 可用性，请参阅《Amazon Aurora 用户指南》中的 Amazon Aurora 蓝绿部署概述。
>
> 在某些情况下，RDS for PostgreSQL 使用逻辑复制而不是物理复制来保持绿色环境与蓝色环境同步。有关更多信息，请参阅 the section called “PostgreSQL 复制方法”。

主题

- 区域和版本可用性
- 使用 Amazon RDS 蓝绿部署的优势
- 蓝绿部署的工作流
- 授权访问 Amazon RDS 蓝绿部署操作
- Amazon RDS 蓝绿部署的限制和注意事项
- Amazon RDS 蓝绿部署最佳实践
- PostgreSQL 蓝绿部署的复制方法

区域和版本可用性

功能可用性和支持因每个数据库引擎的特定版本以及 Amazon Web Services 区域而异。有关更多信息，请参阅 the section called “蓝绿部署”。

使用 Amazon RDS 蓝绿部署的优势

通过使用 Amazon RDS 蓝绿部署，您可以随时了解最新的安全补丁，提高数据库性能，并在短暂且可预测的停机时间内采用更新的数据库功能。蓝绿部署降低了数据库更新（例如主要或次要引擎版本升级）的风险和减少了停机时间。

蓝绿部署提供以下优势：

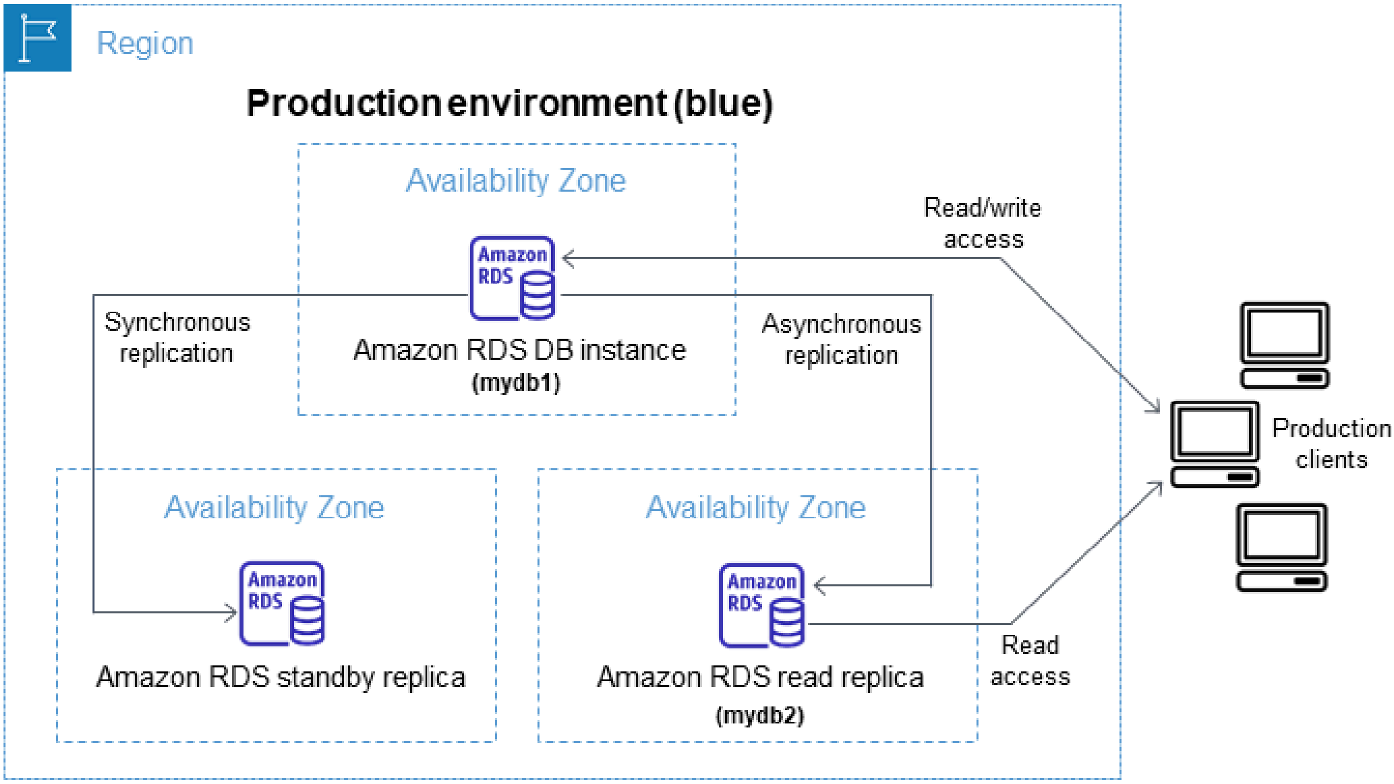
- 轻松创建生产就绪的暂存环境。
- 自动将数据库更改从生产环境复制到暂存环境。
- 在不影响生产环境的情况下在安全的暂存环境中测试数据库更改。
- 通过数据库补丁和系统更新保持最新状态。
- 实施和测试更新的数据库功能。
- 在不更改应用程序的情况下，将您的暂存环境切换为新的生产环境。
- 使用内置切换防护机制安全切换。
- 消除切换期间的数据丢失。
- 快速切换，通常不到一分钟，具体取决于您的工作负载。

蓝绿部署的工作流

使用蓝绿部署进行数据库更新时，请完成以下主要步骤。

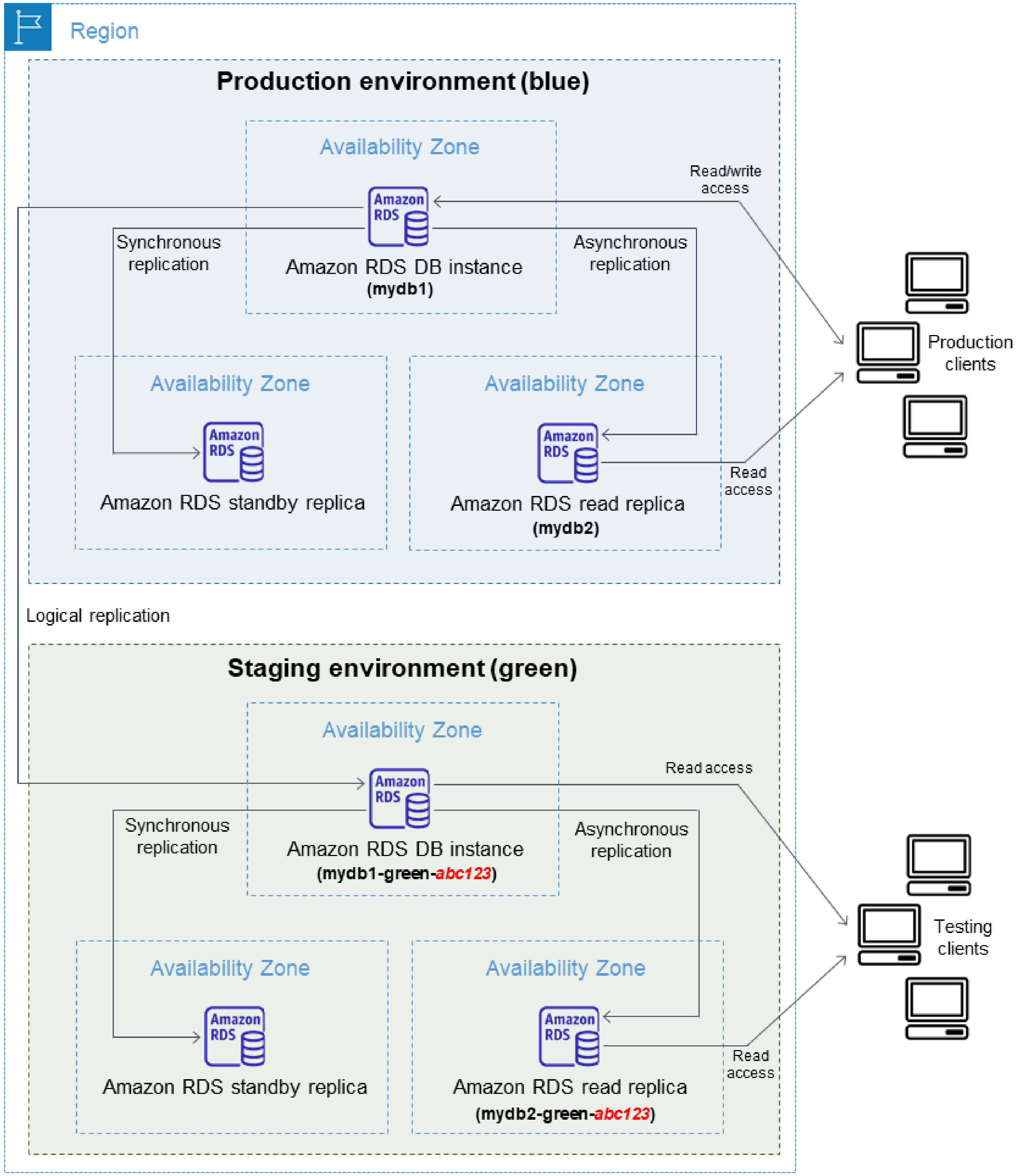
1. 确定需要更新的生产环境。

   例如，此映像中的生产环境具有多可用区数据库实例部署（mydb1）和只读副本（mydb2）。



2. 创建蓝绿部署。有关说明，请参阅[在 Amazon RDS 中创建蓝绿部署]。

   下图显示了步骤 1 中生产环境的蓝绿部署示例。在创建蓝绿部署时，RDS 会复制主数据库实例的完整拓扑和配置以创建绿色环境。复制的数据库实例名称附加了 `-green-random-characters`。映像中的暂存环境包含多可用区数据库实例部署（mydb1-green-*abc123*）和只读副本（mydb2-green-*abc123*）。



创建蓝绿部署时，您可以升级数据库引擎版本并为绿色环境中的数据库实例指定不同的数据库参数组。RDS 还配置从蓝色环境中的主数据库实例到绿色环境中的主数据库实例的复制。

创建蓝绿部署后，默认情况下，绿色环境中的数据库实例为只读状态。

3. 如果需要，对暂存环境进行其他更改。例如，您可以在绿色环境中更改一个或多个数据库实例使用的数据库实例类。

   有关修改数据库实例的信息，请参阅修改 Amazon RDS 数据库实例。

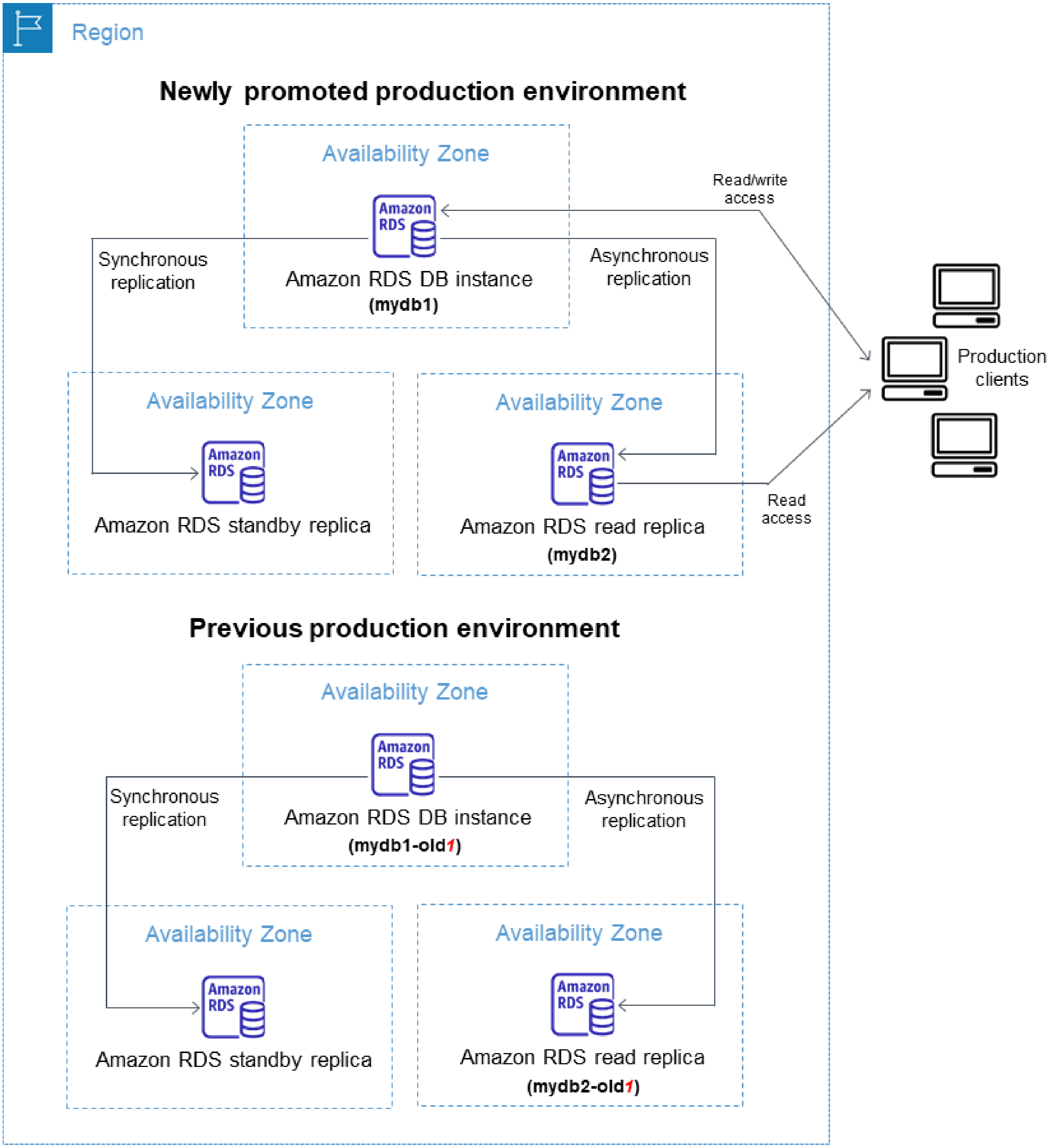
4. 测试您的暂存环境。

   在测试期间，建议您将绿色环境中的数据库保持为只读状态。在绿色环境中启用写入操作需谨慎，因为它们可能导致复制冲突。它们还可能导致切换后生产数据库中出现意外数据。要对 RDS for MySQL 启用写入操作，请将 `read_only` 参数设置为 `0`，然后重启数据库实例。对于使用逻辑复制的 RDS for PostgreSQL 部署，在会话级别将 `default_transaction_read_only` 参数设置为 `off`。对于那些使用物理复制的部署，不能对绿色环境启用写入操作。

5. 准备就绪后，切换以将暂存环境转换为新的生产环境。有关说明，请参阅切换 Amazon RDS 中的蓝绿部署。

   切换会导致停机。停机时间通常不到一分钟，但根据您的工作负载，停机时间可能会更长。

   下图显示了切换后的数据库实例。



切换后，处于绿色环境中的数据库实例将成为新的生产数据库实例。当前生产环境中的名称和端点分配给新转换的生产环境，无需更改您的应用程序。因此，您的生产流量现在流向新的生产环境。

先前蓝色环境中的数据库实例通过向当前名称附加 `-old`*n*（其中 *n* 是一个数字）来重命名。例如，假设蓝色环境中数据库实例的名称为 `mydb1`。切换之后，数据库实例名称将为 `mydb1-old1`。

在本例的映像中，切换期间会发生以下变化：

- 名为 `mydb1-green-abc123` 的绿色环境多可用区数据库实例部署将成为名为 `mydb1` 的生产多可用区数据库实例部署。
- 名为 `mydb2-green-abc123` 的绿色环境只读副本成为生产只读副本 `mydb2`。
- 名为 `mydb1` 的蓝色环境多可用区数据库实例部署成为 `mydb1-old1`。
- 名为 `mydb2` 的蓝色环境只读副本成为 `mydb2-old1`。

6. 如果您不再需要蓝绿部署，可将其删除。有关说明，请参阅删除 Amazon RDS 中的蓝绿部署。

   切换后，之前的生产环境不会被删除，因此如有必要，您可以将其用于回归测试。

授权访问 Amazon RDS 蓝绿部署操作

用户必须具有所需的权限才能执行与蓝绿部署相关的操作。您可以创建 IAM policy，以便为用户和角色授予权限以对所需的指定资源执行特定的 API 操作。然后，可以将这些策略附加到需要这些权限的 IAM 权限集或角色。有关更多信息，请参阅 Amazon RDS 的 Identity and Access Management。

创建蓝绿部署的用户必须具有执行以下 RDS 操作的权限：

- `rds:CreateBlueGreenDeployment`
- `rds:AddTagsToResource`
- `rds:CreateDBInstanceReadReplica`

切换蓝绿部署的用户必须具有执行以下 RDS 操作的权限：

- `rds:SwitchoverBlueGreenDeployment`
- `rds:ModifyDBInstance`
- `rds:PromoteReadReplica`

删除蓝绿部署的用户必须具有执行以下 RDS 操作的权限：

- `rds:DeleteBlueGreenDeployment`
- `rds:DeleteDBInstance`

Amazon RDS 代表您预调配和修改暂存环境中的资源。这些资源包括使用内部定义命名约定的数据库实例。因此，附加的 IAM 策略不能包含部分资源名称模式，例如 `my-db-prefix-*`。仅支持通配符（*）。通常，我们建议使用资源标签和其它支持的属性来控制对这些资源的访问权限，而不是使用通配符。有关更多信息，请参阅 Actions, resources, and condition keys for Amazon RDS。

Amazon RDS 蓝绿部署的限制和注意事项

要在 Amazon RDS 中进行蓝绿部署，需要仔细考虑复制插槽、资源管理、实例大小以及对数据库性能的潜在影响等因素。以下各节提供的指导有助于您优化部署策略，以确保尽可能地减少停机时间、实现无缝过渡并有效地管理数据库环境。

主题

蓝绿部署的限制

以下限制适用于蓝绿部署。

主题

蓝绿部署的一般限制

以下一般限制适用于蓝绿部署：

- 蓝绿部署不支持使用 Amazon Secrets Manager 管理主用户密码。
- 如果在蓝色数据库上启用了专用日志卷（DLV，Dedicated Log Volume），则必须在所有数据库实例（包括只读副本）上启用专用日志卷。
- 在切换期间，蓝色和绿色环境无法与 Amazon Redshift 进行零 ETL 集成。您必须先删除集成，接着切换，然后重新创建集成。
- 创建蓝绿部署时，必须在绿色环境中禁用事件调度器（`event_scheduler` 参数）。这样可以防止在绿色环境中生成事件并导致不一致。

- 您无法将未加密的数据库实例更改为加密的数据库实例。此外，无法将加密的数据库实例更改为未加密的数据库实例。
- 无法将蓝色数据库实例更改为比其相应的绿色数据库实例更高的引擎版本。
- 蓝色环境和绿色环境中的资源必须位于同一个 Amazon Web Services 账户中。
- 以下功能不支持蓝绿部署：
  - Amazon RDS 代理
  - 级联只读副本
  - 跨区域只读副本
  - Amazon CloudFormation
  - 多可用区数据库集群部署

  多可用区数据库实例部署支持蓝绿部署。有关多可用区部署的更多信息，请参阅配置和管理 Amazon RDS 的多可用区部署。

RDS for MySQL 蓝绿部署的限制

以下限制适用于 RDS for MySQL 蓝绿部署：

- 蓝色数据库实例不能是外部二进制日志副本。
- 如果源数据库与自定义选项组相关联，则在创建蓝绿部署时无法指定主要版本升级。

  在这种情况下，您可以创建蓝绿部署，而无需指定主要版本升级。然后，您可以在绿色环境中升级数据库。有关更多信息，请参阅 升级数据库实例引擎版本。
- 蓝绿部署不支持适用于 MySQL 的 Amazon JDBC 驱动程序。有关更多信息，请参阅 GitHub 上的 Known Limitations。

采用物理复制的 RDS for PostgreSQL 蓝绿部署的限制

以下限制适用于使用物理复制的 RDS for PostgreSQL 蓝绿部署。有关蓝绿部署何时使用物理复制而不是逻辑复制的说明，请参阅 the section called “PostgreSQL 复制方法”。

- 创建绿色环境后，您无法执行手动主要版本升级。
- 使用物理复制的蓝绿部署不支持绿色环境中的架构更改，因为它严格来说是只读的。
- 蓝色数据库实例不能是逻辑源（发布者）或副本（订阅用户）。

RDS for PostgreSQL 蓝绿部署的限制

以下限制适用于使用逻辑复制的 RDS for PostgreSQL 蓝绿部署。有关蓝绿部署何时使用逻辑复制而不是物理复制的说明，请参阅 the section called “PostgreSQL 复制方法”。

- Unlogged（未记录的）表不会复制到绿色环境。
- 蓝色数据库实例不能是逻辑源（发布者）或副本（订阅者）。
- 如果将蓝色数据库实例配置为外部数据包装器（FDW）扩展的外部服务器，则必须使用实例端点名称而不是 IP 地址。这会让配置在切换后仍能正常运行。
- 在蓝绿部署中，每个数据库都需要一个逻辑复制插槽。随着数据库数量增长，资源开销会增加，并可能导致复制滞后，尤其是在数据库实例没有充分扩展的情况下。影响取决于数据库工作负载和连接数等因素。要缓解这种情况，可以考虑纵向扩展数据库实例类，或减少源实例上的数据库数量。
- 绿色环境中的逻辑复制 apply process 是单线程的。如果蓝色环境生成大量写入流量，则绿色环境可能无法跟上。这可能会导致复制滞后或失败，特别是对于产生持续高写入吞吐量的工作负载。确保彻底测试工作负载。对于需要主要版本升级和处理大量写入工作负载的场景，可以考虑使用其它方法，例如使用 Amazon Database Migration Service（Amazon DMS）。
- 以下限制适用于 PostgreSQL 扩展：
  - 创建蓝绿部署时，必须在蓝色环境中禁用 `pg_partman` 扩展。该扩展将执行 `CREATE TABLE` 等 DDL 操作，这会中断从蓝色环境到绿色环境的逻辑复制。
  - 创建蓝绿部署后，`pg_cron` 扩展必须在所有绿色数据库上保持禁用状态。该扩展具有以超级用户身份运行并绕过绿色环境只读设置的后台工作进程，这可能会导致复制冲突。
  - 创建蓝绿部署时，必须在蓝色环境中禁用 `pglogical` 和 `pgactive` 扩展。将绿色环境切换为新的生产环境后，您可以重新启用这些扩展。此外，蓝色数据库不能是外部实例的逻辑订阅者。
  - 如果您使用的是 `pgAudit` 扩展，则它必须保留在蓝色和绿色数据库实例的自定义数据库参数组上的共享库（`shared_preload_libraries`）中。有关更多信息，请参阅 the section called “设置 pgAudit 扩展”。

蓝绿部署的特定于逻辑复制的限制

PostgreSQL 有某些与逻辑复制相关的限制，这会导致创建 RDS for PostgreSQL 数据库实例的蓝绿部署存在限制。

下表描述了适用于 RDS for PostgreSQL 的蓝绿部署的逻辑复制限制。有关更多信息，请参阅 PostgreSQL 逻辑复制文档中的限制。

限制	说明
数据定义语言（DDL）语句，（例如 CREATE TABLE 和 CREATE SCHEMA）不会从蓝色环境复制到绿色环境。	如果 Amazon RDS 在蓝色环境中检测到 DDL 更改，则您的绿色数据库将进入复制已降级状态。必须删除蓝绿部署和所有绿色数据库，然后重新创建。
对序列对象执行的 NEXTVAL 操作在蓝色环境和绿色环境之间不同步。	在切换期间，Amazon RDS 会增加绿色环境中的序列值，以匹配蓝色环境中的那些序列值。如果您有成千上万的序列，这可能会延迟切换。
蓝色环境中的大型对象不会复制到绿色环境中。这既包括现有的大型对象，也包括在蓝绿部署过程中任何新创建或修改的大型对象。	如果 Amazon RDS 检测到在蓝色环境中创建或修改了存储在 pg_largeobject 系统表中的大型对象，则您的绿色数据库将进入复制已降级状态。必须删除蓝绿部署和绿色数据库，然后重新创建。
在绿色环境中，实体化视图不会自动刷新。	蓝色环境中刷新的实体化视图不会在绿色环境中刷新。切换后，可以使用 REFRESH MATERIALIZED VIEW 命令手动刷新实体化视图，也可以计划进行刷新。
不允许对没有主键的表执行 UPDATE 和 DELETE 操作。	在创建蓝绿部署之前，请确保所有表都有主键或使用 REPLICA IDENTITY FULL。但是，仅当主键或唯一键不存在时才使用 REPLICA IDENTITY FULL，因为它会影响复制性能。有关更多信息，请参阅 PostgreSQL 文档。

蓝绿部署注意事项

Amazon RDS 使用每种资源的 `DbiResourceId` 跟踪蓝绿部署中的资源。此资源 ID 是资源的 Amazon Web Services 区域唯一的不可变标识符。

资源 ID 与数据库实例 ID 是分开的。每一个都列在 RDS 控制台的数据库配置中。

当您切换蓝绿部署时，资源的名称（实例 ID）会发生变化，但每个资源都保持相同的资源 ID。例如，在蓝色环境中，数据库实例标识符可能为 `mydb`。切换后，同一个数据库实例可能重命名为 `mydb-old1`。但是，数据库实例的资源 ID 在切换期间不会更改。因此，当将绿色资源切换为新的生产资源时，它们的资源 ID 与之前生产中的蓝色资源 ID 不匹配。

切换蓝绿部署后，请考虑将资源 ID 更新为新转换的生产资源的 ID，以获得与生产资源一起使用的集成功能和服务。具体而言，请考虑以下更新：

- 如果您使用 RDS API 和资源 ID 执行筛选，请在切换后调整筛选中使用的资源 ID。
- 如果您使用 CloudTrail 来审计资源，请调整 CloudTrail 的使用者，以便在切换后跟踪新的资源 ID。有关更多信息，请参阅 监控 Amazon CloudTrail 中的 Amazon RDS API 调用。
- 如果您使用性能详情API，请在切换后调整 API 调用中的资源 ID。有关更多信息，请参阅 在 Amazon RDS 上使用性能详情监控数据库负载。

  您可以在切换后监控同名数据库，但它不包含切换之前的数据。
- 如果您在 IAM 策略中使用资源 ID，请确保在必要时添加新转换的资源的资源 ID。有关更多信息，请参阅 Amazon RDS 的 Identity and Access Management。
- 如果您有与数据库实例关联的 IAM 角色，请务必在切换后重新关联它们。附加的角色不会自动复制到绿色环境。
- 如果您使用 IAM 数据库身份验证对数据库实例进行身份验证，请确保用于数据库访问的 IAM 策略同时包含在策略的 `Resource` 元素下方列出的蓝色和绿色数据库。这是在切换后连接到绿色数据库所必需的。有关更多信息，请参阅 the section called “创建和使用适用于 IAM 数据库访问的 IAM 策略”。
- 如果您在蓝绿部署中使用 Amazon Backup 管理资源的自动备份，请在切换后调整 Amazon Backup 使用的资源 ID。有关更多信息，请参阅 使用 Amazon Backup 管理 Amazon RDS 的自动备份。
- 如果您想为属于蓝绿部署的数据库实例还原手动或自动数据库快照，请确保通过检查快照拍摄时间来还原正确的数据库快照。有关更多信息，请参阅 还原到数据库实例。
- 如果您想描述以前的蓝色环境数据库实例自动备份或将其还原到某个时间点，请使用资源 ID 进行操作。

由于数据库实例的名称在切换期间会发生变化，因此您不能将其以前的名称用于 `DescribeDBInstanceAutomatedBackups` 或 `RestoreDBInstanceToPointInTime` 操作。

有关更多信息，请参阅 [将 Amazon RDS 的数据库实例还原到指定时间]。

- 当您在蓝绿部署的绿色环境中向数据库实例添加只读副本时，当您切换时，新的只读副本不会替换蓝色环境中的只读副本。但是，新的只读副本将在切换后保留在新的生产环境中。
- 切换后，Amazon Database Migration Service（Amazon DMS）复制任务无法恢复，因为蓝色环境中的检查点在绿色环境中无效。您必须使用新的检查点重新创建 DMS 任务后才能继续复制。
- 当您在蓝绿部署的绿色环境中删除数据库实例时，您无法创建新的数据库实例来替换蓝绿部署中的该实例。

  如果您创建一个与已删除的数据库实例具有相同名称和相同 Amazon 资源名称（ARN）的新数据库实例，则它具有不同的 `DbiResourceId`，因此它不属于绿色环境。

  如果您在绿色环境中删除数据库实例，则会导致以下行为：

  - 如果蓝色环境中存在同名的数据库实例，则不会将其切换到绿色环境中的数据库实例。不会通过将 `-old`*n* 添加到数据库实例名称来重命名此数据库实例。
  - 切换后，任何指向蓝色环境中数据库实例的应用程序都将继续使用相同的数据库实例。

  相同的行为适用于数据库实例和只读副本。

Amazon RDS 蓝绿部署最佳实践

以下是蓝绿部署的最佳实践。

主题

蓝绿部署的一般最佳实践

在创建蓝绿部署时，请考虑以下一般最佳实践。

- 切换之前，在绿色环境中全面测试数据库实例 。

- 使绿色环境中的数据库保持只读。建议您在绿色环境中谨慎启用写入操作，因为它们可能导致复制冲突。它们还可能导致切换后生产数据库中出现意外数据。
- 如果您使用蓝绿部署来实现架构更改，则仅进行与复制兼容的更改。

  例如，您可以在表末尾添加新列，而无需中断从蓝色部署到绿色部署的复制。但是，模式更改（例如重命名列或重命名表）会中断向绿色部署的复制。

  有关与复制兼容的更改的更多信息，请参阅 MySQL 文档中的在源和副本上使用不同的表定义进行复制以及 PostgreSQL 逻辑复制文档中的限制。

> **Note**
>
> 此限制不适用于使用物理复制的 RDS for PostgreSQL 蓝绿部署。有关更多信息，请参阅 the section called "采用物理复制的 RDS for PostgreSQL 蓝绿部署的限制"。

- 创建蓝绿部署后，必要时处理延迟加载。切换之前，请确保数据加载已完成。有关更多信息，请参阅蓝绿部署的延迟加载和存储初始化。
- 切换蓝绿部署时，请遵循切换最佳实践。有关更多信息，请参阅 the section called "切换最佳实践"。

RDS for MySQL 蓝绿部署最佳实践

从 RDS for MySQL 数据库实例创建蓝绿部署时，请考虑以下最佳实践。

- 避免使用未针对复制进行优化的非事务性存储引擎，例如 MyISAM。
- 针对二进制日志复制优化只读副本和绿色环境。如果您的数据库引擎支持，请在创建蓝绿部署之前，先启用 GTID、并行复制和崩溃安全复制，以确保数据一致性和持久性。有关更多信息，请参阅 使用基于 GTID 的复制。
- 如果绿色环境出现副本滞后，请考虑以下事项：
  - 在绿色数据库参数组中临时将 `innodb_flush_log_at_trx_commit` 参数设置为 `2`。当复制赶上进度之后，请在切换之前恢复为默认值 `1`。如果使用临时参数值时发生意外关闭或崩溃，请重建绿色环境以避免未检测到的数据损坏。
  - 为减少写入延迟和提高复制吞吐量，可临时将绿色多可用区数据库实例改为单可用区数据库实例。切换前重新启用多可用区。

RDS for PostgreSQL 蓝绿部署的最佳实践

从 RDS for PostgreSQL 数据库实例创建蓝绿部署时，请考虑以下最佳实践。

主题

RDS for PostgreSQL 蓝绿部署的一般最佳实践

从 RDS for PostgreSQL 数据库实例创建蓝绿部署时，请考虑以下一般的最佳实践。

- 创建蓝绿部署之前，请将所有 PostgreSQL 扩展更新到最新版本。有关更多信息，请参阅 the section called “升级 PostgreSQL 扩展”。
- 长时间运行的事务可能会导致严重的副本延迟。要减少副本延迟，可考虑执行下列操作：
  - 减少长期运行的事务，这些事务可能会延迟到绿色环境赶上蓝色环境之后再运行。
  - 减少蓝色环境上的批量操作，直到绿色环境赶上蓝色环境之后再运行。
  - 在创建蓝绿部署之前，对事务繁忙的表启动手动真空冻结操作。
  - 对于 PostgreSQL 12 及更高版本，请对大型表或繁忙表禁用 `index_cleanup` 参数，以提高蓝色数据库的正常维护速率。有关更多信息，请参阅 the section called “尽快对表执行 vacuum 操作”。

    > **Note**
    >
    > 在执行 vacuum 操作期间定期跳过索引清理可能会导致索引膨胀，从而降低扫描性能。最佳实践是仅在使用蓝绿部署时使用此方法。部署完成后，建议恢复定期的索引维护和清理。

- 复制缓慢会导致发送方和接收方经常重启，从而延迟同步。要确保它们保持活动状态，请在蓝色环境中将 `wal_sender_timeout` 参数设置为 `0`，在绿色环境中将 `wal_receiver_timeout` 参数设置为 `0`，从而禁用超时。
- 为防止从蓝色环境中删除预写日志（WAL）段，请将 PostgreSQL 13 及更低版本的 `wal_keep_segments` 参数设置为 15625。对于版本 14 及更高版本，如果有足够的可用存储空间，还可将 `wal_keep_size` 参数设置为 1 TiB。

采用物理复制的 RDS for PostgreSQL 蓝绿部署最佳实践

通过物理复制，Amazon RDS 可创建源数据库实例的只读副本。有关相关参数、监控、调整和故障排除，请参阅the section called “使用 RDS for PostgreSQL 只读副本”。

有关蓝绿部署何时使用物理复制而不是逻辑复制的说明，请参阅 the section called “PostgreSQL 复制方法”。

采用逻辑复制的 RDS for PostgreSQL 蓝绿部署最佳实践

在创建使用逻辑复制的蓝绿部署时，请考虑以下最佳实践。有关蓝绿部署何时使用逻辑复制而不是物理复制的说明，请参阅 the section called “PostgreSQL 复制方法”。

- 如果您的数据库有足够的可用内存，请在蓝色环境中增加 `logical_decoding_work_mem` 数据库参数的值。这样做可以减少磁盘上的解码次数，改为使用内存。有关更多信息，请参阅 PostgreSQL 文档。
  - 您可以使用 `ReplicationSlotDiskUsage` CloudWatch 指标监控写入磁盘的事务溢出量。该指标提供对复制槽的磁盘使用情况的见解，有助于确定事务数据何时超过内存容量并存储在磁盘上。您可以使用 `FreeableMemory` CloudWatch 指标监控可用内存。有关更多信息，请参阅 the section called “Amazon RDS 的 Amazon CloudWatch 实例级指标”。
  - 在 RDS for PostgreSQL 版本 14 及更高版本中，您可以使用 `pg_stat_replication_slots` 系统视图监控逻辑溢出文件的大小。
- 如果您使用的是 `aws_s3` 扩展，请在创建绿色环境之后，通过 IAM 角色赋予绿色数据库实例访问 Amazon S3 的权限。这允许导入和导出命令在切换后继续运行。有关说明，请参阅the section called “设置 Amazon S3 存储桶的访问权限”。
- 检查 UPDATE 和 DELETE 语句的性能，并评估在 WHERE 子句中使用的列上创建索引能否优化这些查询。在绿色环境中重播操作时，这可以提高性能。
- 如果您使用的是触发器，请确保它们不会影响名称以“rds”开头的 `pg_catalog.pg_publication`、`pg_catalog.pg_subscription` 和 `pg_catalog.pg_replication_slots` 对象的创建、更新及删除。
- 如果您为绿色环境指定更高的引擎版本，请对所有数据库运行 `ANALYZE` 操作来刷新 `pg_statistic` 表。在主要版本升级期间不会传输优化程序统计数据，因此您必须重新生成所有统计数据，来避免出现性能问题。有关主要版本升级期间的其他最佳实践，请参阅the section called “如何执行主要版本升级”。
- 如果在源中使用触发器来操作数据，请避免将触发器配置为 `ENABLE REPLICA` 或 `ENABLE ALWAYS`。否则，复制系统会传播更改并执行触发器，从而导致重复。

PostgreSQL 蓝绿部署的复制方法

Amazon RDS for PostgreSQL 的蓝绿部署主要使用物理复制。但是，如果您在创建蓝绿部署时请求主要版本升级，并且您的源数据库实例运行下表所列的 PostgreSQL 版本之一，Amazon RDS 将改用逻辑复制。

下表概述了 Amazon RDS 何时在 PostgreSQL 蓝绿部署中使用物理复制或逻辑复制。

源 PostgreSQL 数据库实例版本	蓝绿部署中的升级操作	复制方法
• 16.1 及所有更高的主要和次要版本 • 15.4 及更高的 15 版本 • 14.9 及更高的 14 版本 • 13.12 及更高的 13 版本 • 12.16 及更高的 12 版本 • 11.21 及更高的 11 版本	主要版本升级 （绿色实例在比蓝色实例更高的主要引擎版本上）	逻辑复制
支持的所有版本	次要版本升级，或者不升级 （绿色实例在与蓝色实例相同的主要引擎版本上）	物理复制

> **Note**
>
> 以下 RDS for PostgreSQL 版本的蓝绿部署不支持主要版本升级：版本 15.3 及更低版本、14.8 及更低版本、13.11 及更低版本、12.15 及更低版本或 11.20 及更低版本。

有关使用物理或逻辑复制的蓝绿部署的限制的信息，请参阅以下各节：

- the section called “采用物理复制的 RDS for PostgreSQL 蓝绿部署的限制”
- the section called “RDS for PostgreSQL 蓝绿部署的限制”

在 Amazon RDS 中创建蓝绿部署

创建蓝绿部署时，您需要指定要在部署中复制的源数据库实例。您选择的数据库实例是生产数据库实例，它将成为蓝色环境中的主数据库实例。此数据库实例复制到绿色环境，RDS 配置从蓝色环境中的数据库实例到绿色环境中的数据库实例的复制。

RDS 将蓝色环境的拓扑和功能复制到暂存区域。如果蓝色数据库实例具有只读副本，它们将被复制为绿色实例的副本。所有绿色副本分配得到的存储与绿色主实例一致，而其他存储参数则继承自蓝色副本。

如果蓝色数据库实例为多可用区数据库实例部署，则绿色数据库实例将创建为多可用区数据库实例部署。

主题

- 准备进行蓝绿部署
- 创建蓝绿部署时指定更改
- 蓝绿部署的延迟加载和存储初始化
- 创建蓝绿部署
- 创建蓝绿部署的设置

准备进行蓝绿部署

在创建蓝绿部署之前，必须执行某些步骤，具体取决于您的数据库实例运行的引擎。

主题

- 准备 RDS for MySQL 或 RDS for MariaDB 数据库实例以进行蓝绿部署
- 准备 RDS for PostgreSQL 数据库实例以进行采用物理复制的蓝绿部署
- 准备 RDS for PostgreSQL 数据库实例以进行采用逻辑复制的蓝绿部署

准备 RDS for MySQL 或 RDS for MariaDB 数据库实例以进行蓝绿部署

在为 RDS for MySQL 或 RDS for MariaDB 数据库实例创建蓝绿部署之前，必须启用自动备份。有关说明，请参阅the section called “启用自动备份”。

准备 RDS for PostgreSQL 数据库实例以进行采用物理复制的蓝绿部署

在创建使用物理复制的 RDS for PostgreSQL 蓝绿部署之前，务必执行以下操作：有关使用物理复制或逻辑复制的版本列表，请参阅 the section called “PostgreSQL 复制方法”。

- 在数据库实例上启用自动备份。有关说明，请参阅the section called “启用自动备份”。
- 确认数据库实例不是外部复制的源或目标。有关更多信息，请参阅 the section called “一般限制”。

准备 RDS for PostgreSQL 数据库实例以进行采用逻辑复制的蓝绿部署

在创建使用逻辑复制的 RDS for PostgreSQL 蓝绿部署之前，务必执行以下操作：有关使用逻辑复制或物理复制的版本列表，请参阅 the section called “PostgreSQL 复制方法”。

- 在开启逻辑复制（`rds.logical_replication`）的情况下，将实例与自定义数据库参数组关联。从蓝色环境复制到绿色环境需要逻辑复制。有关说明，请参阅the section called “修改数据库参数组中的参数”。

  由于蓝绿部署要求每个数据库至少有一个后台工作线程，因此请务必根据您的工作负载调整以下配置设置：有关调整每个设置的说明，请参阅 PostgreSQL 文档中的 Configuration Settings。

  - `max_replication_slots`
  - `max_wal_senders`
  - `max_logical_replication_workers`
  - `max_worker_processes`

  启用逻辑复制并设置所有配置选项后，请务必重启数据库实例，以便您的更改生效。蓝绿部署要求数据库实例与数据库参数组同步，否则创建将失败。有关更多信息，请参阅 the section called “重启数据库实例”。
- 确认数据库实例不是外部复制的源或目标。有关更多信息，请参阅 the section called “一般限制”。
- 确保数据库实例中的所有表都有主键。PostgreSQL 逻辑复制不允许对没有主键的表进行 `UPDATE` 或 `DELETE` 操作。
- RDS for PostgreSQL 使用 PostgreSQL 的本机逻辑复制，将预写日志（WAL）段存储在蓝色实例上，直到这些段在绿色环境中重播。创建蓝绿部署之前，请通过检查以下指标来验证蓝色实例是否具有足够的容量：

  - `FreeStorageSpace`
  - `TransactionLogsGeneration`
  - `TransactionLogsDiskUsage`

- `OldestReplicationSlotLag`

要估算蓝色实例所需的额外存储空间，请在工作负载高峰期监控 `TransactionLogsGeneration` CloudWatch 指标。例如，如果您的工作负载在 24 小时内生成 100 GB 的 WAL 数据，请确保至少有 100 GB 的额外存储空间来容纳一天的 WAL 段。有关更多信息，请参阅 监控数据库实例中的指标。

创建蓝绿部署时指定更改

创建蓝绿部署时，可以在绿色环境中对数据库实例进行以下更改：

部署后，您可以在绿色环境中对数据库实例进行其他修改。例如，您可以指定更高的引擎版本或不同的参数组。

有关修改数据库实例的信息，请参阅修改 Amazon RDS 数据库实例。

主题

- 指定更高的引擎版本
- 指定其它数据库参数组
- 修改存储和性能设置
- 使用 RDS 优化写入
- 升级存储配置

指定更高的引擎版本

如果要测试数据库引擎升级，可以指定更高的引擎版本。切换后，数据库将升级到您指定的主要或次要数据库引擎版本。

指定其它数据库参数组

您可以测试参数更改如何影响绿色环境中的数据库实例，或者在升级时为新的主要数据库引擎版本指定参数组。

如果您指定不同的数据库参数组，则指定的数据库参数组将与绿色环境中的所有数据库实例相关联。如果您未指定不同的参数组，则绿色环境中的每个数据库实例都与其所对应的蓝色数据库实例的参数组相关联。

修改存储和性能设置

在绿色环境中调整存储和性能设置以优化资源分配。这些设置包括分配的存储、预调配 IOPS、存储类型和存储吞吐量（对于 gp3 存储）。

您可以将绿色数据库实例的存储类型改为 gp2、gp3、io1 或 io2。对于 gp3 存储，您还可以调整存储吞吐量，以增强高需求工作负载的数据传输性能，或者减少低强度应用程序的成本。有关更多信息，请参阅 Amazon RDS 数据库实例存储。

您也可以选择增加或减少绿色环境中分配的存储。但是，只有当目标分配存储比当前存储使用量高出至少 20% 时，才应减少存储。如果减少分配的存储，Amazon RDS 会启动存储配置升级。有关更多信息，请参阅 the section called “升级存储配置”。

如果蓝色数据库实例使用磁性存储，您必须将绿色数据库实例改为通用或预调配 IOPS 存储类型，这样才能增加或减少分配的存储。

使用 RDS 优化写入

您可以使用蓝绿部署升级到支持 RDS 优化写入的数据库实例类。您只能在使用支持的数据库实例类创建的数据库上启用 RDS 优化写入。因此，此选项会创建一个使用支持的数据库实例类的绿色数据库，这使您能够在绿色数据库实例上开启 RDS 优化写入。

如果您要从不支持 RDS 优化写入的数据库实例类升级到支持 RDS 优化写入的数据库实例类，则还必须升级绿色数据库实例的存储配置。有关更多信息，请参阅 the section called “升级存储配置”。

您只能升级主要绿色数据库实例的数据库实例类。默认情况下，绿色环境中的只读副本将继承蓝色环境中的数据库实例设置。绿色环境成功创建后，您必须手动修改绿色环境中只读副本的数据库实例类。

不支持某些实例类升级，具体取决于蓝色数据库实例的引擎版本和实例类。有关数据库实例类的更多信息，请参阅 the section called “数据库实例类”。

升级存储配置

如果您的蓝色数据库未使用最新的存储配置，RDS 可以将绿色数据库实例从较旧的存储配置（32 位文件系统）迁移到首选配置。您可以使用 RDS 蓝绿部署来克服较旧 32 位文件系统的存储和文件大小的扩展限制。此外，如果指定的数据库实例类支持优化写入，则此设置会更改存储配置，使其与 RDS 优化写入兼容。

> **Note**
>
> 升级存储配置是一项 I/O 密集型操作，会导致蓝绿部署的创建时间更长。如果蓝色数据库实例使用预调配 IOPS SSD（io1 或 io2 Block Express）存储，并且您为绿色环境预调配了 4xlarge

或更大的实例，则存储升级过程会更快。涉及通用 SSD（gp2）存储的存储升级可能会耗尽您的 I/O 积分余额，从而导致升级时间更长。有关更多信息，请参阅 the section called “数据库实例存储”。

在存储升级期间，绿色数据库实例暂时不可用，而蓝色数据库实例仍可用。在此期间，复制会暂停。监控蓝色实例上的存储，并在存储空间达到 90% 时考虑扩展，因为绿色实例在升级后会自动扩展 10%。

只有当您的蓝色数据库未使用最新的存储配置，或者您要在同一请求中更改数据库实例类时，此选项才可用。您只能在最初创建蓝绿部署时升级存储配置。

蓝绿部署的延迟加载和存储初始化

当您创建蓝绿部署时，Amazon RDS 通过从数据库快照还原在绿色环境中创建主数据库实例。之后，绿色数据库实例及其只读副本继续在后台加载数据，这一过程称为延迟加载。

延迟加载仅在应用程序请求时才加载数据块。如果您尝试访问尚未加载的数据，Amazon EBS 会立即从 Amazon S3 检索它们，而其余数据继续在后台加载。有关更多信息，请参阅 Amazon EBS 快照。

为了加速整个卷的性能，Amazon RDS 提供存储初始化功能，以读取绿色环境卷中的所有块。Amazon EBS 主动从 Amazon S3 下载块，从首次使用起即可提供最大的卷性能。存储初始化完全在后台进行，这确保不会影响数据库实例的可用性或正在进行的活动，例如修补或升级。

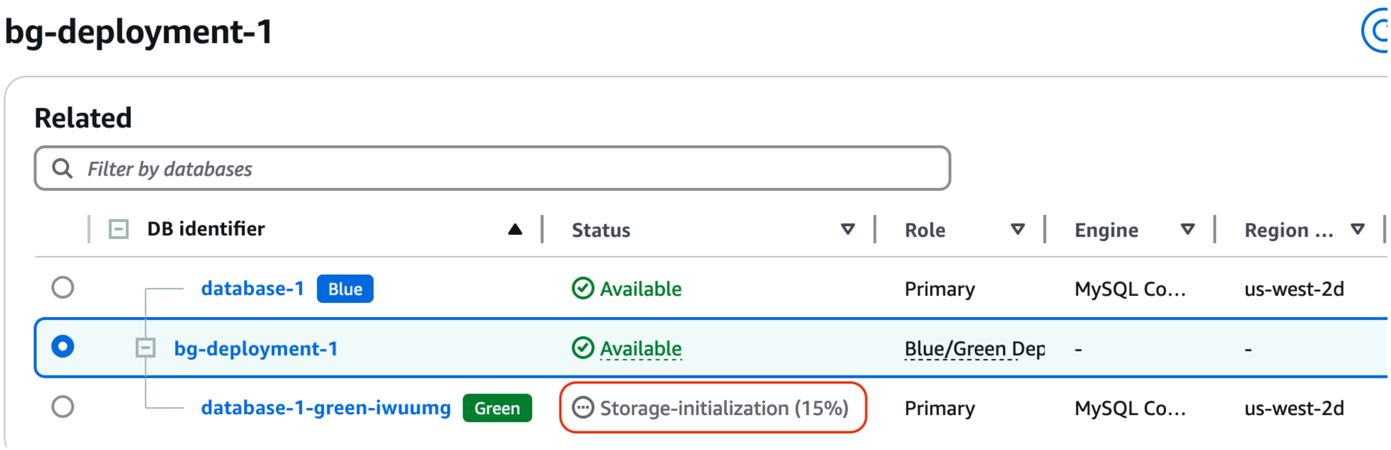
存储初始化仅适用于蓝绿部署中具有 `gp2`、`gp3`、`io1` 和 `io2` 卷类型的实例。它支持除 t3 和 t4 系列之外的所有实例类。如果您将单可用区部署中的绿色数据库实例修改为多可用区数据库实例部署，则存储初始化将包括多可用区配置中的辅助节点。

在存储初始化期间，实例仍完全可用并能用于数据库操作，但存储可能要等到初始化完成后才能达到全部性能。在存储初始化进行期间，实例的整体状态变为存储初始化，而进度指示器反映了数据库实例所有卷的最低初始化水平。

可使用控制台、Amazon CLI 或 Amazon RDS API 监控存储初始化。

Console

在 Amazon Web Services Management Console中，您可以查看存储初始化的进度以及数据库实例的状态。



Amazon CLI

通过 Amazon CLI，您可以使用 describe-db-instances 命令监控存储初始化。响应中的 `PercentProgress` 字段显示已从 Amazon S3 检索了多少百分比的数据。

```
aws rds describe-db-instances --db-instance-identifier my-db-instance

{
    "DBInstances": [
        {
            "DBInstanceIdentifier": "my-db-instance",
            "DBInstanceClass": "db.m5.2xlarge",
            "Engine": "postgres",
            "DBInstanceStatus": "storage-initialization",
            ...
            "PercentProgress": "34"
        }
    ]
}
```

Amazon RDS API

利用 Amazon RDS API，您可以通过调用 DescribeDBInstances 操作来检索存储初始化的状态。

进度指示器会随着后台初始化作业的进展而更新，这样，您就可以跟踪存储就绪情况，直到整个存储初始化完成。当您的绿色数据库实例变为完全可操作时，存储初始化可帮助优化性能。

创建蓝绿部署

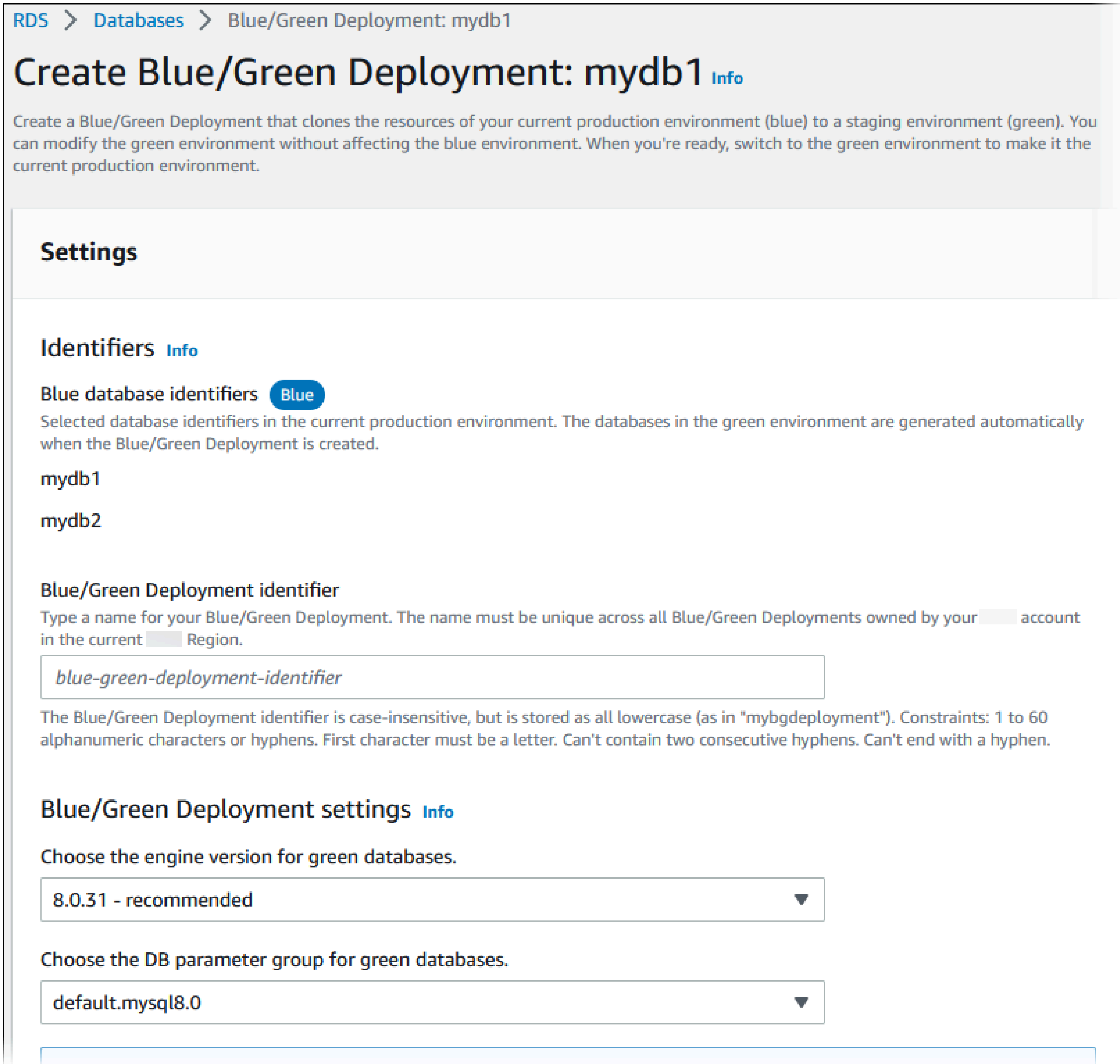
您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 创建蓝绿部署。

控制台

创建蓝绿部署

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases（数据库），然后选择要复制到绿色环境的数据库实例。
3. 依次选择操作和创建蓝绿部署。

   将出现创建蓝绿部署页面。



4. 查看蓝色数据库标识符。确保它们与您在蓝色环境中预期的数据库实例相匹配。如果不符合预期，请选择 Cancel（取消）。

5. 对于蓝绿部署名称，输入蓝绿部署的名称。

6. 在剩余部分中，指定绿色环境的设置。有关每项设置的信息，请参阅 the section called "可用设置"。

   部署后，可以在绿色环境中对数据库进行其他修改。

7. 选择创建。

Amazon CLI

要使用 Amazon CLI 创建蓝绿部署，请使用 create-blue-green-deployment 命令。有关所有可用选项的信息，请参阅the section called “可用设置”。

Example

对于 Linux、macOS 或 Unix：

```
aws rds create-blue-green-deployment \
    --blue-green-deployment-name my-blue-green-deployment \
    --source arn:aws:rds:us-east-2:123456789012:db:mydb1 \
    --target-engine-version 8.0.31 \
    --target-db-parameter-group-name mydbparametergroup
```

对于 Windows：

```
aws rds create-blue-green-deployment ^
    --blue-green-deployment-name my-blue-green-deployment ^
    --source arn:aws:rds:us-east-2:123456789012:db:mydb1 ^
    --target-engine-version 8.0.31 ^
    --target-db-parameter-group-name mydbparametergroup
```

RDS API

要使用 Amazon RDS API 创建蓝绿部署，请使用 `CreateBlueGreenDeployment` 操作。有关各选项的信息，请参阅 the section called “可用设置”。

创建蓝绿部署的设置

下表说明创建蓝绿部署时可供您选择的设置。有关 Amazon CLI 选项的更多信息，请参阅 create-blue-green-deployment。有关 RDS API 参数的更多信息，请参阅 CreateBlueGreenDeployment。

控制台设置	设置说明	CLI 选项和 RDS API 参数
分配的存储空间	要为绿色数据库实例分配的存储量（以 GiB 为单位）。您可以选择增加或减少分配的存储。 如果您的蓝色数据库实例使用磁性（`standard`）存储，则您必须将	CLI 选项： `--target-allocated-storage` API 参数：

控制台设置	设置说明	CLI 选项和 RDS API 参数
	绿色数据库实例改为通用或预调配 IOPS 存储类型，这样才能在绿色环境中修改分配的存储。 有关更多信息，请参阅 [Amazon RDS 数据库实例存储]。	`TargetAllocatedStorage`

蓝绿部署标识符	蓝绿部署的名称。	CLI 选项：
		
		`--blue-green-deployment-name`
		
		API 参数：
		
		`BlueGreenDeploymentName`

蓝色数据库标识符	要复制到绿色环境的实例的标识符。使用 CLI 或 API 时，请指定实例的 Amazon 资源名称（ARN）。	CLI 选项：
		
		`--source`
		
		API 参数：
		
		`Source`

绿色数据库的数据库参数组	与绿色环境中的数据库相关联的参数组。	CLI 选项：
		
		`--target-db-parameter-group-name`
		
		`--target-db-cluster-parameter-group-name`
		
		API 参数：
		
		`TargetDBParameterGroupName`
		
		`TargetDBClusterParameterGroupName`

控制台设置	设置说明	CLI 选项和 RDS API 参数
为绿色数据库启用优化写入	在绿色主数据库实例上启用 RDS 优化写入。有关更多信息，请参阅 the section called “使用 RDS 优化写入”。 如果您要从不支持优化写入的数据库实例类更改为支持优化写入的数据库实例类，则还必须执行存储配置升级。有关更多信息，请参阅 the section called “升级存储配置”。	对于 CLI 和 API，指定支持 RDS 优化写入的目标数据库实例类时，会自动在绿色主数据库实例上启用优化写入。

绿色数据库的引擎版本	将绿色环境中的数据库升级到指定的数据库引擎版本。
	
	如果未指定，则使用与蓝色环境中相应的数据库实例相同的引擎版本，在绿色环境中创建每个数据库。
	
	如果您选择使用逻辑复制的 RDS for PostgreSQL 数据库实例，请回顾并确认逻辑复制的限制。有关更多信息，请参阅 the section called “蓝绿部署的特定于逻辑复制的限制”。
		CLI 选项：
		
		`--target-engine-version`
		
		RDS API 参数：
		
		`TargetEngineVersion`

绿色数据库实例类	绿色环境中各个数据库实例的计算和内存容量，例如 `db.m5d.xlarge` 。
	
	只有在为绿色数据库启用 RDS 优化写入时，此选项才可见。
		CLI 选项：
		
		`--target-db-instance-class`
		
		RDS API 参数：
		
		`TargetDBInstanceClass`

控制台设置	设置说明	CLI 选项和 RDS API 参数

预置 IOPS	最初为绿色数据库分配的预调配每秒输入/输出操作次数（IOPS）。
	
	此值仅适用于绿色主数据库实例，不适用于绿色副本。
		CLI 选项：
		
		`--target-iops`
		
		RDS API 参数：
		
		`TargetIops`

存储配置升级	选择是否升级您的存储文件系统配置。如果您启用此设置，RDS 会将绿色数据库从旧存储文件系统迁移到首选配置。
	
	只有当您的蓝色数据库未使用最新的存储配置，或者您在同一请求中启用 RDS 优化写入时，此选项才可用。您只能在最初创建蓝绿部署时升级存储配置。
	
	有关更多信息，请参阅 the section called “升级存储文件系统”。
		CLI 选项：
		
		`--upgrade-target-storage-config`
		
		RDS API 参数：
		
		`UpgradeTargetStorageConfig`

Storage throughput（存储吞吐量）	绿色数据库的存储吞吐量值。仅在您为存储类型选择通用型 SSD（gp3）时，此设置才可见。
	
	此值仅适用于绿色主数据库实例，不适用于绿色副本。
	
	有关更多信息，请参阅 gp3 存储（推荐）。
		CLI 选项：
		
		`--target-storage-throughput`
		
		RDS API 参数：
		
		`TargetStorageThroughput`

控制台设置	设置说明	CLI 选项和 RDS API 参数

存储类型	绿色数据库的存储类型。以下为支持的存储类型：
	• 通用 SSD（gp2）
	• 通用 SSD（gp3）
	• 预调配 IOPS（io1）
	• 预调配 IOPS SSD（io2）
	
	此值仅适用于绿色主数据库实例，不适用于绿色副本。
	
	有关更多信息，请参阅 Amazon RDS 存储类型。
		CLI 选项：
		
		`--target-storage-type`
		
		RDS API 参数：
		
		`TargetStorageType`

查看 Amazon RDS 中的蓝绿部署

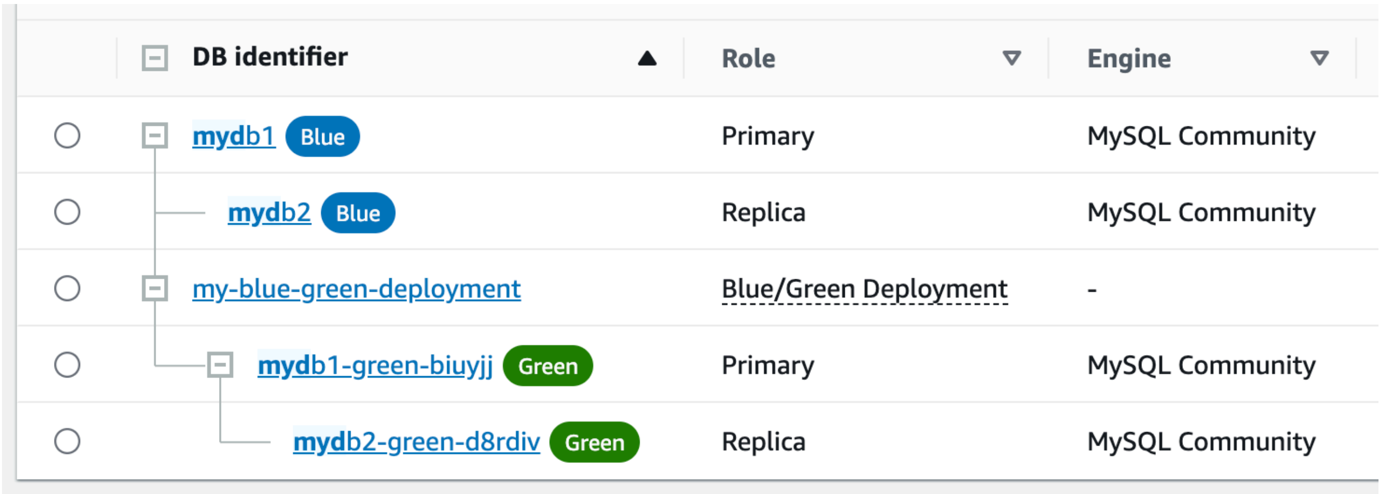
您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 查看有关蓝绿部署的详细信息。

您还可以查看和订阅事件，以了解有关蓝绿部署的信息。有关更多信息，请参阅 蓝绿部署事件。

控制台

查看有关蓝绿部署的详细信息

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases（数据库），然后在列表中找到蓝绿部署。

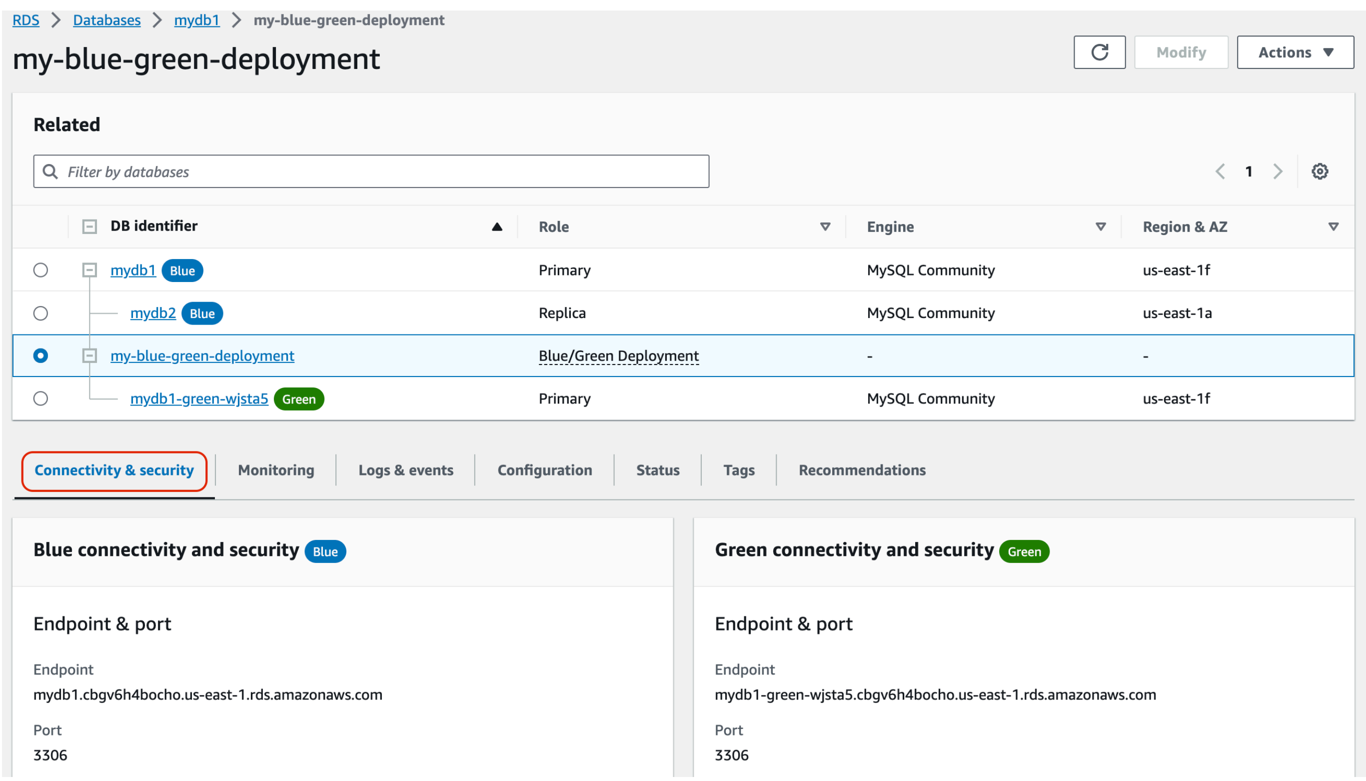


蓝绿部署的 Role（角色）值为 Blue/Green Deployment（蓝绿部署）。

3. 选择要查看的蓝绿部署的名称以显示其详细信息。

   每个选项卡都有一个用于蓝色部署的部分和一个用于绿色部署的部分。例如，在配置选项卡上，如果在绿色环境中升级数据库引擎版本，则蓝色环境和绿色环境中的数据库引擎版本可能不同。

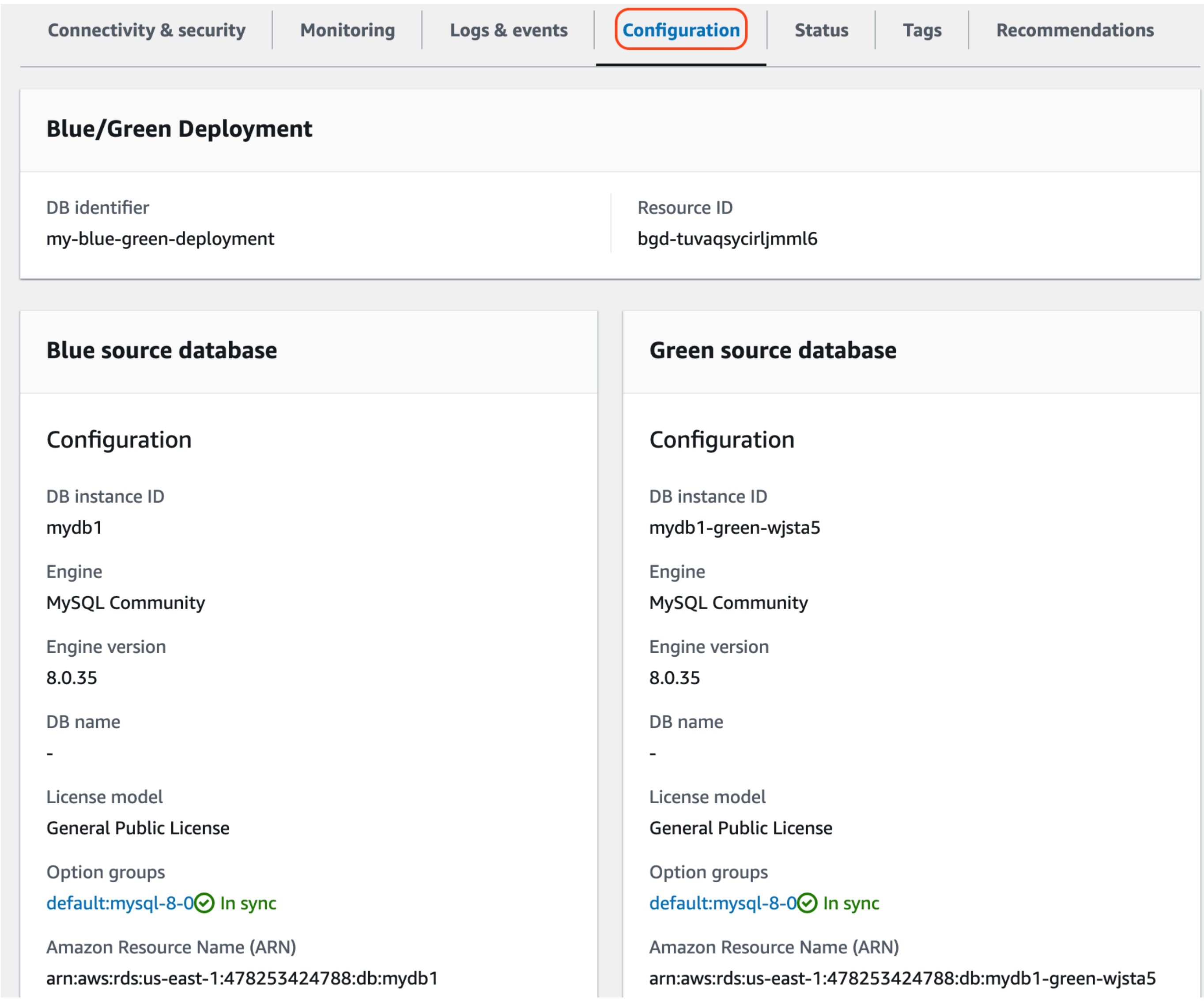
   下图显示连接和安全选项卡的示例：



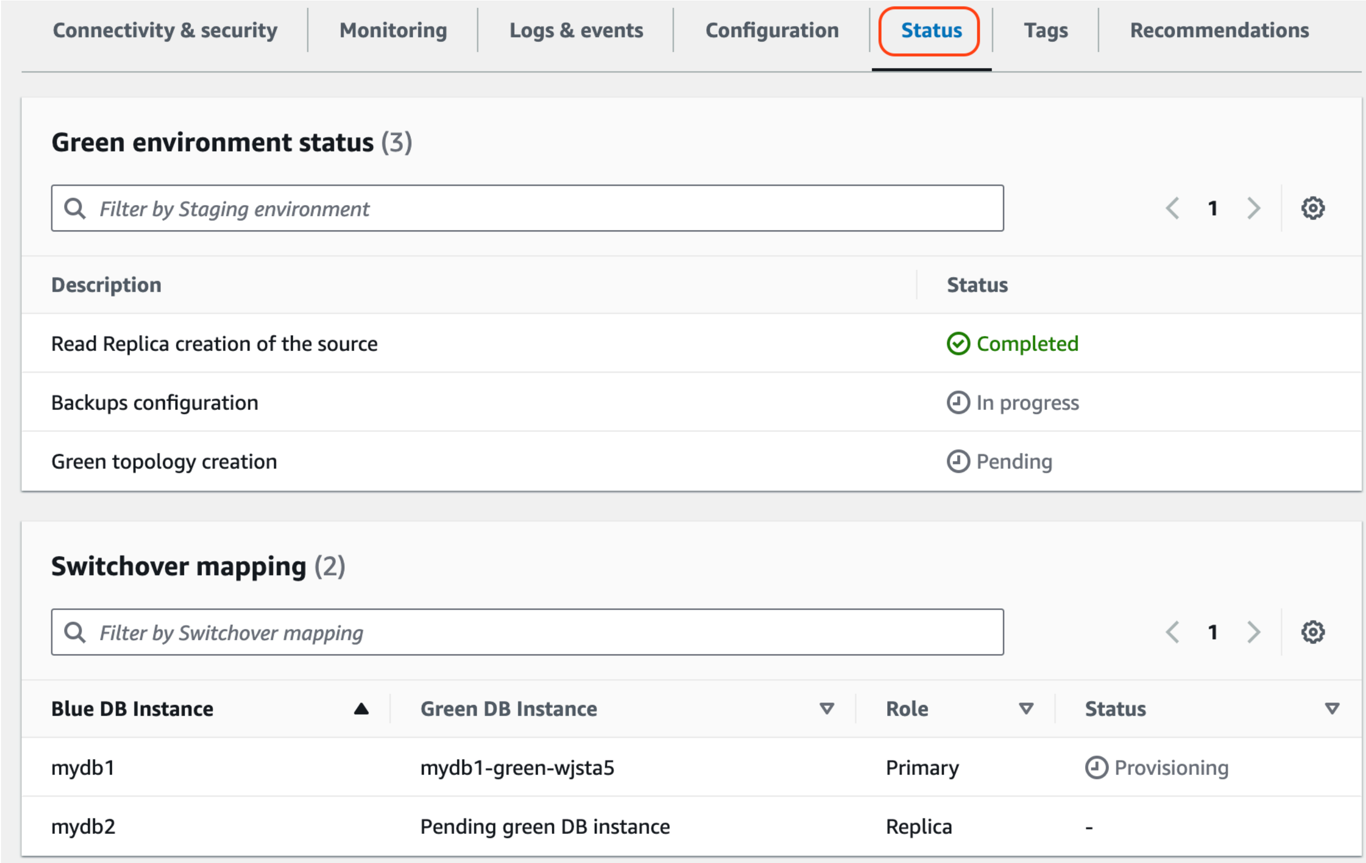
连接和安全选项卡还包括一个名为复制的部分，其中显示了复制的当前状态以及蓝色和绿色环境之间的副本滞后。如果复制状态为 `Replicating`，则表示蓝绿部署复制成功。

对于使用逻辑复制的 RDS for PostgreSQL 蓝绿部署，如果您在蓝色环境中进行了不受支持的 DDL 或大型对象更改，则复制状态会变成 `Replication degraded`。有关更多信息，请参阅 [the section called "蓝绿部署的特定于逻辑复制的限制"]。

下图显示配置选项卡的示例：



下图显示状态选项卡的示例：



Amazon CLI

要使用 Amazon CLI 查看有关蓝绿部署的详细信息，请使用 [describe-blue-green-deployments](https://docs.aws.amazon.com/cli/latest/reference/rds/describe-blue-green-deployments.html) 命令。

Example 通过筛选蓝绿部署的名称来查看有关蓝绿部署的详细信息

当您使用 [describe-blue-green-deployments](https://docs.aws.amazon.com/cli/latest/reference/rds/describe-blue-green-deployments.html) 命令时，可以按 `--blue-green-deployment-name` 进行筛选。

以下示例显示名为 *`my-blue-green-deployment`* 的蓝绿部署的详细信息。

```
aws rds describe-blue-green-deployments \
  --filters Name=blue-green-deployment-name,Values=my-blue-green-deployment
```

Example 通过指定蓝绿部署的标识符查看有关蓝绿部署的详细信息

使用 [describe-blue-green-deployments](https://docs.aws.amazon.com/cli/latest/reference/rds/describe-blue-green-deployments.html) 命令时，您可以指定 `--blue-green-deployment-identifier` 选项。

以下示例显示带有标识符 *bgd-1234567890abcdef* 的蓝绿部署的详细信息。

```
aws rds describe-blue-green-deployments \
  --blue-green-deployment-identifier bgd-1234567890abcdef
```

RDS API

要使用 Amazon RDS API 查看有关蓝绿部署的详细信息，请使用 [DescribeBlueGreenDeployments](#) 操作并指定 `BlueGreenDeploymentIdentifier`。

切换 Amazon RDS 中的蓝绿部署

切换将绿色环境转换为新的生产环境。当绿色数据库实例具有只读副本时，它们也会被转换。在切换之前，生产流量将路由到蓝色环境中的数据库实例和只读副本。切换后，生产流量将路由到绿色环境中的数据库实例和只读副本。

切换 蓝绿部署不同于在蓝绿部署中提升 绿色数据库实例。如果您通过从操作菜单中选择提升来手动提升绿色数据库实例，则蓝色环境与绿色环境之间的复制会中断，蓝绿部署将进入无效配置状态。

主题

切换超时

您可以指定 30 秒与 3600 秒（一小时）之间的切换超时时间。如果切换所花的时间超过指定的持续时间，则所有更改都将回滚，并且不会对任一环境进行任何更改。默认的超时期间为 300 秒（5 分钟）。

切换防护机制

当您开始切换时，Amazon RDS 会运行一些基本检查，以测试蓝绿环境是否准备好进行切换。这些检查称为切换防护机制。如果环境还没有准备好进行切换，这些切换防护机制可以防止发生切换。因此，它们可以避免比预期更长的停机时间，并防止切换开始时可能导致的蓝色和绿色环境之间的数据丢失。

Amazon RDS 在绿色环境中运行以下防护机制检查：

- 复制运行状况 - 检查绿色主数据库实例复制状态是否为正常运行。绿色主数据库实例是蓝色主数据库实例的副本。
- 复制滞后 - 检查绿色主数据库实例的副本滞后是否在切换的允许范围内。允许的限制基于指定的超时期间。副本滞后表示绿色主数据库实例滞后于其蓝色主数据库实例的程度。有关更多信息，请参阅 the section called "在切换之前监控副本滞后"。
- 主动写入 - 确保绿色主数据库实例上没有主动写入。

Amazon RDS 在蓝色环境中运行以下防护机制检查：

- 外部复制 – 对于 RDS for PostgreSQL，请确保蓝色环境不是自行管理的逻辑源（发布者）或副本（订阅用户）。如果是，建议您删除蓝色环境中所有数据库的自行管理的复制插槽和订阅，继续切换，然后重新创建它们以恢复复制。对于 RDS for MySQL 和 RDS for MariaDB，请确保蓝色数据库不是外部二进制日志副本。如果是，请确保它未在主动复制。
- 长时间运行的活动写入 - 确保蓝色主数据库实例上没有长时间运行的活动写入，因为它们会增加副本滞后。
- 长时间运行的 DDL 语句 - 确保蓝色主数据库实例上没有长时间运行的 DDL 语句，因为它们会增加副本滞后。
- 不支持的 PostgreSQL 更改 – 对于使用逻辑复制的 RDS for PostgreSQL 蓝绿部署，确保在蓝色环境中没有进行 DDL 更改，也没有添加或修改大型对象。有关更多信息，请参阅 the section called "蓝绿部署的特定于逻辑复制的限制"。

  如果 Amazon RDS 检测到不支持的 PostgreSQL 更改，它会将复制状态更改为 `Replication degraded`，并通知您蓝绿部署无法进行切换。要继续切换，建议您删除并重新创建蓝绿部署和所有绿色数据库。为此，请选择操作、使用绿色数据库删除。

切换操作

当您切换蓝绿部署时，RDS 会执行以下操作：

1. 运行防护机制检查，以验证蓝色和绿色环境是否已准备好进行切换。
2. 在两个环境中停止对主数据库实例进行新的写入操作。
3. 在这两个环境中删除与数据库实例的连接，并且不允许新的连接。
4. 等待复制在绿色环境中赶上进度，以便绿色环境与蓝色环境同步。
5. 重命名这两个环境中的数据库实例。

   RDS 重命名绿色环境中的数据库实例，以匹配蓝色环境中相应的数据库实例。例如，假设蓝色环境中数据库实例的名称为 `mydb`。还假设绿色环境中相应的数据库实例的名称为 `mydb-green-abc123`。在切换期间，绿色环境中数据库实例的名称更改为 `mydb`。

   RDS 通过在当前名称后面附加 `-old`*n*（其中 *n* 是一个数字）来重命名蓝色环境中的数据库实例。例如，假设蓝色环境中数据库实例的名称为 `mydb`。切换后，数据库实例名称可能为 `mydb-old1`。

   RDS 还重命名绿色环境中的端点，以匹配蓝色环境中的相应端点，以便无需更改应用程序。
6. 允许在这两种环境中连接到数据库。
7. 允许在新的生产环境中对主数据库实例进行写入操作。

   切换之后，先前的生产主数据库实例仅允许读取操作，直到您将 `read_only` 参数（对于 RDS for MySQL）或 `default_transaction_read_only` 参数（对于 RDS for PostgreSQL）设置为 `0` 并重新启动数据库实例。

您可以使用 Amazon EventBridge 监控切换的状态。有关更多信息，请参阅 the section called “蓝绿部署事件”。

如果在蓝色环境中配置了标签，则这些标签将在切换期间复制到新的生产环境。有关标签的更多信息，请参阅 为 Amazon RDS 资源添加标签。

如果切换开始后由于任何原因在完成之前停止，则所有更改都将回滚，并且不会对任一环境进行任何更改。

切换最佳实践

在切换之前，我们强烈建议您通过完成以下任务来遵循最佳实践：

- 在绿色环境中彻底测试资源。确保它们正常高效地运行。
- 监控相关的 Amazon CloudWatch 指标。有关更多信息，请参阅 the section called “在切换之前验证 CloudWatch 指标”。
- 确定最佳切换时间。

在切换期间，两个环境中的数据库都会切断写入操作。确定生产环境中流量最低的时间。长时间运行的事务（例如活动的 DDL）会延长您的切换时间，从而延长生产工作负载的停机时间。

如果您的数据库实例上有大量连接，请考虑在切换蓝绿部署之前，手动将连接减少到应用程序所需的最低数量。实现此目标的一种方法是创建一个脚本，该脚本监控蓝绿部署的状态，并在检测到状态已更改为 `SWITCHOVER_IN_PROGRESS` 时开始清理连接。

- 确保两个环境中的数据库实例均处于 `Available` 状态。
- 确保绿色环境中的主数据库实例运行状况良好且正在复制。
- 确保您的网络和客户端配置不会将 DNS 缓存存活时间（TTL）增加到五秒以上，这是 RDS DNS 区域的默认值。
  否则，应用程序将在切换后继续向蓝色环境发送写入流量。
- 切换之前，请确保数据加载已完成。有关更多信息，请参阅 the section called “延迟加载和存储初始化”。
- 对于使用逻辑复制的 RDS for PostgreSQL 蓝绿部署，执行以下操作：
  - 在切换之前，请查看逻辑复制限制并采取任何必需的操作。有关更多信息，请参阅 the section called “蓝绿部署的特定于逻辑复制的限制”。
  - 运行 `ANALYZE` 操作以刷新 `pg_statistics` 表。这样可以降低切换后出现性能问题的风险。

> **Note**
> 在切换期间，您无法修改切换中包含的任何数据库实例。

在切换之前验证 CloudWatch 指标

在切换蓝绿部署之前，建议您检查 Amazon CloudWatch 中以下指标的值。

- `DatabaseConnections` – 使用此指标估算蓝绿部署上的活动水平，并在切换之前确保该值处于部署的可接受水平。如果开启了性能详情，则 `DBLoad` 是更准确的指标。

有关更多信息，请参阅 the section called “RDS 的 CloudWatch 指标”。

在切换之前监控副本滞后

在您切换蓝绿部署之前，请确保副本滞后接近于零，以减少停机时间。

RDS for MySQL 和 RDS for MariaDB

对于 MySQL 和 MariaDB 蓝绿部署，请在绿色环境中检查 `ReplicaLag` CloudWatch 指标，以确定当前的副本滞后。有关更多信息，请参阅 the section called “诊断并解决只读副本之间的滞后”。

RDS for PostgreSQL

对于使用物理复制的 PostgreSQL 蓝绿部署，请参阅the section called “监控和调整复制过程”中关于确定当前副本滞后的说明。

对于使用逻辑复制的 PostgreSQL 蓝绿部署，请在蓝色环境中检查 `OldestReplicationSlotLag` CloudWatch 指标，以确定当前的副本滞后。有关更多信息，请参阅 the section called “Amazon RDS 的 Amazon CloudWatch 实例级指标”。

此外，您还可以在蓝色环境中运行以下 SQL 查询：

```
SELECT slot_name,
       confirmed_flush_lsn as flushed,
       pg_current_wal_lsn(),
       (pg_current_wal_lsn() - confirmed_flush_lsn) AS lsn_distance
FROM pg_catalog.pg_replication_slots
WHERE slot_type = 'logical';

slot_name        |    flushed    | pg_current_wal_lsn | lsn_distance
-----------------+---------------+--------------------+------------
logical_replica1 | 47D97/CF32980 | 47D97/CF3BAC8      | 37192
```

`confirmed_flush_lsn` 表示发送到副本的最后一个日志序列号（LSN）。`pg_current_wal_lsn` 表示数据库现在的位置。如果 `lsn_distance` 为 0，则表示已捕获副本。

有关蓝绿部署何时使用物理复制或逻辑复制的说明，请参阅 the section called “PostgreSQL 复制方法”。

切换蓝绿部署

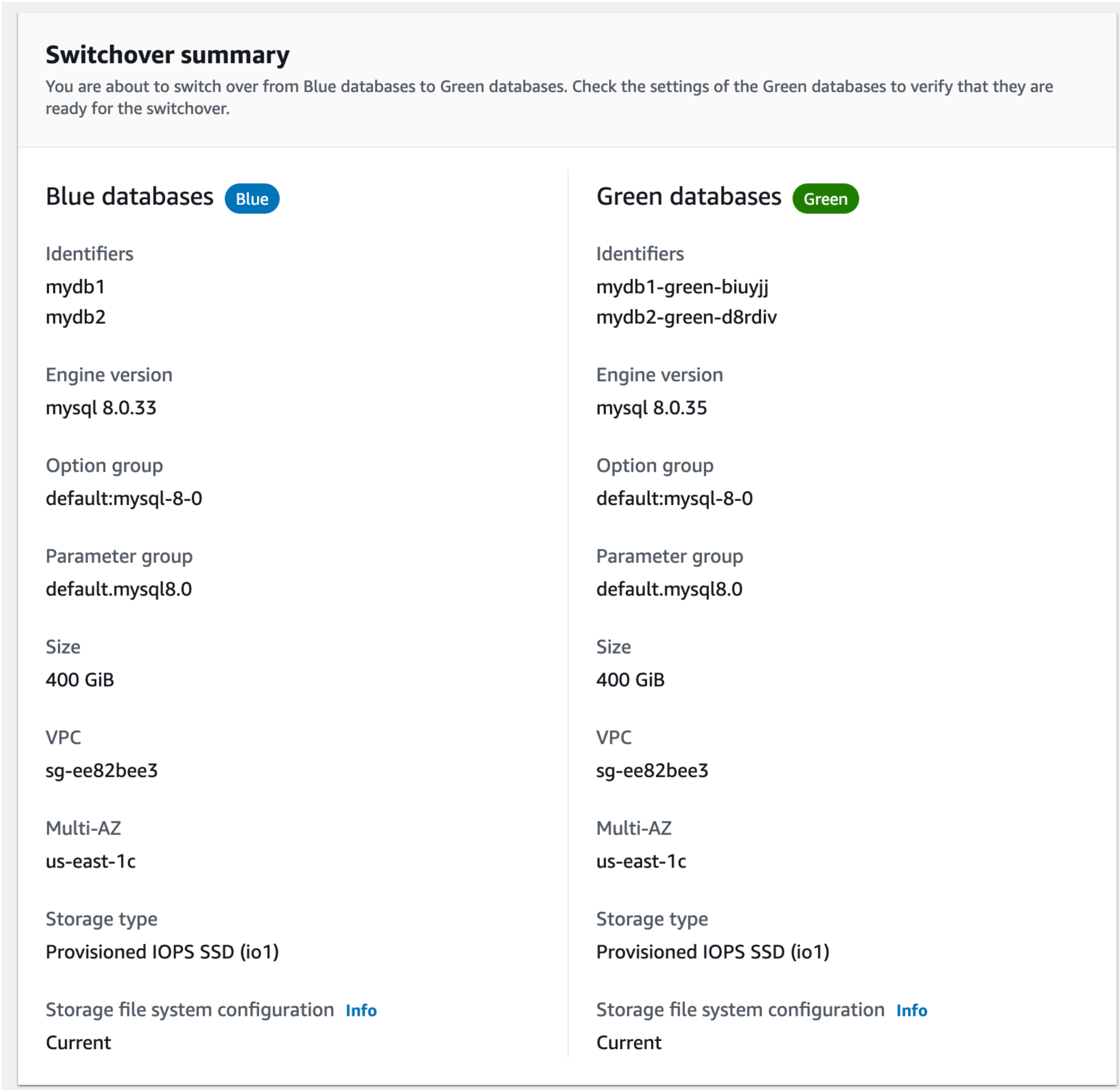
您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 切换蓝绿部署。

控制台

切换蓝绿部署

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases（数据库），然后选择要切换的蓝绿部署。
3. 对于 Actions（操作），选择 Switch over（切换）。

   将出现 Switch over（切换）页面。



4. 在 Switch over（切换）页面上，查看切换摘要。确保两个环境中的资源都符合您的期望。如果不符合预期，请选择 Cancel（取消）。
5. 对于超时设置，输入切换的时间限制。
6. 如果您的实例运行 RDS for PostgreSQL，请查看并确认切换前建议。有关更多信息，请参阅 the section called “蓝绿部署的特定于逻辑复制的限制”。
7. 选择 Switch over（切换）。

Amazon CLI

要使用 Amazon CLI 切换蓝绿部署，请使用带有以下选项的 switchover-blue-green-deployment 命令：

- `--blue-green-deployment-identifier` – 指定蓝绿部署的资源 ID。
- `--switchover-timeout` – 指定切换的时间限制，以秒为单位。默认值为 300。

Example 切换蓝绿部署

对于 Linux、macOS 或 Unix：

```
aws rds switchover-blue-green-deployment \
    --blue-green-deployment-identifier bgd-1234567890abcdef \
    --switchover-timeout 600
```

对于 Windows：

```
aws rds switchover-blue-green-deployment ^
    --blue-green-deployment-identifier bgd-1234567890abcdef ^
    --switchover-timeout 600
```

RDS API

要使用 Amazon RDS API 切换蓝绿部署，请使用带有以下参数的 `SwitchoverBlueGreenDeployment` 操作：

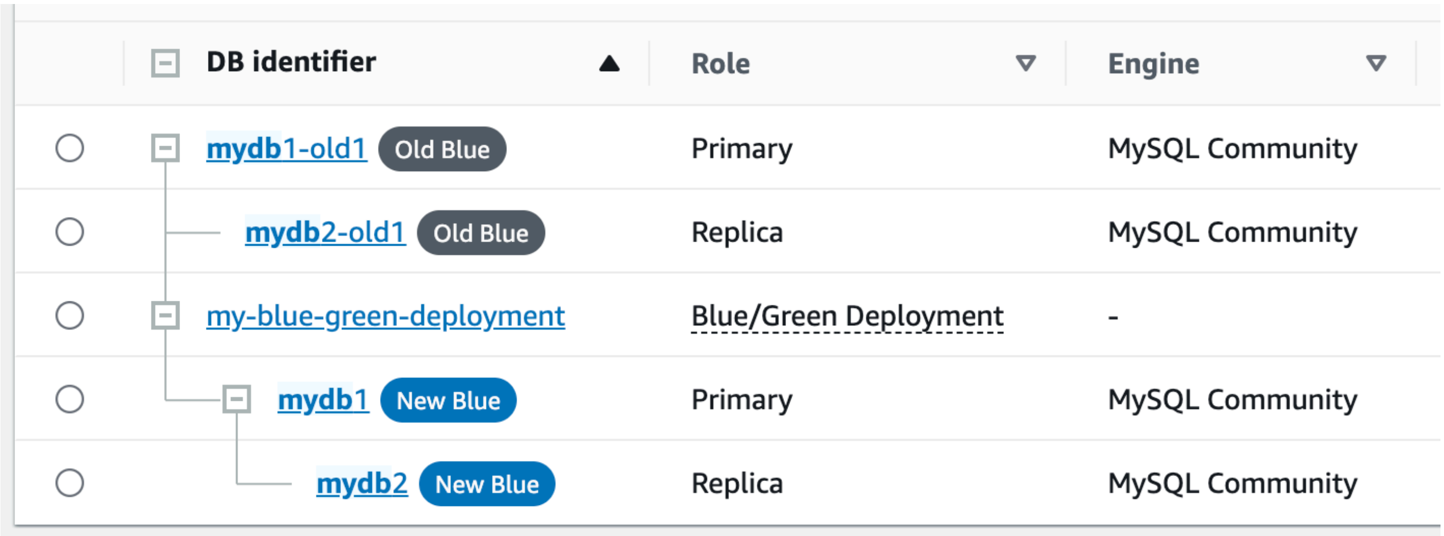
- `BlueGreenDeploymentIdentifier` – 指定蓝绿部署的资源 ID。
- `SwitchoverTimeout` – 指定切换的时间限制，以秒为单位。默认值为 300。

切换后

切换后，将保留先前蓝色环境中的数据库实例。标准成本适用于这些资源。蓝色和绿色环境之间的复制会停止。

RDS 通过在当前资源名称后面附加 `-old`*n*（其中 *n* 是一个数字）来重命名蓝色环境中的数据库实例。旧的蓝色环境中的数据库实例始终为只读的，直到您将 `read_only` 参数（对于 RDS for MySQL）或 `default_transaction_read_only` 参数（对于 RDS for PostgreSQL）设置为 `0`。RDS 会命名绿色环境 `-new`*n* 中的数据库实例。

如果您删除蓝绿部署资源，RDS 会保留 `-old`*n* 和 `-new`*n* 资源。



更新使用者的父节点

RDS 提供完全托管式只读副本。但是，它还提供用于设置自托管式副本（也称为外部副本）的选项。外部副本允许您使用第三方资源作为复制目标。

切换 RDS for MariaDB 或 RDS for MySQL 蓝绿部署后，如果蓝色数据库实例在切换之前有任何外部副本或二进制日志使用者，则必须在切换后更新其父节点以保持复制连续性。

更新父节点

1. 切换后，之前处于绿色环境中的数据库实例会发出一个事件，其中包含主日志文件名和主日志位置。要找到该事件，请导航到 RDS 控制台，然后从左侧导航窗格中选择事件。
2. 在切换之前，按源为旧绿色数据库实例名称的事件进行筛选。
3. 找到包含二进制日志坐标的事件。事件消息类似于：`Binary log coordinates in green environment after switchover: file mysql-bin-changelog.`*`000003`*` and position `*`40134574`*。
4. 确保使用者或副本已应用来自旧蓝色环境的所有二进制日志。然后，使用提供的二进制日志坐标在使用者上恢复复制。例如，如果您在 EC2 上运行 MySQL 副本，则可以使用下列命令：

   MySQL 8.0.22 及更低的主要和次要版本

   ```
   CHANGE MASTER TO MASTER_HOST='{new-writer-endpoint}', MASTER_LOG_FILE='mysql-bin-changelog.000003', MASTER_LOG_POS=40134574;
   ```

   MySQL 8.0.23 及更高的主要和次要版本

```
CHANGE REPLICATION SOURCE TO SOURCE_HOST='{new-writer-endpoint}',
 SOURCE_LOG_FILE='mysql-bin-changelog.000003', SOURCE_LOG_POS=40134574;
```

如果使用者是另一个 RDS for MySQL 或 RDS for MariaDB 数据库实例，则按顺序运行以下存储过程：

1. the section called “mysql.rds_stop_replication”
2. mysql.rds_reset_external_master（对于版本 8.0 及更低版本）或 mysql_rds_reset_external_source（对于版本 8.4 及更高版本）
3. mysql.rds_set_external_master（对于版本 8.0 及更低版本）或 mysql_rds_set_external_source（对于版本 8.4 及更高版本）
4. the section called “mysql.rds_start_replication”

删除 Amazon RDS 中的蓝绿部署

您可以在切换蓝绿部署之前或之后将其删除。

当您在切换蓝绿部署之前将其删除时，Amazon RDS 会在绿色环境中可选删除数据库实例：

- 如果您选择在绿色环境中删除数据库实例（`--delete-target`），则这些实例必须已关闭删除保护功能。
- 如果您未在绿色环境中删除数据库实例（`--no-delete-target`），则实例会被保留，但它们不再是蓝绿部署的一部分。对于 RDS for MySQL，将继续在环境之间进行复制。对于 RDS for PostgreSQL，绿色环境将提升为独立环境，因此复制停止。

切换后，控制台中不提供用于删除绿色数据库的选项。使用 Amazon CLI 删除蓝绿部署时，如果部署状态为 `SWITCHOVER_COMPLETED`，则无法指定 `--delete-target` 选项。

> **Important**
>
> 删除蓝绿部署不会影响蓝色环境。

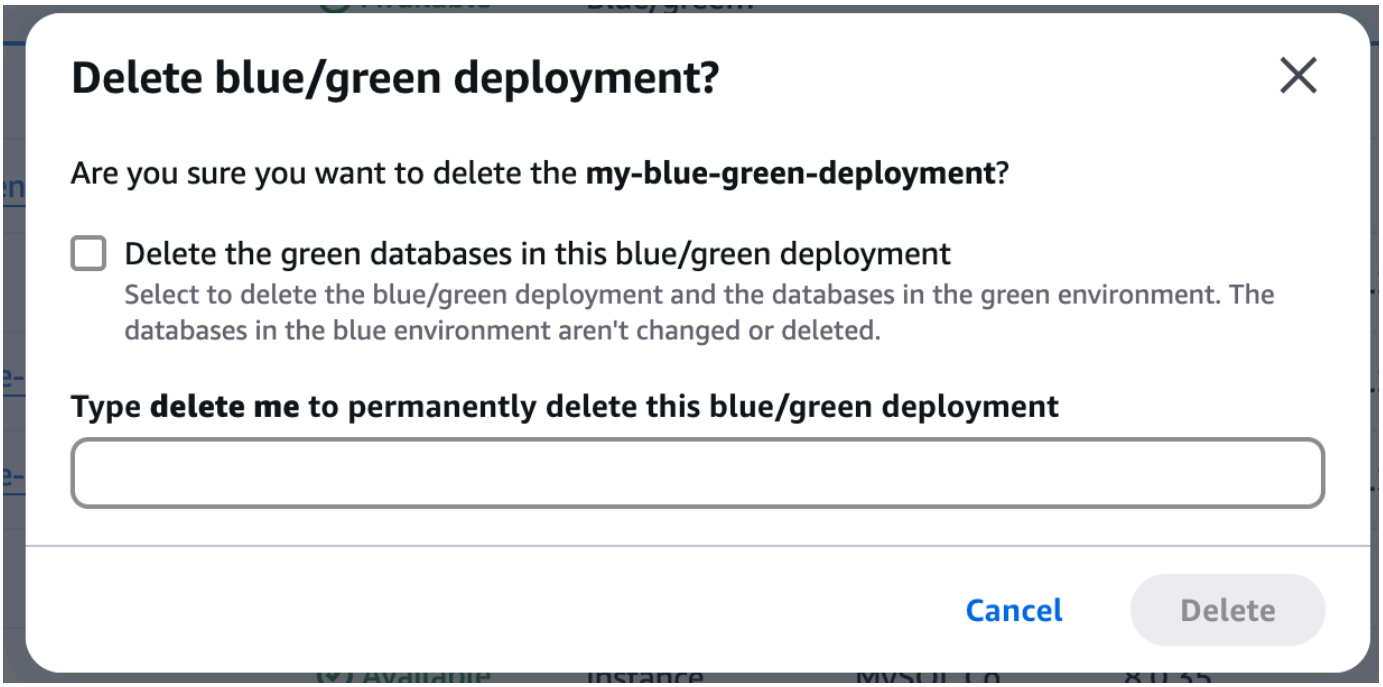
您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 删除蓝绿部署。

控制台

删除蓝绿部署

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases（数据库），然后选择要删除的蓝绿部署。
3. 对于操作，选择删除。

   将显示是否删除蓝绿部署？窗口。



   要删除绿色数据库，请选择删除此蓝绿部署中的绿色数据库。

4. 在框中输入 **`delete me`**。
5. 选择删除。

Amazon CLI

要使用 Amazon CLI 删除蓝绿部署，请使用带有以下选项的 delete-blue-green-deployment 命令：

- `--blue-green-deployment-identifier` – 要删除的蓝绿部署的资源 ID。
- `--delete-target` – 指定删除绿色环境中的数据库实例。如果蓝绿部署的状态为 `SWITCHOVER_COMPLETED`，则无法指定此选项。

- `--no-delete-target` – 指定保留绿色环境中的数据库实例。

Example 删除绿色环境中的蓝绿部署和数据库实例

对于 Linux、macOS 或 Unix：

```
aws rds delete-blue-green-deployment \
    --blue-green-deployment-identifier bgd-1234567890abcdef \
    --delete-target
```

对于 Windows：

```
aws rds delete-blue-green-deployment ^
    --blue-green-deployment-identifier bgd-1234567890abcdef ^
    --delete-target
```

Example 删除蓝绿部署，但保留绿色环境中的数据库实例

对于 Linux、macOS 或 Unix：

```
aws rds delete-blue-green-deployment \
    --blue-green-deployment-identifier bgd-1234567890abcdef \
    --no-delete-target
```

对于 Windows：

```
aws rds delete-blue-green-deployment ^
    --blue-green-deployment-identifier bgd-1234567890abcdef ^
    --no-delete-target
```

RDS API

要使用 Amazon RDS API 删除蓝绿部署，请使用带有以下参数的 DeleteBlueGreenDeployment 操作：

- `BlueGreenDeploymentIdentifier` – 要删除的蓝绿部署的资源 ID。
- `DeleteTarget` – 指定 `TRUE` 以删除绿色环境中的数据库实例，或指定 `FALSE` 以保留它们。如果蓝绿部署的状态为 `SWITCHOVER_COMPLETED`，则不能为 `TRUE`。

备份、还原和导出数据

本节介绍如何从 Amazon RDS 数据库实例或多可用区数据库集群备份、还原和导出数据。

有关备份 Amazon RDS 数据库实例或多可用区数据库集群的信息，请参阅以下主题。

- 备份简介
- 管理自动备份
- 管理手动备份

有关还原 Amazon RDS 数据库实例或多可用区数据库集群的信息，请参阅还原到数据库实例。

有关复制、共享或导出数据库快照的信息，请参阅以下主题。

- 复制 Amazon RDS 的数据库快照
- 共享 Amazon RDS 的数据库快照
- 将 Amazon RDS 的数据库快照数据导出到 Amazon S3

有关查看 Amazon Backup 管理的自动备份的信息，请参阅使用 Amazon Backup 管理 Amazon RDS 的自动备份。

备份简介

在数据库实例的备份时段中，Amazon RDS 创建并保存数据库实例或多可用区数据库集群的自动备份。RDS 创建数据库实例的存储卷快照，并备份整个数据库实例而不仅仅是单个数据库。RDS 根据您指定的备份保留期保存数据库实例的自动备份。如果需要，您可以将数据库实例恢复到备份保留期中的任意时间点。

自动备份遵循以下规则：

- 数据库实例必须处于 `available` 状态才能进行自动备份。当数据库实例处于 `available` 之外的状态（例如 `storage_full`）时，不会进行自动备份。
- 数据库快照副本在相同数据库的相同 Amazon Web Services 区域中运行时，不会进行自动备份。

还可以通过创建数据库快照来手动备份数据库实例。有关手动创建数据库快照的更多信息，请参阅为 Amazon RDS 的单可用区数据库实例创建数据库快照。

数据库实例的第一个快照包含整个数据库的数据。相同数据库的后续快照为增量快照，这意味着仅保存在最新快照后更改的数据。

您可以复制自动和手动数据库快照，还可以共享手动数据库快照。有关复制数据库快照的更多信息，请参阅复制 Amazon RDS 的数据库快照。有关共享数据库快照的更多信息，请参阅共享 Amazon RDS 的数据库快照。

备份存储

您的每个 Amazon Web Services 区域的 Amazon RDS 备份存储由该区域的自动备份和手动数据库快照组成。备份存储总空间相当于该区域中所有备份的存储空间之和。将数据库快照移至另一个区域将增加目标区域的备份存储。备份存储在 Amazon S3 中。

如果您在删除数据库实例时选择保留自动备份，则自动备份将保存完整的保留周期。如果您在删除数据库实例时未选择 Retain automated backups (保留自动备份)，则所有自动备份将随同数据库实例删除。自动备份删除后无法恢复。如果您选择让 Amazon RDS 在删除数据库实例之前创建最终数据库快照，则可以使用该快照来恢复数据库实例。（可选）您可以使用之前创建的手动快照。手动快照不会删除。您在每个区域最多可拥有 100 个手动快照。

管理自动备份

本节介绍如何管理数据库实例和多可用区数据库集群的自动备份。

主题

备份时段

自动备份在每天的首选备份时段中进行。如果备份所需的时间超过了分配给备份时段的时间，则备份将在该时段结束后继续，直至完成。备份时段不能与数据库实例或多可用区数据库集群的每周维护时段重叠。

在自动备份时段期间，启动备份进程时可能会短时间暂停存储 I/O (通常不到几秒)。在备份多可用区部署时，可能需要等待几分钟。对于 MariaDB、MySQL、Oracle 和 PostgreSQL，多可用区部署的备份期间不会暂停主数据库上的 I/O 活动，因为备份获取自备用数据库。对于 SQL Server，在单可用区和多可用区部署的备份期间，I/O 活动会短暂挂起，因为备份是从主实例进行的。对于 Db2，在备份期间，I/O 活动会短暂挂起，即使备份是从备用数据库进行的。

启动备份时，如果数据库实例或集群具有繁重的工作负载，则有时会跳过自动备份。如果跳过备份，您仍然可以执行时间点恢复（PITR），并仍在下一个备份时段尝试备份。有关 PITR 的更多信息，请参阅将 Amazon RDS 的数据库实例还原到指定时间。

如果创建数据库实例或多可用区数据库集群时未指定首选备份时段，Amazon RDS 将分配 30 分钟的原定设置备份时段。此时段是从每个 Amazon Web Services 区域的 8 小时时间段中随机选择出来的。下表列出了已分配默认备份时段的各个 Amazon Web Services 区域的时间块。

区域名称	区域	时间段
美国东部（弗吉尼亚州北部）	us-east-1	03:00–11:00 UTC
美国东部（俄亥俄州）	us-east-2	03:00–11:00 UTC
美国西部（加利福尼亚北部）	us-west-1	06:00–14:00 UTC
美国西部（俄勒冈州）	us-west-2	06:00–14:00 UTC
非洲（开普敦）	af-south-1	03:00–11:00 UTC
亚太地区（香港）	ap-east-1	06:00–14:00 UTC
亚太地区（海得拉巴）	ap-south-2	06:30–14:30 UTC
亚太地区（雅加达）	ap-southeast-3	08:00–16:00 UTC
亚太地区（马来西亚）	ap-southeast-5	09:00–17:00 UTC
亚太地区（墨尔本）	ap-southeast-4	11:00–19:00 UTC
亚太地区（孟买）	ap-south-1	16:30–00:30 UTC
Asia Pacific (Osaka)	ap-northeast-3	00:00–08:00 UTC
Asia Pacific (Seoul)	ap-northeast-2	13:00–21:00 UTC
亚太地区（新加坡）	ap-southeast-1	14:00–22:00 UTC
亚太地区（悉尼）	ap-southeast-2	12:00–20:00 UTC
亚太地区（东京)	ap-northeast-1	13:00–21:00 UTC
加拿大（中部）	ca-central-1	03:00–11:00 UTC

区域名称	区域	时间段
加拿大西部（卡尔加里）	ca-west-1	18:00–02:00 UTC
中国（北京）	cn-north-1	06:00–14:00 UTC
中国（宁夏）	cn-northwest-1	06:00–14:00 UTC
Europe (Frankfurt)	eu-central-1	20:00–04:00 UTC
欧洲（爱尔兰）	eu-west-1	22:00–06:00 UTC
欧洲地区（伦敦）	eu-west-2	22:00–06:00 UTC
欧洲（米兰）	eu-south-1	02:00–10:00 UTC
欧洲（巴黎）	eu-west-3	07:29–14:29 UTC
欧洲（西班牙）	eu-south-2	02:00–10:00 UTC
Europe (Stockholm)	eu-north-1	23:00–07:00 UTC
欧洲（苏黎世）	eu-central-2	02:00–10:00 UTC
以色列（特拉维夫）	il-central-1	03:00–11:00 UTC
中东（巴林）	me-south-1	06:00–14:00 UTC
中东（阿联酋）	me-central-1	05:00–13:00 UTC
南美洲（圣保罗）	sa-east-1	23:00–07:00 UTC
Amazon GovCloud（美国东部）	us-gov-east-1	17:00–01:00 UTC
Amazon GovCloud（美国西部）	us-gov-west-1	06:00–14:00 UTC

备份保留期

您可以在创建数据库实例或多可用区数据库集群时设置备份保留期。如果您使用 Amazon RDS API 或 Amazon CLI 创建数据库实例，并且未设置备份保留期，则默认备份保留期为一天。如果您使用控制台创建数据库实例，则默认备份保留期为七天。

创建数据库实例或集群后，您可以修改备份保留期。您可以将数据库实例的备份保留期设置为在 0 到 35 天之间。要禁用自动备份，请将备份保留期设置为 0。对于多可用区数据库集群，可以将备份保留期设置为在 1 到 35 天之间。手动快照限制（每个区域 100 个）不适用于自动备份。

> **Important**
>
> 如果将数据库实例的备份保留期从 0 更改为非零值或从非零值更改为 0，则会发生中断。

RDS 不包括计算备份保留期时处于 `stopped` 状态的时间。数据库实例或集群停止后，无法创建自动备份。如果数据库实例已停止，备份的保留时间可能会超过备份保留期。

启用自动备份

如果数据库实例尚未启用自动备份，您可以随时启用。您可以将备份保留期设置为非零正值来启用自动备份。在开启自动备份后，您的数据库实例将脱机并立即创建备份。

> **Note**
>
> 如果在 Amazon Backup 中管理备份，将无法启用自动备份。有关更多信息，请参阅使用 Amazon Backup 管理 Amazon RDS 的自动备份。

控制台

立即启用自动备份

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择数据库，然后选择要修改的数据库实例或多可用区数据库集群。
3. 选择 Modify(修改)。
4. 对于备份保留期，请选择非零正值，例如三天。

5. 选择 Continue (继续)。
6. 选择立即应用。
7. 选择修改数据库实例或修改集群以保存更改并启用自动备份。

Amazon CLI

要启用自动备份，请使用 Amazon CLI modify-db-instance 或 modify-db-cluster 命令。

包括以下参数：

- `--db-instance-identifier`（或 `--db-cluster-identifier` 用于可用区数据库集群）
- `--backup-retention-period`
- `--apply-immediately` 或者 `--no-apply-immediately`

在以下示例中，启用了自动备份并将备份保留期设置为三天。更改将立即应用。

Example

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier my_db_instance  \
    --backup-retention-period 3 \
    --apply-immediately
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier my_db_instance  ^
    --backup-retention-period 3 ^
    --apply-immediately
```

RDS API

要启用自动备份，请使用 RDS API ModifyDBInstance 或 ModifyDBCluster 操作以及以下必需参数：

- `DBInstanceIdentifier` 或 `DBClusterIdentifier`
- `BackupRetentionPeriod`

查看自动备份

要查看自动备份，请在导航窗格中选择 Automated backups（自动备份）。要查看与自动备份关联的单个快照，请在导航窗格中选择 Snapshots（快照）。或者，您可以描述与自动备份关联的各个快照。然后，您可以直接从这些快照之一还原数据库实例。

自动快照名称遵循模式 `rds:<database-name>-yyyy-mm-dd-hh-mm`，其中 `yyyy-mm-dd-hh-mm` 表示创建快照的日期和时间。

要使用 Amazon CLI 对现有的数据库实例的自动备份进行描述，请使用以下命令之一：

```
aws rds describe-db-instance-automated-backups --db-instance-
identifier DBInstanceIdentifier
```

或

```
aws rds describe-db-instance-automated-backups --dbi-resource-id DbiResourceId
```

要使用 RDS API 对现有的数据库实例的保留的自动备份进行描述，请使用以下参数之一调用 `DescribeDBInstanceAutomatedBackups` 操作：

- `DBInstanceIdentifier`
- `DbiResourceId`

保留自动备份

> **Note**
> 您只能保留数据库实例而不是多可用区数据库集群的自动备份。

删除数据库实例时，可以选择保留自动备份。自动备份可以保留的天数等于删除数据库实例时为该实例配置的备份保留期。

保留的自动备份包含系统快照和来自数据库实例的事务日志。还包括您的数据库实例属性，如分配的存储空间和数据库实例类，这些是将其还原到活动实例所必需的属性。

保留的自动备份和手动快照在被删除之前会产生账单费用。有关更多信息，请参阅 保留成本。

您可以为运行 Db2、MariaDB、MySQL、PostgreSQL、Oracle 和 Microsoft SQL Server 引擎的 RDS 实例保留自动备份。

您可以使用 Amazon Web Services Management Console、RDS API 和 Amazon CLI 还原或移除保留的自动备份。

主题

- 保留期
- 查看保留的备份
- 还原
- 保留成本
- 限制

保留期

保留的自动备份中的系统快照和事务日志与源数据库实例的系统快照和事务日志以同样的方式过期。因为没有为此实例创建任何新的快照或日志，所以保留的自动备份最终将完全过期。事实上，它们的寿命与最后一个系统快照一样长，取决于源实例被删除时所具有的保留周期设置。保留的自动备份将在最后一个系统快照过期时由系统删除。

您可以通过删除数据库实例的相同方式移除保留的自动备份。您可以使用控制台或 RDS API 操作 `DeleteDBInstanceAutomatedBackup` 移除保留的自动备份。

最终快照独立于保留的自动备份。我们强烈建议您制作最终快照，即使您保留自动备份也是如此，因为保留的自动备份最终将过期。最终快照不会过期。

查看保留的备份

要查看保留的自动备份，请在导航窗格中选择 Automated backups (自动备份)，然后选择 Retained (保留)。要查看与保留的自动备份关联的单个快照，请在导航窗格中选择 Snapshots (快照)。或者，您可以通过描述来查看与保留的自动备份关联的各个快照。然后，您可以直接从这些快照之一还原数据库实例。

要使用 Amazon CLI 对保留的自动备份进行描述，请使用以下命令：

```
aws rds describe-db-instance-automated-backups --dbi-resource-id DbiResourceId
```

要使用 RDS API 对保留的自动备份进行描述，请使用 `DbiResourceId` 参数调用 `DescribeDBInstanceAutomatedBackups` 操作。

还原

有关从自动备份还原数据库实例的信息，请参阅 将 Amazon RDS 的数据库实例还原到指定时间。

保留成本

保留的自动备份的成本是与之关联的系统快照的总存储空间的成本。事务日志或实例元数据没有额外费用。备份的所有其他定价规则适用于可还原的实例。

例如，假设正在运行的实例的总分配存储空间为 100 GB。此外，假设您具有 50 GB 的手动快照，另加 75 GB 与保留的自动备份关联的系统快照。在这种情况下，您将仅为额外的 25 GB 的备份存储空间付费，如下所示：(50 GB + 75 GB) – 100 GB = 25 GB。

限制

以下限制适用于保留的自动备份：

- 一个 Amazon 区域中保留的自动备份的最大数量为 40。这不包含在数据库实例限额中。您可以同时具有 40 个运行的数据库实例和额外的 40 个保留的自动备份。
- 保留的自动备份不包含有关参数或选项组的信息。
- 您可以将已删除的实例还原到删除时保留周期内的某个时间点。
- 您无法修改保留的自动备份。这是因为它包含在您删除源实例时已存在的系统备份、事务日志和数据库实例属性。

删除保留的自动备份

当不再需要保留的自动备份时，可以删除它们。

控制台

删除保留的自动备份

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Automated backups (自动备份)。
3. 在保留选项卡中，选择要删除的保留自动备份。
4. 对于 Actions，选择 Delete。

5. 在确认页面上，输入 **delete me** 并选择 Delete (删除)。

Amazon CLI

可以通过使用 Amazon CLI 命令 delete-db-instance-automated-backup 及以下选项删除保留的自动备份：

- `--dbi-resource-id` – 源数据库实例的资源标识符。

  可以通过运行 Amazon CLI 命令 describe-db-instance-automated-backups 查找保留自动备份的源数据库实例的资源标识符。

Example

以下示例删除具有源数据库实例资源标识符 `db-123ABCEXAMPLE` 的保留自动备份。

对于 Linux、macOS 或 Unix：

```
aws rds delete-db-instance-automated-backup \
    --dbi-resource-id db-123ABCEXAMPLE
```

对于 Windows：

```
aws rds delete-db-instance-automated-backup ^
    --dbi-resource-id db-123ABCEXAMPLE
```

RDS API

可以通过使用 Amazon RDS API 操作 DeleteDBInstanceAutomatedBackup 及以下参数删除保留的自动备份。

- `DbiResourceId` – 源数据库实例的资源标识符。

  可以通过使用 Amazon RDS API 操作 DescribeDBInstanceAutomatedBackups 查找保留自动备份的源数据库实例的资源标识符。

禁用自动备份

在某些情况下（例如，加载大量数据时），您可能希望临时禁用自动备份。

> **Important**
>
> 我们强烈建议不要禁用自动备份，因为该操作会禁用时间点恢复。禁用数据库实例或多可用区数据库集群的自动备份将删除此数据库的所有现有自动备份。如果在禁用后又重新启用自动备份，则只能从重新启用自动备份的时间开始进行还原。

控制台

立即禁用自动备份

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择数据库，然后选择要修改的数据库实例或多可用区数据库集群。
3. 选择修改。
4. 对于 Backup retention period (备份保留期)，选择 0 days (0 天)。
5. 选择 Continue (继续)。
6. 选择立即应用。
7. 选择修改数据库实例或修改集群以保存更改并禁用自动备份。

Amazon CLI

要立即禁用自动备份，请使用 modify-db-instance 或 modify-db-cluster 命令并使用 `--apply-immediately` 将备份保留期设置为 0。

Example

以下示例立即禁用多可用区数据库集群上的自动备份。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-cluster \
    --db-cluster-identifier mydbcluster \
    --backup-retention-period 0 \
    --apply-immediately
```

对于 Windows：

```
aws rds modify-db-cluster ^
    --db-cluster-identifier mydbcluster ^
    --backup-retention-period 0 ^
    --apply-immediately
```

要了解修改的生效时间，请对于数据库实例调用 `describe-db-instances`（或对于多可用区数据库集群调用 `describe-db-clusters`），直至备份保留期的值为 0 且 mydbcluster 状态可用。

```
aws rds describe-db-clusters --db-cluster-identifier mydcluster
```

RDS API

要立即禁用自动备份，请使用以下参数调用 ModifyDBInstance 或 ModifyDBCluster 操作：

- `DBInstanceIdentifier = mydbinstance`–、 或 `DBClusterIdentifier = mydbcluster`
- `BackupRetentionPeriod = 0`

Example

```
https://rds.amazonaws.com/
    ?Action=ModifyDBInstance
    &DBInstanceIdentifier=mydbinstance
    &BackupRetentionPeriod=0
    &SignatureVersion=2
    &SignatureMethod=HmacSHA256
    &Timestamp=2009-10-14T17%3A48%3A21.746Z
    &AWSAccessKeyId=<&AWS; Access Key ID>
    &Signature=<Signature>
```

使用不支持的 MySQL 存储引擎进行自动备份

对于 MySQL 数据库引擎，仅支持将自动备份用于 InnoDB 存储引擎。将这些功能与 MyISAM 等其他 MySQL 存储引擎配合使用，可能会导致在从备份还原时出现性能不可靠问题。具体而言，由于类似 MyISAM 的存储引擎不支持可靠的崩溃恢复，因此如果发生崩溃，就可能会损坏您的表。为此，我们建议您使用 InnoDB 存储引擎。

- 要将现有的 MyISAM 表转换为 InnoDB 表，您可以使用 `ALTER TABLE` 命令，例如：`ALTER TABLE table_name ENGINE=innodb, ALGORITHM=COPY;`

- 如果选择使用 MyISAM，您可以尝试使用 REPAIR 命令手动修复崩溃后损坏的表。有关更多信息，请参阅 MySQL 文档中的 REPAIR TABLE 语句。然而，正如 MySQL 文档所指出，您可能再也不能恢复所有数据。
- 如果要在还原前拍摄 MyISAM 表的快照，请执行以下步骤：
  1. 停止 MyISAM 表的所有活动（即关闭所有会话）。

     可以通过为从 SHOW FULL PROCESSLIST 命令返回的每个进程调用 mysql.rds_kill 命令来关闭所有会话。
  2. 锁定并刷新每个 MyISAM 表。例如，以下命令锁定并刷新两个名为 myisam_table1 和 myisam_table2 的表：

     ```
     mysql> FLUSH TABLES myisam_table, myisam_table2 WITH READ LOCK;
     ```
  3. 创建数据库实例或多可用区数据库集群的快照。完成快照后，释放 MyISAM 表上的锁定和恢复活动。可以使用以下命令为表解锁：

     ```
     mysql> UNLOCK TABLES;
     ```

  这些步骤强制 MyISAM 将存储在内存中的数据刷新到磁盘上，这可确保从数据库快照还原后，以清空状态开始下一操作。有关创建数据库快照的更多信息，请参阅为 Amazon RDS 的单可用区数据库实例创建数据库快照。

使用不支持的 MariaDB 存储引擎进行自动备份

对于 MariaDB 数据库引擎，仅支持将自动备份与 InnoDB 存储引擎配合使用。将这些功能与 Aria 等其他 MariaDB 存储引擎配合使用，可能会导致在从备份还原时出现性能不可靠问题。即使 Aria 是用来防止崩溃的 MyISAM 的替代品，您的表仍可能在崩溃时损坏。为此，我们建议您使用 InnoDB 存储引擎。

- 要将现有的 Aria 表转换为 InnoDB 表，您可以使用 ALTER TABLE 命令。例如：ALTER TABLE *table_name* ENGINE=innodb, ALGORITHM=COPY;
- 如果选择使用 Aria，您可以尝试使用 REPAIR TABLE 命令手动修复崩溃后损坏的表。有关更多信息，请参阅 http://mariadb.com/kb/en/mariadb/repair-table/。
- 如果要在还原前拍摄 Aria 表的快照，请执行以下步骤：
  1. 停止 Aria 表的所有活动 (即关闭所有会话)。
  2. 锁定并刷新每个 Aria 表。

3. 创建数据库实例或多可用区数据库集群的快照。完成快照后，释放 Aria 表上的锁定和恢复活动。这些步骤强制 Aria 将存储在内存中的数据刷新到磁盘上，从而确保从数据库快照还原后，以清空状态开始下一操作。

将自动备份复制到其他 Amazon Web Services 区域

要添加灾难恢复功能，您可以将 Amazon RDS 数据库实例配置为将快照和事务日志复制到所选的目标 Amazon Web Services 区域。为数据库实例配置备份复制时，RDS 会在数据库实例上的所有快照和事务日志准备就绪后立刻启动跨区域副本。

数据库快照副本需支付数据传输费用。复制数据库快照后，将为目标区域的存储量收取标准费用。有关更多详细信息，请参阅 RDS 定价。

有关使用备份复制的示例，请参阅 Amazon 在线技术研讨会：使用 Amazon RDS for Oracle 跨区域自动备份的托管式灾难恢复。

> **Note**
>
> 多可用区数据库集群不支持自动备份复制。有关自动备份的限制的更多信息，请参阅 Amazon RDS 的配额和限制。

有关为 Amazon RDS 配置和管理自动备份的信息，请参阅以下主题。

主题

- 启用 Amazon RDS 的跨区域自动备份
- 查找有关 Amazon RDS 的复制备份的信息
- 从 Amazon RDS 的复制备份还原到指定时间
- 停止 Amazon RDS 的自动备份复制
- 删除 Amazon RDS 的复制备份

区域和版本可用性

功能可用性和支持因每个数据库引擎的特定版本以及 Amazon Web Services 区域而异。有关跨区域自动备份的版本和区域可用性的更多信息，请参阅支持 Amazon RDS 中跨区域自动备份的区域和数据库引擎。

源和目标 Amazon Web Services 区域 支持

以下 Amazon Web Services 区域之间支持备份复制：

源区域	可用目标区域
非洲（开普敦）	欧洲地区（法兰克福）、欧洲地区（爱尔兰）、欧洲地区（伦敦）
亚太地区（香港）	亚太地区（新加坡）、亚太地区（东京）
亚太地区（海得拉巴）	亚太地区（孟买）
亚太地区（墨尔本）	亚太地区（悉尼）
亚太地区（马来西亚）	亚太地区（新加坡）
亚太地区（孟买）	亚太地区（海得拉巴）、亚太地区（新加坡） 美国东部（弗吉尼亚州北部）、美国东部（俄亥俄州）、美国西部（俄勒冈州）
亚太地区（大阪）	亚太区域（东京）
亚太地区 (首尔)	亚太地区（新加坡）、亚太地区（东京） 美国东部（弗吉尼亚州北部）、美国东部（俄亥俄州）、美国西部（俄勒冈州）
亚太地区（新加坡）	亚太地区（香港）、亚太地区（马来西亚）、亚太地区（孟买）、亚太地区（首尔）、亚太地区（悉尼）、亚太地区（东京） 美国东部（弗吉尼亚州北部）、美国东部（俄亥俄州）、美国西部（俄勒冈州）
亚太地区（悉尼）	亚太地区（墨尔本）、亚太地区（新加坡） 美国东部（弗吉尼亚州北部）、美国西部（加利福尼亚北部）、美国西部（俄勒冈州）
亚太地区（东京）	亚太地区（香港）、亚太地区（大阪）、亚太地区（首尔）、亚太地区（新加坡） 美国东部（弗吉尼亚州北部）、美国东部（俄亥俄州）、美国西部（俄勒冈州）

源区域	可用目标区域
加拿大 (中部)	加拿大西部（卡尔加里） 欧洲地区（爱尔兰） 美国东部（弗吉尼亚北部）、美国东部（俄亥俄）、美国西部（加利福尼亚北部）、美国西部（俄勒冈）
加拿大西部（卡尔加里）	加拿大（中部）
中国（北京）	中国（宁夏）
中国（宁夏）	中国（北京）
欧洲地区（法兰克福）	非洲（开普敦） 欧洲地区（爱尔兰）、欧洲地区（伦敦）、欧洲地区（巴黎）、欧洲地区（斯德哥尔摩）、欧洲（苏黎世） 美国东部（弗吉尼亚州北部）、美国东部（俄亥俄州）、美国西部（俄勒冈州）
欧洲地区（爱尔兰）	非洲（开普敦） 加拿大（中部） 欧洲地区（法兰克福）、欧洲地区（伦敦）、欧洲地区（巴黎）、欧洲地区（斯德哥尔摩）、欧洲（苏黎世） 美国东部（弗吉尼亚州北部）、美国东部（俄亥俄州）、美国西部（加利福尼亚北部）、美国西部（俄勒冈州）
欧洲地区（伦敦）	非洲（开普敦） 欧洲（法兰克福）、欧洲（爱尔兰）、欧洲（巴黎）、欧洲（斯德哥尔摩） 美国东部（弗吉尼亚北部）

源区域	可用目标区域
欧洲（巴黎）	欧洲（法兰克福）、欧洲（伦敦）、欧洲（巴黎）、欧洲（斯德哥尔摩） 美国东部（弗吉尼亚北部）
Europe (Stockholm)	欧洲（法兰克福）、欧洲（伦敦）、欧洲（巴黎）、欧洲（斯德哥尔摩） 美国东部（弗吉尼亚州北部）
欧洲（苏黎世）	欧洲（法兰克福）、欧洲（爱尔兰）
南美洲（圣保罗）	美国东部（弗吉尼亚州北部）、美国东部（俄亥俄州）
Amazon GovCloud（美国东部）	Amazon GovCloud（美国西部）
Amazon GovCloud（美国西部）	Amazon GovCloud（美国东部）
美国东部（弗吉尼亚北部）	亚太地区（孟买）、亚太地区（首尔）、亚太地区（悉尼）、亚太地区（东京） Canada (Central) 欧洲（法兰克福）、欧洲（伦敦）、欧洲（巴黎）、欧洲（斯德哥尔摩） 南美洲（圣保罗） 美国东部（弗吉尼亚州北部）、美国西部（加利福尼亚北部）、美国西部（俄勒冈州）

源区域	可用目标区域
US East (Ohio)	亚太地区（孟买）、亚太地区（首尔）、亚太地区（悉尼）、亚太地区（东京） Canada (Central) 欧洲（法兰克福）、欧洲（爱尔兰） 南美洲（圣保罗） 美国东部（弗吉尼亚州北部）、美国西部（加利福尼亚北部）、美国西部（俄勒冈州）
美国西部（加利福尼亚北部）	亚太地区（悉尼） 加拿大（中部） 欧洲（爱尔兰） 美国东部（弗吉尼亚州北部）、美国东部（俄亥俄州）、美国西部（俄勒冈州）
美国西部（俄勒冈州）	亚太地区（孟买）、亚太地区（首尔）、亚太地区（悉尼）、亚太地区（东京） Canada (Central) 欧洲（法兰克福）、欧洲（爱尔兰） 美国东部（弗吉尼亚州北部）、美国东部（俄亥俄州）、美国西部（加利福尼亚北部）

您还可以使用 `describe-source-regions` Amazon CLI 命令找出哪些 Amazon Web Services 区域可以相互复制。有关更多信息，请参阅 [查找有关 Amazon RDS 的复制备份的信息]。

限制

以下限制适用于 Amazon RDS 的跨区域自动备份。

- 多可用区数据库集群不支持自动备份复制。

- 默认情况下，每个 Amazon Web Services 账户最多具有 20 个跨区域自动备份。

启用 Amazon RDS 的跨区域自动备份

您可以使用 Amazon RDS 控制台对新的或现有的数据库实例启用备份复制。您还可以使用 `start-db-instance-automated-backups-replication` Amazon CLI 命令或 `StartDBInstanceAutomatedBackupsReplication` RDS API 操作。对于每个 Amazon Web Services 区域，您可以将多达 20 个备份复制到每个目标 Amazon Web Services 账户。

> **Note**
>
> 为了能够复制自动备份，请确保启用它们。有关更多信息，请参阅 启用自动备份。

控制台

您可以为新的或现有的数据库实例启用备份复制：

- 对于新的数据库实例，请在启动实例时将其启用。有关更多信息，请参阅"数据库实例的设置"。
- 对于现有数据库实例，请按以下流程操作。

如欲为现有数据库实例启用备份复制

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Automated backups (自动备份)。
3. 在当前区域选项卡上，选择要为其启用备份复制的数据库实例。
4. 对于操作，选择管理跨区域复制。
5. 在 Backup replication（备份复制）下，选择 Enable replication to another Amazon Web Services 区域（启用复制到其他 Amazon Web Services 区域）。
6. 选择目标区域。
7. 选择复制备份保留期。
8. 如果您已在源数据库实例上启用加密，请选择用于加密备份的 Amazon KMS key 或输入密钥 ARN。
9. 选择保存。

在源区域中，复制备份在自动备份页面的当前区域选项卡上列出。在目标区域中，复制备份在自动备份页面的复制备份选项卡上列出。

Amazon CLI

使用 `start-db-instance-automated-backups-replication` Amazon CLI 命令启用备份复制。

以下 CLI 示例将自动备份从美国西部（俄勒冈州）区域中的数据库实例复制到美国东部（弗吉尼亚州北部）地区。它还使用目标区域中的 Amazon KMS key 对复制的备份进行加密。

如欲启用备份复制

- 运行以下命令之一。

  对于 Linux、macOS 或 Unix：

```
aws rds start-db-instance-automated-backups-replication \
--region us-east-1 \
--source-db-instance-arn "arn:aws:rds:us-west-2:123456789012:db:mydatabase" \
--kms-key-id "arn:aws:kms:us-east-1:123456789012:key/AKIAIOSFODNN7EXAMPLE" \
--backup-retention-period 7
```

  对于 Windows：

```
aws rds start-db-instance-automated-backups-replication ^
--region us-east-1 ^
--source-db-instance-arn "arn:aws:rds:us-west-2:123456789012:db:mydatabase" ^
--kms-key-id "arn:aws:kms:us-east-1:123456789012:key/AKIAIOSFODNN7EXAMPLE" ^
--backup-retention-period 7
```

RDS API

使用带以下参数的 `StartDBInstanceAutomatedBackupsReplication` RDS API 操作来启用备份复制：

- `Region`（如果您不是从目标区域调用 API 操作）
- `SourceDBInstanceArn`
- `BackupRetentionPeriod`
- `KmsKeyId`（可选）

- `PreSignedUrl`（在您使用 时必需）`KmsKeyId`

> **Note**
>
> 如果您对备份进行加密，则还必须包含预签名 URL。有关预签名 URL 的更多信息，请参阅 Amazon Simple Storage Service API 参考中的验证请求：使用查询参数（Amazon签名版本 4）和Amazon一般参考中的签名版本 4 签名流程。

查找有关 Amazon RDS 的复制备份的信息

您可以使用以下 CLI 命令查找有关复制备份的信息：

- `describe-source-regions`
- `describe-db-instances`
- `describe-db-instance-automated-backups`

以下 `describe-source-regions` 示例列出了其中的自动备份可以复制到美国西部（俄勒冈州）目标区域的源 Amazon Web Services 区域。

如欲显示有关源区域的信息

- 运行以下命令。

```
aws rds describe-source-regions --region us-west-2
```

输出显示备份可以从 US East (N. Virginia) 而不能从 美国东部（俄亥俄州） 或 美国西部（加利福尼亚北部） 复制到 美国西部（俄勒冈州） 中。

```
{
    "SourceRegions": [
        ...
        {
            "RegionName": "us-east-1",
            "Endpoint": "https://rds.us-east-1.amazonaws.com",
            "Status": "available",
            "SupportsDBInstanceAutomatedBackupsReplication": true
        },
```

```
        {
            "RegionName": "us-east-2",
            "Endpoint": "https://rds.us-east-2.amazonaws.com",
            "Status": "available",
            "SupportsDBInstanceAutomatedBackupsReplication": false
        },
            "RegionName": "us-west-1",
            "Endpoint": "https://rds.us-west-1.amazonaws.com",
            "Status": "available",
            "SupportsDBInstanceAutomatedBackupsReplication": false
        }
    ]
}
```

以下 `describe-db-instances` 示例显示了数据库实例的自动备份。

如欲显示数据库实例的复制备份

- 运行以下命令之一。

  对于 Linux、macOS 或 Unix：

  ```
  aws rds describe-db-instances \
  --db-instance-identifier mydatabase
  ```

  对于 Windows：

  ```
  aws rds describe-db-instances ^
  --db-instance-identifier mydatabase
  ```

输出包括复制备份。

```
{
    "DBInstances": [
        {
            "StorageEncrypted": false,
            "Endpoint": {
                "HostedZoneId": "Z1PVIF0B656C1W",
                "Port": 1521,
            ...
```

```
            "BackupRetentionPeriod": 7,
            "DBInstanceAutomatedBackupsReplications":
 [{"DBInstanceAutomatedBackupsArn": "arn:aws:rds:us-east-1:123456789012:auto-backup:ab-L2IJCEXJP7XQ7HOJ4SIEXAMPLE"}]
        }
    ]
}
```

以下 `describe-db-instance-automated-backups` 示例显示了数据库实例的自动备份。

如欲显示数据库实例的自动备份

- 运行以下命令之一。

  对于 Linux、macOS 或 Unix：

  ```
  aws rds describe-db-instance-automated-backups \
  --db-instance-identifier mydatabase
  ```

  对于 Windows：

  ```
  aws rds describe-db-instance-automated-backups ^
  --db-instance-identifier mydatabase
  ```

输出显示了 美国西部（俄勒冈州） 中的源数据库实例和自动备份，且备份已复制到 US East (N. Virginia)。

```
{
    "DBInstanceAutomatedBackups": [
        {
            "DBInstanceArn": "arn:aws:rds:us-west-2:868710585169:db:mydatabase",
            "DbiResourceId": "db-L2IJCEXJP7XQ7HOJ4SIEXAMPLE",
            "DBInstanceAutomatedBackupsArn": "arn:aws:rds:us-west-2:123456789012:auto-backup:ab-L2IJCEXJP7XQ7HOJ4SIEXAMPLE",
            "BackupRetentionPeriod": 7,
            "DBInstanceAutomatedBackupsReplications":
 [{"DBInstanceAutomatedBackupsArn": "arn:aws:rds:us-east-1:123456789012:auto-backup:ab-L2IJCEXJP7XQ7HOJ4SIEXAMPLE"}]
            "Region": "us-west-2",
            "DBInstanceIdentifier": "mydatabase",
            "RestoreWindow": {
```

```
                "EarliestTime": "2020-10-26T01:09:07Z",
                "LatestTime": "2020-10-31T19:09:53Z",
            }
            ...
        }
    ]
}
```

以下 describe-db-instance-automated-backups 示例使用 --db-instance-automated-backups-arn 选项来显示目标区域中的复制备份。

如欲显示复制备份

- 运行以下命令之一。

  对于 Linux、macOS 或 Unix：

```
aws rds describe-db-instance-automated-backups \
--db-instance-automated-backups-arn "arn:aws:rds:us-east-1:123456789012:auto-backup:ab-L2IJCEXJP7XQ7HOJ4SIEXAMPLE"
```

  对于 Windows：

```
aws rds describe-db-instance-automated-backups ^
--db-instance-automated-backups-arn "arn:aws:rds:us-east-1:123456789012:auto-backup:ab-L2IJCEXJP7XQ7HOJ4SIEXAMPLE"
```

输出显示了 美国西部（俄勒冈州） 中的源数据库实例，以及 US East (N. Virginia) 中的复制备份。

```
{
    "DBInstanceAutomatedBackups": [
        {
            "DBInstanceArn": "arn:aws:rds:us-west-2:868710585169:db:mydatabase",
            "DbiResourceId": "db-L2IJCEXJP7XQ7HOJ4SIEXAMPLE",
            "DBInstanceAutomatedBackupsArn": "arn:aws:rds:us-east-1:123456789012:auto-backup:ab-L2IJCEXJP7XQ7HOJ4SIEXAMPLE",
            "Region": "us-west-2",
            "DBInstanceIdentifier": "mydatabase",
            "RestoreWindow": {
                "EarliestTime": "2020-10-26T01:09:07Z",
                "LatestTime": "2020-10-31T19:01:23Z"
```

```
            },
            "AllocatedStorage": 50,
            "BackupRetentionPeriod": 7,
            "Status": "replicating",
            "Port": 1521,
            ...
        }
    ]
}
```

从 Amazon RDS 的复制备份还原到指定时间

您可以使用 Amazon RDS 控制台将数据库实例从复制备份还原到特定时间点。您还可以使用 `restore-db-instance-to-point-in-time` Amazon CLI 命令或 `RestoreDBInstanceToPointInTime` RDS API 操作。

有关时间点恢复 (PITR) 的一般信息，请参阅 将 Amazon RDS 的数据库实例还原到指定时间。

> **Note**
>
> 在跨 Amazon Web Services 区域复制自动备份时，请注意以下数据库引擎限制：
>
> - 在 RDS for SQL Server 上，不会复制选项组。
> - 在 RDS for Oracle 上，不会复制以下选项：`NATIVE_NETWORK_ENCRYPTION`、`OEM`、`OEM_AGENT` 和 `SSL`。
>
> 如果您已将自定义选项组与 数据库实例关联，则可以在目标区域中重新创建该选项组。然后在目标区域中还原数据库实例，并将自定义选项组与其关联。有关更多信息，请参阅 使用选项组。

控制台

将数据库实例从复制备份还原到指定时间

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 从区域选择器中选择目标区域（备份复制到其中的区域）。
3. 在导航窗格中，选择 Automated backups (自动备份)。

4. 在复制备份选项卡上，选择要还原的数据库实例。
5. 对于 Actions (操作)，选择 Restore to point in time (还原到时间点)。
6. 选择最近可还原时间以还原到可能的最近时间，或选择自定义来选择时间。

   如果您选择自定义，请输入要将实例还原到的日期和时间。

   > **Note**
   >
   > 时间以您的本地时区显示，表示为协调世界时 (UTC) 的偏移量。例如，UTC-5 是东部标准时间/中部夏令时。

7. 对于数据库实例标识符，请输入目标还原后的数据库实例的名称。
8. （可选）根据需要选择其他选项，例如启用自动扩展。
9. 选择还原到时间点。

Amazon CLI

使用命令 restore-db-instance-to-point-in-time Amazon CLI 创建新的数据库实例。

将数据库实例从复制备份还原到指定时间

- 运行以下命令之一。

  对于 Linux、macOS 或 Unix：

```
aws rds restore-db-instance-to-point-in-time \
    --source-db-instance-automated-backups-arn "arn:aws:rds:us-
east-1:123456789012:auto-backup:ab-L2IJCEXJP7XQ7HOJ4SIEXAMPLE" \
    --target-db-instance-identifier mytargetdbinstance \
    --restore-time 2020-10-14T23:45:00.000Z
```

  对于 Windows：

```
aws rds restore-db-instance-to-point-in-time ^
    --source-db-instance-automated-backups-arn "arn:aws:rds:us-
east-1:123456789012:auto-backup:ab-L2IJCEXJP7XQ7HOJ4SIEXAMPLE" ^
    --target-db-instance-identifier mytargetdbinstance ^
    --restore-time 2020-10-14T23:45:00.000Z
```

RDS API

要将数据库实例还原到指定时间，请结合以下参数调用 [RestoreDBInstanceToPointInTime](#) Amazon RDS 操作：

- `SourceDBInstanceAutomatedBackupsArn`
- `TargetDBInstanceIdentifier`
- `RestoreTime`

停止 Amazon RDS 的自动备份复制

您可以使用 Amazon RDS 控制台停止数据库实例的备份复制。您还可以使用 `stop-db-instance-automated-backups-replication` Amazon CLI 命令或 `StopDBInstanceAutomatedBackupsReplication` RDS API 操作。

复制备份已保留，但受创建时设置的备份保留期的限制。

控制台

从源区域的自动备份页面停止备份复制。

停止向 Amazon Web Services 区域的备份复制

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 从区域选择器中选择源区域。
3. 在导航窗格中，选择 Automated backups (自动备份)。
4. 在当前区域选项卡上，选择要停止备份复制的数据库实例。
5. 对于操作，选择管理跨区域复制。
6. 在 Backup replication（备份复制）下，清除 Enable replication to another Amazon Web Services 区域（启用复制到其他 Amazon Web Services 区域）复选框。
7. 选择保存。

在目标区域中，复制备份在自动备份页面的保留选项卡上列出。

Amazon CLI

使用 [stop-db-instance-automated-backups-replication](#) Amazon CLI 命令停止备份复制。

以下 CLI 示例停止在 美国西部（俄勒冈州） 区域中复制数据库实例的自动备份。

如欲停止备份复制

- 运行以下命令之一。

  对于 Linux、macOS 或 Unix：

```
aws rds stop-db-instance-automated-backups-replication \
--region us-east-1 \
--source-db-instance-arn "arn:aws:rds:us-west-2:123456789012:db:mydatabase"
```

  对于 Windows：

```
aws rds stop-db-instance-automated-backups-replication ^
--region us-east-1 ^
--source-db-instance-arn "arn:aws:rds:us-west-2:123456789012:db:mydatabase"
```

RDS API

使用带以下参数的 `StopDBInstanceAutomatedBackupsReplication` RDS API 操作停止备份复制：

- `Region`
- `SourceDBInstanceArn`

删除 Amazon RDS 的复制备份

您可以使用 Amazon RDS 控制台删除数据库实例的复制备份。您还可以使用 `delete-db-instance-automated-backups` Amazon CLI 命令或 `DeleteDBInstanceAutomatedBackup` RDS API 操作。

控制台

从自动备份页面中删除目标区域中的复制备份。

如欲删除复制备份

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。

2. 从区域选择器中选择目标区域。
3. 在导航窗格中，选择 Automated backups (自动备份)。
4. 在复制备份选项卡上，选择要删除其复制备份的数据库实例。
5. 对于 Actions (操作)，选择 Delete (删除)。
6. 在确认页面上，输入 **delete me** 并选择 Delete (删除)。

Amazon CLI

使用 delete-db-instance-automated-backup Amazon CLI 命令删除复制备份。

您可以使用 describe-db-instances CLI 命令查找复制备份的 Amazon Resource Name (ARN)。有关更多信息，请参阅“查找有关 Amazon RDS 的复制备份的信息”。

如欲删除复制备份

- 运行以下命令之一。

  对于 Linux、macOS 或 Unix：

```
aws rds delete-db-instance-automated-backup \
--db-instance-automated-backups-arn "arn:aws:rds:us-east-1:123456789012:auto-backup:ab-L2IJCEXJP7XQ7HOJ4SIEXAMPLE"
```

  对于 Windows：

```
aws rds delete-db-instance-automated-backup ^
--db-instance-automated-backups-arn "arn:aws:rds:us-east-1:123456789012:auto-backup:ab-L2IJCEXJP7XQ7HOJ4SIEXAMPLE"
```

RDS API

使用带 DeleteDBInstanceAutomatedBackup 参数的 DBInstanceAutomatedBackupsArn RDS API 操作删除复制备份。

管理手动备份

本节介绍如何管理数据库实例和数据库集群的手动备份。

有关为Single-AZ数据库实例创建数据库快照的信息，请参阅为 Amazon RDS 的单可用区数据库实例创建数据库快照。

有关为多可用区数据库集群创建数据库快照的信息，请参阅创建 Amazon RDS 的多可用区数据库集群快照。

有关删除数据库快照的信息，请参阅删除 Amazon RDS 的数据库快照。

为 Amazon RDS 的单可用区数据库实例创建数据库快照

Amazon RDS 创建数据库实例的存储卷快照，并备份整个数据库实例而不仅仅是单个数据库。在单可用区数据库实例上创建此数据库快照会导致 I/O 短暂性停止，可持续数秒至数分钟，具体取决于数据库实例的大小和类。对于 MariaDB、MySQL、Oracle 和 PostgreSQL，多可用区部署的备份期间不会暂停主数据库上的 I/O 活动，因为备份获取自备用数据库。对于 SQL Server，多可用区部署的备份期间将短时间暂停 I/O 活动。

创建数据库快照时，需要识别出将要备份的数据库实例，然后为数据库快照命名，以便稍后从此快照还原。创建快照所用时间因数据库大小而异。由于快照包含整个存储卷，因此，文件 (如临时文件) 的大小也会影响创建快照所需时间。

> **Note**
>
> 数据库实例必须处于 `available` 状态才能拍摄数据库快照。
>
> 对于 PostgreSQL 数据库实例，未记录表中的数据可能无法从快照还原。有关更多信息，请参阅"使用 PostgreSQL 的最佳实践"。

与自动备份不同，手动快照不受备份保留期的限制。快照不会过期。

对于 MariaDB、MySQL 和 PostgreSQL 数据的非常长期的备份，我们建议将快照数据导出到 Amazon S3。如果不再支持数据库引擎的主要版本，则无法从快照还原到该版本。有关更多信息，请参阅"将 Amazon RDS 的数据库快照数据导出到 Amazon S3"。

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 创建数据库快照。

控制台

创建数据库快照

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择快照。

   将显示手动快照列表。
3. 选择拍摄快照。

   将显示 Take DB snapshot (拍摄数据库快照) 窗口。

4. 选择您要拍摄快照的数据库实例。
5. 输入快照名称。
6. 选择拍摄快照。

此时将显示手动快照列表，其中新数据库快照的状态显示为 `Creating`。在其状态为 `Available` 后，您可以看到其创建时间。

Amazon CLI

使用 Amazon CLI 创建数据库快照时，需要识别出将要备份的数据库实例，然后为数据库快照命名，以便稍后从此快照还原。结合以下参数使用 Amazon CLI [`create-db-snapshot`] 命令执行该操作：

- `--db-instance-identifier`
- `--db-snapshot-identifier`

在此示例中，您将给名为 *`mydbinstance`* 的数据库实例创建名为 *`mydbsnapshot`* 的数据库快照。

Example

对于 Linux、macOS 或 Unix：

```
aws rds create-db-snapshot \
    --db-instance-identifier mydbinstance \
    --db-snapshot-identifier mydbsnapshot
```

对于 Windows：

```
aws rds create-db-snapshot ^
    --db-instance-identifier mydbinstance ^
    --db-snapshot-identifier mydbsnapshot
```

RDS API

使用 Amazon RDS API 创建数据库快照时，需要指定要备份的数据库实例，然后为数据库快照命名，以便稍后从此快照还原。您可以使用具有以下参数的 Amazon RDS API [`CreateDBSnapshot`] 命令执行该操作：

- `DBInstanceIdentifier`
- `DBSnapshotIdentifier`

创建 Amazon RDS 的多可用区数据库集群快照

创建多可用区数据库集群快照时，请确保标识要备份的多可用区数据库集群，然后为数据库集群快照命名，以便稍后从此快照还原。您也可以共享多可用区数据库集群快照。有关说明，请参阅 the section called “共享数据库 快照”。

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 创建多可用区数据库集群快照。

对于非常长期的备份，我们建议将快照数据导出到 Amazon S3。如果不再支持数据库引擎的主要版本，则无法从快照还原到该版本。有关更多信息，请参阅 将 Amazon RDS 的数据库快照数据导出到 Amazon S3。

控制台

创建数据库集群快照

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases（数据库）。
3. 在列表中，请选择要为其创建快照的多可用区数据库集群。
4. 对于操作，选择拍摄快照。

   将显示 Take DB snapshot (拍摄数据库快照) 窗口。
5. 在 Snapshot name（快照名称）框中，请输入快照的名称。
6. 选择拍摄快照。

Snapshots（快照）页面将会显示，其中新多可用区数据库集群快照的状态将显示为 `Creating`。在其状态为 `Available` 后，您可以看到其创建时间。

Amazon CLI

您可以使用 Amazon CLI create-db-cluster-snapshot 命令来创建多可用区数据库集群快照，该命令带有以下选项：

- `--db-cluster-identifier`
- `--db-cluster-snapshot-identifier`

在此示例中，您将为名为 *mymultiazdbcluster* 的数据库集群创建名为 *mymultiazdbclustersnapshot* 的多可用区数据库集群快照。

Example

对于 Linux、macOS 或 Unix：

```
aws rds create-db-cluster-snapshot \
    --db-cluster-identifier mymultiazdbcluster \
    --db-cluster-snapshot-identifier mymultiazdbclustersnapshot
```

对于 Windows：

```
aws rds create-db-cluster-snapshot ^
    --db-cluster-identifier mymultiazdbcluster ^
    --db-cluster snapshot-identifier mymultiazdbclustersnapshot
```

RDS API

您可以使用 Amazon RDS API CreateDBClusterSnapshot 操作创建多可用区数据库集群快照，并采用以下参数：

- `DBClusterIdentifier`
- `DBClusterSnapshotIdentifier`

删除多可用区数据库集群快照

如果不再需要，您可以删除 Amazon RDS 管理的多可用区数据库快照。有关说明，请参阅 the section called “删除数据库快照”。

删除 Amazon RDS 的数据库快照

如果不再需要，您可以删除 Amazon RDS 管理的数据库快照。

> **Note**
>
> 要删除 Amazon Backup 管理的备份，请使用 Amazon Backup 控制台。有关 Amazon Backup 的更多信息，请参阅 Amazon Backup 开发人员指南。

删除数据库快照

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 删除手动、共享或公共数据库快照。

要删除共享或公共快照，您必须登录到拥有快照的 Amazon 账户。

如果您有要删除的自动数据库快照，但不希望删除数据库实例，请将数据库实例的备份保留期更改为 0。当更改生效时，自动快照将被删除。如果您不想等到下一个维护期，则可以立即应用更改。完成更改后，您可以通过将备份保留期设置为大于 0 的数字重新启用自动备份。有关修改 数据库实例的信息，请参阅修改 Amazon RDS 数据库实例。

保留的自动备份和手动快照会一直产生账单费用，直到将其删除为止。有关更多信息，请参阅 保留成本。

如果您删除了数据库实例，您可以通过删除数据库实例的自动备份删除其自动数据库快照。有关自动备份的信息，请参阅 备份简介。

控制台

删除数据库快照

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择快照。

   将显示手动快照列表。
3. 选择要删除的数据库快照。
4. 对于 Actions（操作），选择 Delete snapshot（删除快照）。

5. 在确认页面上选择 Delete (删除)。

Amazon CLI

您可以使用 Amazon CLI 命令 delete-db-snapshot 删除数据库快照。

以下选项用于删除数据库快照。

- `--db-snapshot-identifier` – 数据库快照的标识符。

Example

以下代码删除 `mydbsnapshot` 数据库快照。

对于 Linux、macOS 或 Unix：

```
aws rds delete-db-snapshot \
    --db-snapshot-identifier mydbsnapshot
```

对于 Windows：

```
aws rds delete-db-snapshot ^
    --db-snapshot-identifier mydbsnapshot
```

RDS API

您可以使用 Amazon RDS API 操作 DeleteDBSnapshot 删除数据库快照。

以下参数用于删除数据库快照。

- `DBSnapshotIdentifier` – 数据库快照的标识符。

还原到数据库实例

本节说明如何还原到数据库实例。此页面说明如何从数据库快照还原到 Amazon RDS 数据库实例。

Amazon RDS 创建数据库实例的存储卷快照，并备份整个数据库实例而不仅仅是单个数据库。您可以通过从数据库快照还原来创建新的数据库实例。您可以提供用于还原的数据库快照的名称，然后提供还原后新建的数据库实例的名称。无法从数据库快照还原到现有数据库实例；还原时将新建一个数据库实例。

您可以使用已还原的数据库实例，只要其状态为 `available`。数据库实例继续在后台加载数据。这称为延迟加载。

如果您的访问数据尚未加载，数据库实例会立即从 Amazon S3 下载请求的数据，然后在后台继续加载数据的剩余部分。有关更多信息，请参阅 Amazon EBS 快照。

为了帮助减轻延迟加载对需要快速访问的表的影响，您可以执行涉及全表扫描的操作，例如 `SELECT *`。这样，Amazon RDS 就可以从 S3 下载所有备份表数据。

您可以还原数据库实例并使用与源数据库快照不同的存储类型。在这种情况下，由于将数据迁移到新的存储类型需要额外的工作，还原过程会更慢。在还原到磁性存储或从其中还原时，迁移过程最慢。这是因为磁性存储不具备预置 IOPS 或通用型 (SSD) 存储的 IOPS 功能。

您可以使用 Amazon CloudFormation从数据库实例快照还原数据库实例。有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的 AWS::RDS::DBInstance。

> **Note**
>
> 您无法从共享并且加密的数据库快照恢复数据库实例。您可以改为创建数据库快照副本，并从该副本还原数据库实例。有关更多信息，请参阅 复制 Amazon RDS 的数据库快照。

有关使用 RDS 扩展支持版本还原数据库实例的信息，请参阅使用 Amazon RDS 扩展支持还原数据库实例或多可用区集群。

从快照还原

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 从数据库快照中还原数据库实例。

> **Note**
>
> 还原数据库实例时，无法减少存储量。当您增加分配的存储空间时，必须至少增加 10%。如果您尝试将该值增加不到 10%，则会得到错误。您无法在还原 RDS for SQL Server 数据库实例时增加分配的存储。

控制台

要从数据库快照还原数据库实例，请执行以下操作：

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择快照。
3. 选择要从其还原的数据库快照。
4. 对于操作，选择还原快照。
5. 在 Restore snapshot（还原快照）页面上，对于 DB instance identifier（数据库实例标识符），输入还原的数据库实例的名称。
6. 指定其他设置，如分配的存储大小。

   有关每项设置的信息，请参阅 数据库实例的设置。
7. 选择还原数据库实例。

Amazon CLI

要从数据库快照还原数据库实例，请使用 Amazon CLI 命令 restore-db-instance-from-db-snapshot。

在该示例中，您将从先前创建的名为 `mydbsnapshot` 的数据库快照中进行还原。还原为名为 `mynewdbinstance` 的新数据库实例。此示例还设置分配的存储大小。

您可以指定其他设置。有关每项设置的信息，请参阅 数据库实例的设置。

Example

对于 Linux、macOS 或 Unix：

```
aws rds restore-db-instance-from-db-snapshot \
```

```
    --db-instance-identifier mynewdbinstance \
    --db-snapshot-identifier mydbsnapshot \
    --allocated-storage 100
```

对于 Windows：

```
aws rds restore-db-instance-from-db-snapshot ^
    --db-instance-identifier mynewdbinstance ^
    --db-snapshot-identifier mydbsnapshot ^
    --allocated-storage 100
```

该命令会返回类似以下内容的输出：

```
DBINSTANCE  mynewdbinstance  db.t3.small  MySQL     50       sa              creating
 3  n  8.0.28  general-public-license
```

RDS API

要从数据库快照还原数据库实例，请使用以下参数调用 Amazon RDS API 函数 RestoreDBInstanceFromDBSnapshot：

- `DBInstanceIdentifier`
- `DBSnapshotIdentifier`

注意事项

有关从数据库快照还原到数据库实例时的注意事项，请参阅以下主题。

主题

- 参数组注意事项
- 安全组注意事项
- 选项组注意事项
- 资源标记注意事项
- Db2 注意事项
- Microsoft SQL Server 注意事项
- MySQL 注意事项

- Oracle Database 注意事项

参数组注意事项

我们建议您保留所创建的数据库快照的数据库参数组，以便还原的数据库实例可以与正确的参数组关联。

除非选择不同的实例，否则默认数据库参数组将与还原的实例关联。默认参数组中没有可用的自定义参数设置。

您可以在还原数据库实例时指定参数组。

有关数据库参数组的更多信息，请参阅 Amazon RDS 的参数组。

安全组注意事项

还原数据库实例时，默认的虚拟私有云（VPC）、数据库子网组和 VPC 安全组将与还原的实例关联，除非您选择不同的值。

- 如果您使用 Amazon RDS 控制台，则可以指定要与实例关联的自定义 VPC 安全组，或者创建新的 VPC 安全组。
- 如果您使用 Amazon CLI，则可以在 `restore-db-instance-from-db-snapshot` 命令中包括 `--vpc-security-group-ids` 选项，从而指定要与实例关联的自定义 VPC 安全组。
- 如果您使用的是 Amazon RDS API，则可以在 `VpcSecurityGroupIds.VpcSecurityGroupId.N` 操作中包括 `RestoreDBInstanceFromDBSnapshot` 参数。

一旦完成还原，并且您的新数据库实例可用，您还可以通过修改数据库实例来更改 VPC 设置。有关更多信息，请参阅 修改 Amazon RDS 数据库实例。

选项组注意事项

还原数据库实例时，默认的数据库选项组在大多数情况下会与还原的数据库实例相关联。

例外情况是，源数据库实例与包含持久性或永久性选项的选项组关联。例如，如果源数据库实例使用 Oracle 透明数据加密 (TDE)，则还原后的数据库实例必须使用具有 TDE 选项的选项组。

如果您将数据库实例还原到不同的 VPC 中，则必须执行以下操作之一才能分配数据库选项组：

- 将该 VPC 组的默认选项组分配到实例。
- 分配关联到该 VPC 的另一个选项组。
- 创建新选项组并将其分配到数据库实例。对于持久性或永久性选项（如 Oracle TDE），您必须创建包含持久性或永久性选项的新选项组。

有关数据库选项组的更多信息，请参阅 使用选项组。

资源标记注意事项

从数据库快照还原数据库实例时，RDS 会检查您是否指定新标签。如果是，新标签将添加到还原的数据库实例中。如果没有新的标签，RDS 会在创建快照时将源数据库实例中的标签添加到还原的数据库实例中。

有关更多信息，请参阅 将标签复制到数据库快照。

Db2 注意事项

使用 BYOL 模型时，您的 Amazon RDS for Db2 数据库实例必须与包含您的 IBM Site ID 和 IBM Customer ID 的自定义参数组相关联。否则，尝试从快照还原数据库实例将失败。Amazon RDS for Db2 数据库实例还必须与 Amazon License Manager 自行管理许可证相关联。有关更多信息，请参阅 自带 Db2 许可。

采用“通过 Amazon Web Services Marketplace 获得 Db2 许可证”模式时，您需要对要使用的特定 IBM Db2 版本进行有效 Amazon Web Services Marketplace 订阅。如果您还没有订阅，请在 Amazon Web Services Marketplace 中为该 IBM Db2 版本订阅 Db2。有关更多信息，请参阅 通过 Amazon Web Services Marketplace 获得 Db2 许可证。

Microsoft SQL Server 注意事项

如果您要将 RDS for Microsoft SQL Server 数据库快照还原为新实例，总是可以还原为与快照相同的版本。在某些情况下，您还可以更改数据库实例的版本。以下是更改版本的限制应用：

- 数据库快照必须为新版本分配足够的存储。
- 只支持以下版本更改：
  - 从标准版更改为企业版
  - 从 Web 版更改为标准版或企业版
  - 从速成版更改为 Web 版、标准版或企业版

如果您希望从一个版本更改为另一个新版本，但还原快照不支持，您可以尝试使用本机备份和还原功能。SQL Server 根据数据库已启用的 SQL Server 功能，验证您的数据库是否与新版本兼容。有关更多信息，请参阅 使用本机备份和还原导入和导出 SQL Server 数据库。

MySQL 注意事项

要使用不支持的引擎版本从 RDS for MySQL 数据库快照进行还原，可能必须多次升级数据库快照。有关升级选项的更多信息，请参阅使用 RDS for MySQL 不支持的引擎版本的数据库快照的升级选项。

有关升级 RDS For MySQL 数据库快照的引擎版本的更多信息，请参阅升级 MySQL 数据库快照引擎版本。

Oracle Database 注意事项

从数据库快照还原 Oracle 数据库时，请注意以下事项：

- 还原数据库快照之前，您可以将其升级到更高版本的 Oracle 数据库。有关更多信息，请参阅 升级 Oracle 数据库快照。
- 如果还原使用单租户配置的 CDB 实例的快照，则可以更改 PDB 名称。当您的 CDB 实例使用多租户配置时，则无法更改 PDB 名称。有关更多信息，请参阅 备份和还原 CDB。
- 无法更改 CDB 名称，始终为 `RDSCDB`。对于所有 CDB 实例，此 CDB 名称是相同的。
- 您无法直接与数据库快照中的租户数据库进行交互。如果还原使用多租户配置的 CDB 实例的快照，则将还原其所有租户数据库。在还原数据库快照之前，您可以使用 describe-db-snapshot-tenant-databases 来检查数据库快照中的租户数据库。
- 如果您使用 Oracle GoldenGate，请始终使用 `compatible` 参数保留参数组。从数据库快照还原数据库实例时，请指定具有匹配或更大 `compatible` 值的参数组。
- 还原数据库快照时，您可以选择重命名数据库。如果联机重做日志的总大小超过 20GB，RDS 可能会将您的联机重做日志大小重置为其默认设置 512MB（4 x 128MB）。较小的大小允许还原操作在合理的时间内完成。您可以稍后重新创建联机重做日志并更改大小。
- 您可以在 Amazon Secrets Manager 中管理您的主用户密码。有关更多信息，请参阅 使用 Amazon Secrets Manager 管理主用户密码的概述。

将 Amazon RDS 的数据库实例还原到指定时间

可以将数据库实例还原到特定时间点，创建一个新的数据库实例而无需修改源数据库实例。

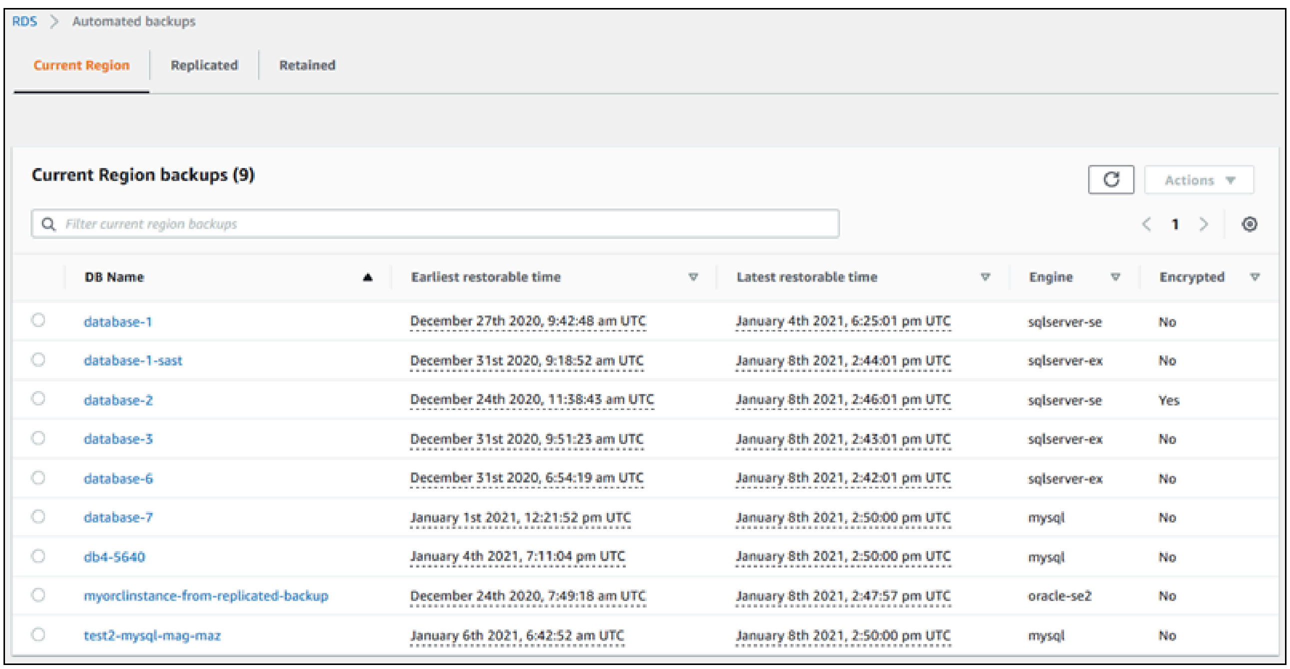
将数据库实例还原到某个时间点时，您可以选择默认的 Virtual Private Cloud (VPC) 安全组，也可以将自定义 VPC 安全组应用于数据库实例。

还原的数据库实例自动与默认数据库参数和选项组关联。但是，您可以通过在还原期间指定自定义参数组和选项组来应用它们。

如果源数据库实例具有资源标签，则 RDS 会向还原的数据库实例添加最新标签。

RDS 每隔 5 分钟将数据库实例的事务日志上传到 Amazon S3。要查看某个数据库实例的最近可还原时间，请使用 Amazon CLI [describe-db-instances](#) 命令，并查看该数据库实例的 `LatestRestorableTime` 字段中返回的值。要在 Amazon RDS 控制台中查看每个数据库实例的最近可还原时间，请选择自动备份。

您可以还原至备份保留期内的任何时间点。要查看每个数据库实例的最早可还原时间，请在 Amazon RDS 控制台中选择自动备份。



> **Note**
>
> 如果使用预置 IOPS 存储作为源数据库实例，我们建议您恢复到相同或相似的数据库实例大小和 IOPS。例如，如果您选择具有不兼容 IOPS 值的数据库实例大小，则可能会出现错误。

有关使用 RDS Extended Support 版本还原数据库实例的信息，请参阅使用 Amazon RDS 扩展支持还原数据库实例或多可用区集群。

从某个时间点进行还原时，某些 Amazon RDS 数据库引擎有一些特殊的注意事项：

- 如果您对 Amazon RDS for Db2 数据库实例使用密码身份验证，则日志中不会捕获用户管理操作（包括 `rdsadmin.add_user`）。这些操作需要完整快照备份。

  使用 BYOL 模型时，您的 RDS for Db2 数据库实例必须与包含您的 IBM Site ID 和 IBM Customer ID 的自定义参数组相关联。否则，尝试将数据库实例还原到特定时间点将失败。Amazon RDS for Db2 数据库实例还必须与 Amazon License Manager 自行管理许可证相关联。有关更多信息，请参阅 自带 Db2 许可。

  采用“通过 Amazon Web Services Marketplace 获得 Db2 许可证”模式时，您需要对要使用的特定 IBM Db2 版本进行有效 Amazon Web Services Marketplace 订阅。如果您还没有订阅，请在 Amazon Web Services Marketplace 中为该 IBM Db2 版本订阅 Db2。有关更多信息，请参阅 通过 Amazon Web Services Marketplace 获得 Db2 许可证。
- 将 RDS for Oracle 数据库实例还原到某个时间点时，可以为还原的数据库实例指定不同的数据库引擎、许可模式和 DBName（SID）。也可以指定 RDS 应在 Amazon Secrets Manager 中存储和管理主用户密码。有关更多信息，请参阅 使用 Amazon Secrets Manager 管理主用户密码的概述。
- 将 Microsoft SQL Server 数据库实例还原到某个时间点时，该实例中的每个数据库均还原到与实例中每个其他数据库相差 1 秒以内的时间点。对于实例内跨多个数据库的事务，还原时可能会发生不一致的情况。
- 对于 SQL Server 数据库实例，`OFFLINE`、`EMERGENCY` 和 `SINGLE_USER` 模式不受支持。将任何数据库设置成上述模式之一，会导致整个实例的最近可还原时间停滞不前。
- 某些操作 (如更改 SQL Server 数据库的恢复模式) 可中断用于时间点恢复的日志序列。在某些情况下，Amazon RDS 可以检测到此问题，最近的可恢复时间将无法向前推进。在其他情况下，例如在 SQL Server 数据库使用 `BULK_LOGGED` 恢复模型时，无法检测到日志序列中断。如果日志序列中断，则可能无法将 SQL Server 数据库实例还原到某个时间点。出于这些原因，Amazon RDS 不支持更改 SQL Server 数据库的恢复模式。

您也可以使用 Amazon Backup 管理 Amazon RDS 数据库实例的备份。如果您的数据库实例与 Amazon Backup 中的备份计划关联，则该备份计划用于时间点恢复。使用 Amazon Backup 创建的备份的名称以 `awsbackup:`*`Amazon-Backup-job-number`* 结尾。有关 Amazon Backup 的更多信息，请参阅 Amazon Backup 开发人员指南。

> **Note**
>
> 本主题中的信息适用于 Amazon RDS。有关还原 Amazon Aurora 数据库集群的更多信息，请参阅将数据库集群还原到指定时间。

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 将数据库实例还原到某个时间点。

> **Note**
>
> 还原数据库实例时，无法减少存储量。当您增加分配的存储空间时，必须至少增加 10%。如果您尝试将该值增加不到 10%，则会得到错误。您无法在还原 RDS for SQL Server 数据库实例时增加分配的存储。

控制台

将数据库实例还原至指定时间

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Automated backups（自动备份）。

   自动备份便会显示在 Current Region（当前区域）选项卡上。
3. 选择要还原的数据库实例。
4. 对于 Actions (操作)，选择 Restore to point in time (还原到时间点)。

   此时会显示还原到时间点窗口。
5. 选择最近可还原时间以还原到可能的最近时间，或选择自定义来选择时间。

   如果您选择 Custom（自定义），请输入要将实例集群还原到的日期和时间。

   > **Note**
   >
   > 时间以您的本地时区显示，表示为协调世界时 (UTC) 的偏移量。例如，UTC-5 是东部标准时间/中部夏令时。

6. 对于数据库实例标识符，请输入目标还原后的数据库实例的名称。名称必须唯一。
7. 根据需要选择其他选项，例如数据库实例类、存储，以及是否要使用存储自动扩展。

   有关每项设置的信息，请参阅 数据库实例的设置。
8. 选择还原到时间点。

Amazon CLI

要将数据库实例还原到指定的时间，请使用 Amazon CLI 命令 restore-db-instance-to-point-in-time 创建新的数据库实例。此示例还设置了分配的存储大小并启用存储自动扩缩。

此操作支持资源标记。使用 `--tags` 选项时，将忽略源数据库实例标签，并使用提供的标签。否则，将使用源实例的最新标签。

您可以指定其他设置。有关每项设置的信息，请参阅 数据库实例的设置。

Example

对于 Linux、macOS 或 Unix：

```
aws rds restore-db-instance-to-point-in-time \
    --source-db-instance-identifier mysourcedbinstance \
    --target-db-instance-identifier mytargetdbinstance \
    --restore-time 2017-10-14T23:45:00.000Z \
    --allocated-storage 100 \
    --max-allocated-storage 1000
```

对于 Windows：

```
aws rds restore-db-instance-to-point-in-time ^
    --source-db-instance-identifier mysourcedbinstance ^
    --target-db-instance-identifier mytargetdbinstance ^
    --restore-time 2017-10-14T23:45:00.000Z ^
    --allocated-storage 100 ^
    --max-allocated-storage 1000
```

RDS API

要将数据库实例还原到指定时间，请结合以下参数调用 Amazon RDS API `RestoreDBInstanceToPointInTime` 操作：

- `SourceDBInstanceIdentifier`
- `TargetDBInstanceIdentifier`
- `RestoreTime`

将多可用区数据库集群还原到指定时间

您可以将多可用区数据库集群还原到特定时间点，从而创建新的多可用区数据库集群。

RDS 会不断将多可用区数据库集群的事务日志上载到 Amazon S3。您可以还原至备份保留期内的任何时间点。要查看多可用区数据库集群的最早可还原时间，请使用 Amazon CLI [describe-db-clusters](https://docs.aws.amazon.com/cli/latest/reference/rds/describe-db-clusters.html) 命令。查看在数据库集群的 `EarliestRestorableTime` 字段中返回的值。要查看某个多可用区数据库集群的最晚可还原时间，请查看该数据库集群的 `LatestRestorableTime` 字段中返回的值。

在将多可用区数据库集群还原到某个时间点时，您可以选择该多可用区数据库集群的默认 VPC 安全组或将自定义 VPC 安全组应用到该多可用区数据库集群。

经过还原的多可用区数据库集群会自动与原定设置数据库集群参数组关联。但是，您可以通过在还原期间指定自定义数据库集群参数组来应用该参数组。

如果源数据库实例具有资源标签，RDS 会向已还原的数据库集群中添加最新标签。

> **Note**
>
> 我们建议您还原到与源数据库集群相同或相似的多可用区数据库集群大小。如果您使用 预置 IOPS 存储，我们还建议您使用相同或类似的 IOPS 值还原。例如，如果您选择具有不兼容 IOPS 值的数据库集群大小，则可能会出现错误。
> 如果源多可用区数据库集群使用通用型 SSD（gp3）存储并且分配的存储空间低于 400 GiB，则无法修改已还原数据库集群的预调配 IOPS。

有关使用 RDS 扩展支持版本还原多可用区数据库集群的信息，请参阅使用 Amazon RDS 扩展支持还原数据库实例或多可用区集群。

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 将多可用区数据库集群还原到某个时间点。

控制台

要将多可用区数据库集群还原到指定的时间点

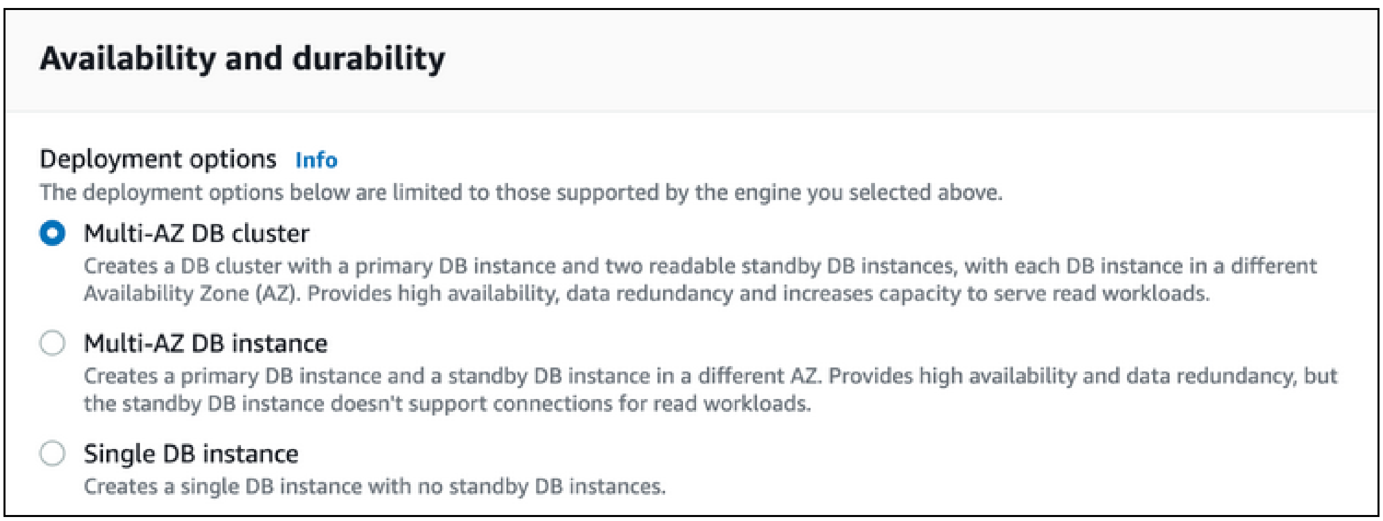
1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。

2. 在导航窗格中，选择 Databases（数据库）。
3. 选择要还原的多可用区数据库集群。
4. 对于 Actions (操作)，选择 Restore to point in time (还原到时间点)。

   此时会显示还原到时间点窗口。
5. 选择最近可还原时间以还原到可能的最近时间，或选择自定义来选择时间。

   如果您选择 Custom（自定义），请输入要将多可用区数据库集群还原到的日期和时间。

   > **Note**
   >
   > 时间以您的本地时区显示，表示为协调世界时 (UTC) 的偏移量。例如，UTC-5 是东部标准时间/中部夏令时。
6. 对于 DB cluster identifier（数据库集群标识符），请输入您还原的多可用区数据库集群的名称。
7. 在 Availability and durability（可用性与持久性）中，选择 Multi-AZ DB cluster（多可用区数据库集群）。

8. 在 DB instance class（数据库实例类）中，请选择数据库实例类。

   目前，多可用区数据库集群仅支持 db.m6gd 和 db.r6gd 数据库实例类。有关数据库实例类的更多信息，请参阅 数据库实例类。
9. 对于其余部分，请指定数据库集群设置。有关每项设置的信息，请参阅 创建多可用区数据库集群的设置。
10. 选择还原到时间点。

Amazon CLI

要将多可用区数据库集群还原到指定时间，请使用 Amazon CLI 命令 restore-db-cluster-to-point-in-time 以创建新的多可用区数据库集群。

目前，多可用区数据库集群仅支持 db.m6gd 和 db.r6gd 数据库实例类。有关数据库实例类的更多信息，请参阅 数据库实例类。

Example

对于 Linux、macOS 或 Unix：

```
aws rds restore-db-cluster-to-point-in-time \
    --source-db-cluster-identifier mysourcemultiazdbcluster \
    --db-cluster-identifier mytargetmultiazdbcluster \
    --restore-to-time 2021-08-14T23:45:00.000Z \
    --db-cluster-instance-class db.r6gd.xlarge
```

对于 Windows：

```
aws rds restore-db-cluster-to-point-in-time ^
    --source-db-cluster-identifier mysourcemultiazdbcluster ^
    --db-cluster-identifier mytargetmultiazdbcluster ^
    --restore-to-time 2021-08-14T23:45:00.000Z ^
    --db-cluster-instance-class db.r6gd.xlarge
```

RDS API

要将数据库集群还原到指定时间，请调用 Amazon RDS API RestoreDBClusterToPointInTime 操作，并使用以下参数：

- `SourceDBClusterIdentifier`
- `DBClusterIdentifier`
- `RestoreToTime`

从快照还原到多可用区数据库集群

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 将快照还原为多可用区数据库集群。您可以将以下每种类型的快照还原到多可用区数据库集群：

- 单可用区部署的快照
- 具有单个数据库实例的多可用区数据库集群部署的快照
- 多可用区数据库集群的快照

有关多可用区部署的信息，请参阅 配置和管理 Amazon RDS 的多可用区部署。

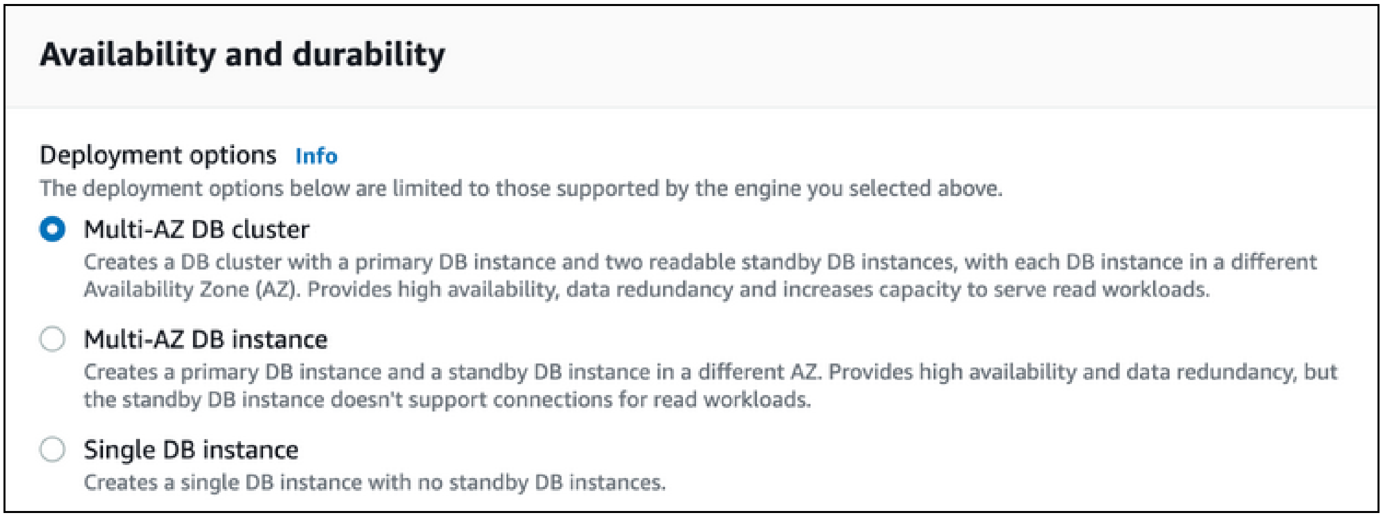
> **Tip**
>
> 您可以通过还原快照，将单可用区部署或多可用区数据库集群部署迁移到多可用区数据库集群部署。

有关使用 RDS 扩展支持版本还原多可用区数据库集群的信息，请参阅使用 Amazon RDS 扩展支持还原数据库实例或多可用区集群。

控制台

要将快照还原到多可用区数据库集群

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择快照。
3. 选择要从其还原的快照。
4. 对于操作，选择还原快照。
5. 在 Restore snapshot（还原快照）页面的 Availability and durability（可用性与持久性）下，选择 Multi-AZ DB cluster（多可用区数据库集群）。



6. 对于 DB cluster identifier（数据库集群标识符），请输入您还原的多可用区数据库集群的名称。
7. 对于其余部分，请指定数据库集群设置。有关每项设置的信息，请参阅 创建多可用区数据库集群的设置。
8. 选择还原数据库实例。

Amazon CLI

要从快照还原多可用区数据库集群，请使用 Amazon CLI 命令 restore-db-cluster-from-snapshot。

在下面的示例中，您将从先前创建的名为 `mysnapshot` 的快照中进行还原。您将还原为名为 `mynewmultiazdbcluster` 的新多可用区数据库集群。您还可以指定多可用区数据库集群中数据库实例使用的数据库实例类。为数据库引擎指定 `mysql` 或 `postgres`。

对于 `--snapshot-identifier` 选项，您可以使用名称或 Amazon Resource Name (ARN) 指定数据库集群快照。但是，您只能使用 ARN 指定数据库快照。

对于 `--db-cluster-instance-class` 选项，为新的多可用区数据库集群指定数据库实例类。多可用区数据库集群仅支持特定的数据库实例类，如 DB.m6gd 和 DB.r6gd 数据库实例类。有关数据库实例类的更多信息，请参阅 数据库实例类。

您也可以指定其他选项。

Example

对于 Linux、macOS 或 Unix：

```
aws rds restore-db-cluster-from-snapshot \
```

```
    --db-cluster-identifier mynewmultiazdbcluster \
    --snapshot-identifier mysnapshot \
    --engine mysql|postgres \
    --db-cluster-instance-class db.r6gd.xlarge
```

对于 Windows：

```
aws rds restore-db-cluster-from-snapshot ^
    --db-cluster-identifier mynewmultiazdbcluster ^
    --snapshot-identifier mysnapshot ^
    --engine mysql|postgres ^
    --db-cluster-instance-class db.r6gd.xlarge
```

还原数据库集群后，可以将多可用区数据库集群添加到与用于创建快照的数据库集群或数据库实例关联的安全组（如果适用）。完成此操作可提供与之前的数据库集群或数据库实例相同的功能。

RDS API

要从快照还原多可用区数据库集群，请调用 RDS API 操作 RestoreDBClusterFromSnapshot，并采用以下参数：

- `DBClusterIdentifier`
- `SnapshotIdentifier`
- `Engine`

您还可以选择指定以下参数。

还原数据库集群后，可以将多可用区数据库集群添加到与用于创建快照的数据库集群或数据库实例关联的安全组（如果适用）。完成此操作可提供与之前的数据库集群或数据库实例相同的功能。

从多可用区数据库集群快照还原到数据库实例

多可用区数据库集群快照是数据库集群的存储卷快照，并备份整个数据库集群而不仅仅是单个数据库。您可以将多可用区数据库集群快照还原到单可用区部署或多可用区数据库实例部署。有关多可用区部署的信息，请参阅 配置和管理 Amazon RDS 的多可用区部署。

> **Note**
>
> 您还可以将多可用区数据库集群快照还原到新的多可用区数据库集群。有关说明，请参阅 从快照还原到多可用区数据库集群。

有关使用 RDS 扩展支持版本还原多可用区数据库集群的信息，请参阅使用 Amazon RDS 扩展支持还原数据库实例或多可用区集群。

使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 将多可用区数据库集群快照还原到单可用区部署或多可用区数据库实例部署。

控制台

将多可用区数据库集群快照还原到单可用区部署或多可用区数据库实例部署

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择快照。
3. 选择要从中进行还原的多可用区数据库集群快照。
4. 对于操作，选择还原快照。
5. 在 Restore snapshot（还原快照）页面的 Availability and durability（可用性和持久性）中，选择以下选项之一：
   - Single DB instance（单个数据库实例）- 将快照还原到一个没有备用数据库实例的数据库实例。
   - Multi-AZ DB instance（多可用区数据库实例）– 将快照还原到具有一个主数据库实例和一个备用数据库实例的多可用区数据库实例部署。
6. 对于 DB instance identifier（数据库实例标识符），请输入还原后的数据库实例的名称。
7. 对于其余部分，请指定数据库实例设置。有关每项设置的信息，请参阅 数据库实例的设置。
8. 选择还原数据库实例。

Amazon CLI

要将多可用区数据库集群快照还原到数据库实例部署，请使用 Amazon CLI 命令 restore-db-instance-from-db-snapshot。

在下面的示例中，您将从先前创建的名为 myclustersnapshot 的多可用区数据库集群快照中进行还原。您可以使用名为 mynewdbinstance 的主数据库实例还原到新的多可用区数据库实例部署。对于 --db-cluster-snapshot-identifier 选项，请指定多可用区数据库集群快照的名称。

对于 --db-instance-class 选项，请为新的数据库实例部署指定数据库实例类。有关数据库实例类的更多信息，请参阅 数据库实例类。

您也可以指定其他选项。

Example

对于 Linux、macOS 或 Unix：

```
aws rds restore-db-instance-from-db-snapshot \
    --db-instance-identifier mynewdbinstance \
    --db-cluster-snapshot-identifier myclustersnapshot \
    --engine mysql \
    --multi-az \
    --db-instance-class db.r6g.xlarge
```

对于 Windows：

```
aws rds restore-db-instance-from-db-snapshot ^
    --db-instance-identifier mynewdbinstance ^
    --db-cluster-snapshot-identifier myclustersnapshot ^
    --engine mysql ^
    --multi-az ^
    --db-instance-class db.r6g.xlarge
```

还原数据库实例后，可以将其添加到与用于创建快照的多可用区数据库集群关联的安全组（如果适用）。完成此操作可提供与先前多可用区数据库集群相同的功能。

RDS API

要将多可用区数据库集群快照还原到数据库实例部署，请使用以下参数调用 RDS API 操作 RestoreDBInstanceFromDBSnapshot：

- `DBInstanceIdentifier`
- `DBClusterSnapshotIdentifier`
- `Engine`

您还可以选择指定以下参数。

还原数据库实例后，可以将其添加到与用于创建快照的多可用区数据库集群关联的安全组（如果适用）。完成此操作可提供与先前多可用区数据库集群相同的功能。

教程：从数据库快照还原 Amazon RDS 数据库实例

通常，在使用 Amazon RDS 时，您可能有一个偶尔使用而不是一直使用的数据库实例。例如，假设您有一个季度客户调查，该调查使用一个 Amazon EC2 实例来托管客户调查网站。您还有一个用于存储调查结果的数据库实例。在此类情况下节省资金的一种方法是在完成调查后创建数据库实例的数据库快照。然后，您可以删除数据库实例，并在需要再次进行调查时将其还原。

在还原数据库实例时，您需要提供要从中进行还原的数据库快照的名称。然后，为通过还原操作创建的新数据库实例提供名称。

有关从快照还原数据库实例的更多详细信息，请参阅 还原到数据库实例。

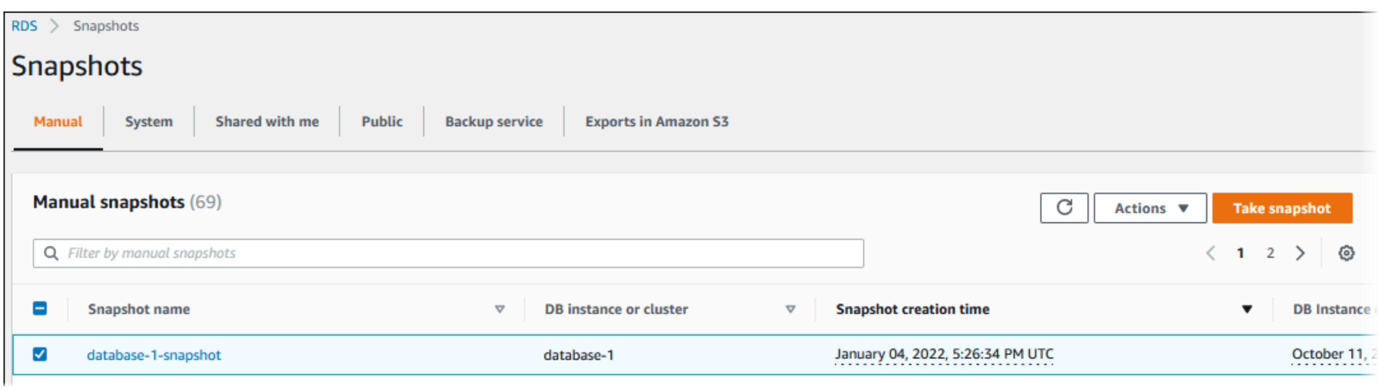
有关 Amazon RDS 的 Amazon KMS 密钥管理的信息，请参阅 Amazon KMS key 管理。

从数据库快照还原数据库实例

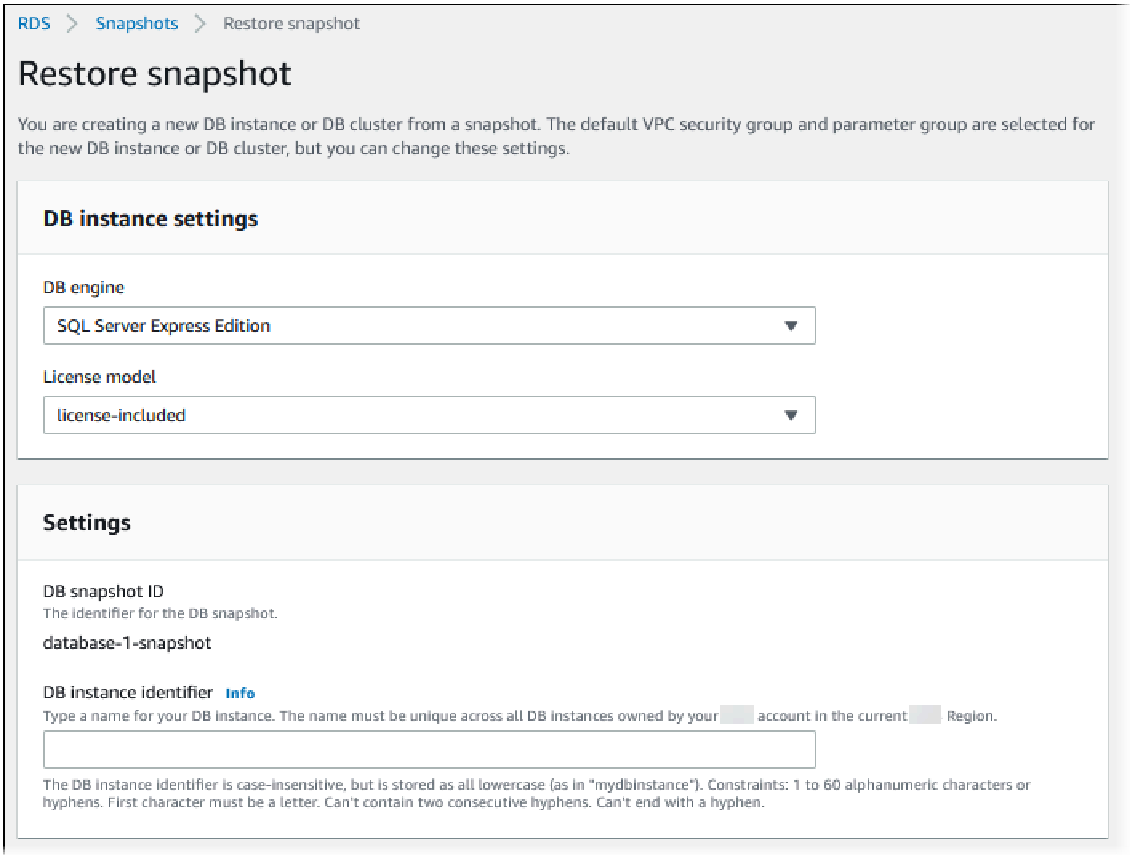
按照以下过程在 Amazon Web Services Management Console 中从快照还原。

要从数据库快照还原数据库实例，请执行以下操作：

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择快照。
3. 选择要从其还原的数据库快照。
4. 对于操作，选择还原快照。



此时会显示还原快照页面。



5. 在 DB instance settings（数据库实例设置）项下，使用 DB engine（数据库引擎）和 License model（许可模式）（适用于 Oracle 或 Microsoft SQL Server）的默认设置。
6. 在 Settings（设置）项下的 DB instance identifier（数据库实例标识符）中，输入要用于还原的数据库实例的唯一名称，如 **`mynewdbinstance`**。

   如果您要从在创建数据库快照之后删除的数据库实例还原，则可以使用该数据库实例的名称。
7. 在可用性与持久性下，选择是否在另一个可用区中创建备用实例。

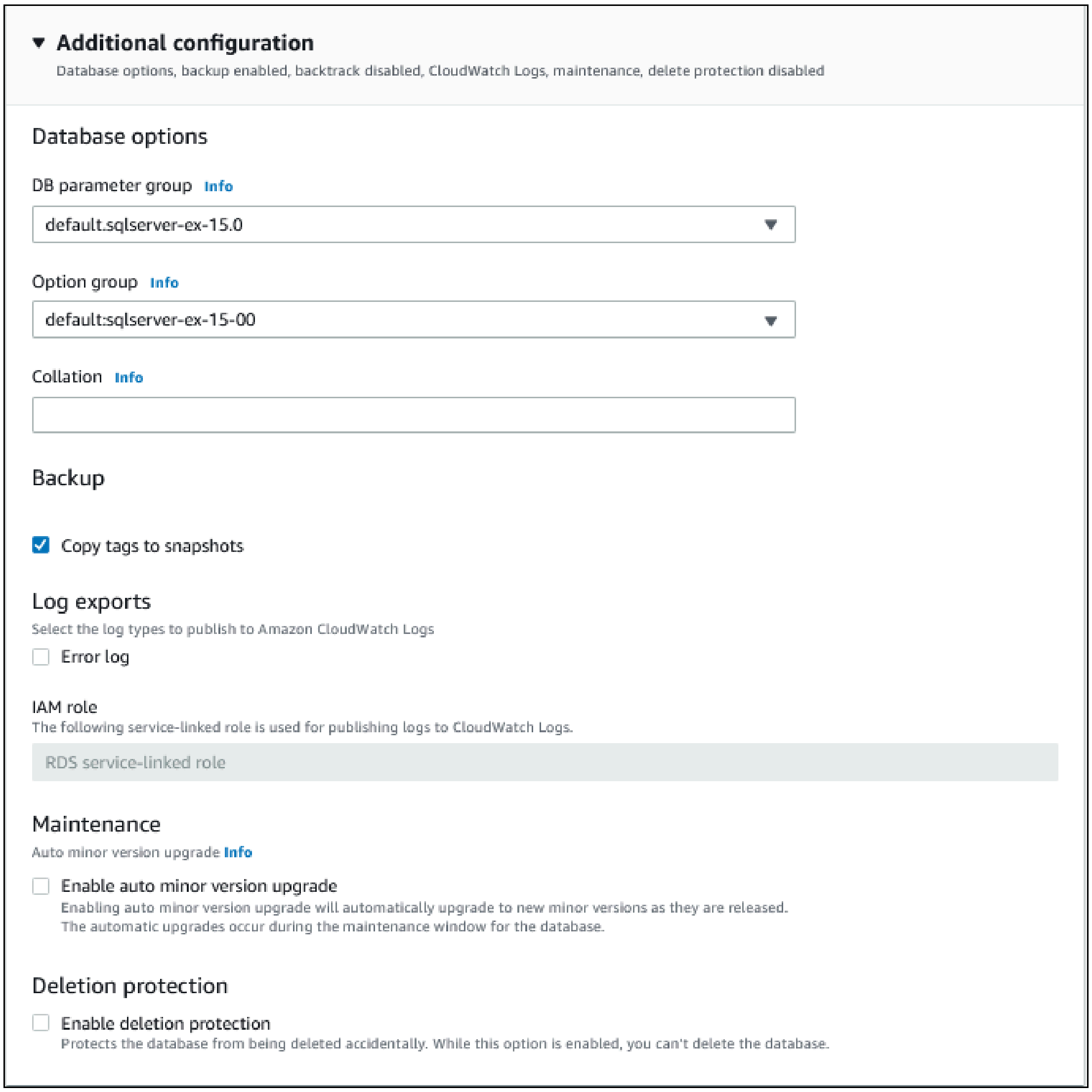
   在本教程中，不创建备用实例。
8. 在 Connectivity（连接）项下，使用以下各项的默认设置：
   - Virtual Private Cloud (VPC)
   - DB subnet group（数据库子网组）
   - 公有访问权限
   - VPC security group (firewall) [VPC 安全组（防火墙）]
9. 选择 DB instance class（数据库实例类）。

在本教程中，请选择 Burstable classes (includes t classes)（突增型类（包括 t 类））， 然后选择 db.t3.small。

10. 使用 Encryption（加密）项的默认设置。

    如果快照的源数据库实例已加密，则还原的数据库实例也会加密。您无法将其设置为未加密。

11. 展开页面底部的 Additional configuration（其他配置）。



12. 在 Database options（数据库选项）下执行以下操作：

    a. 选择 DB parameter group（数据库参数组）。

       在本教程中，请使用默认参数组。

    b. 选择 Option group（选项组）。

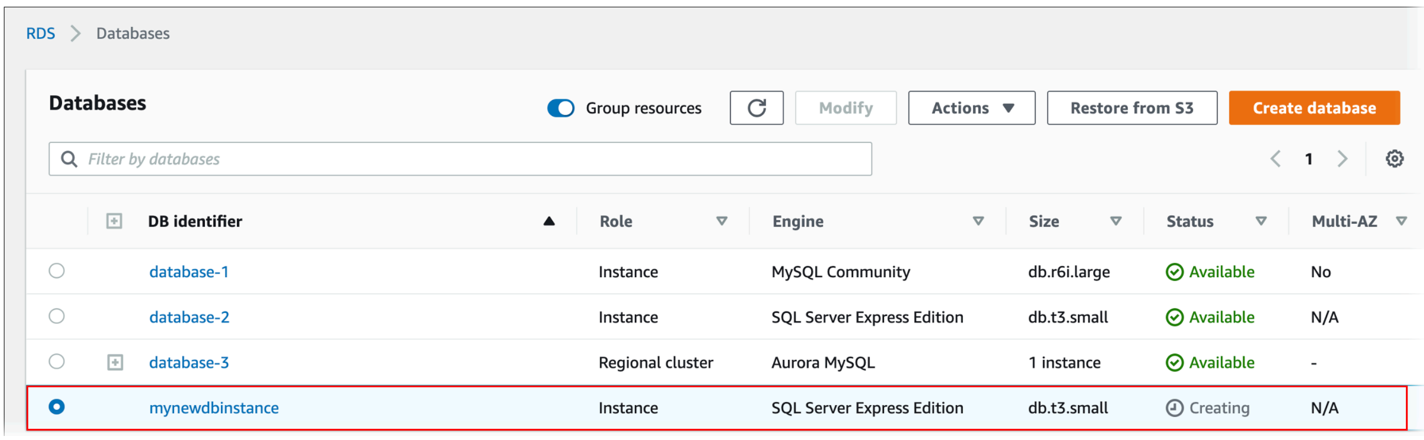
在本教程中，请使用默认选项组。

> ⚠ **Important**
>
> 在某些情况下，您可能会从使用持久或永久选项的数据库实例的数据库快照进行还原。如果是这样，请确保选择使用相同选项的选项组。

c. 为 Deletion protection（删除保护）选择 Enable deletion protection（启用删除保护）复选框。

13. 选择还原数据库实例。

Databases（数据库）页面会显示状态为 `Creating` 的还原数据库实例。



复制 Amazon RDS 的数据库快照

使用 Amazon RDS，您可以复制自动备份或手动数据库快照。在复制快照后，该副本为手动快照。可以创建自动备份或手动快照的多个副本，但每个副本必须具有唯一的标识符。

您可以复制位于同一 Amazon Web Services 区域中的快照，可以跨 Amazon Web Services 区域复制快照，也可以复制共享快照。

复制数据库快照

对于每个 Amazon Web Services 账户，一次最多可以从一个 Amazon Web Services 区域向另一个区域复制 20 个数据库快照。如果您将数据库快照复制到另一个 Amazon Web Services 区域，则可创建保留在该 Amazon Web Services 区域中的手动数据库快照。从源 Amazon Web Services 区域复制出数据库快照会产生 Amazon RDS 数据传输费用。

在新 Amazon Web Services 区域中创建数据库快照副本后，该数据库快照副本的行为与该 Amazon Web Services 区域中所有其他数据库快照的行为相同。

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 复制数据库快照。

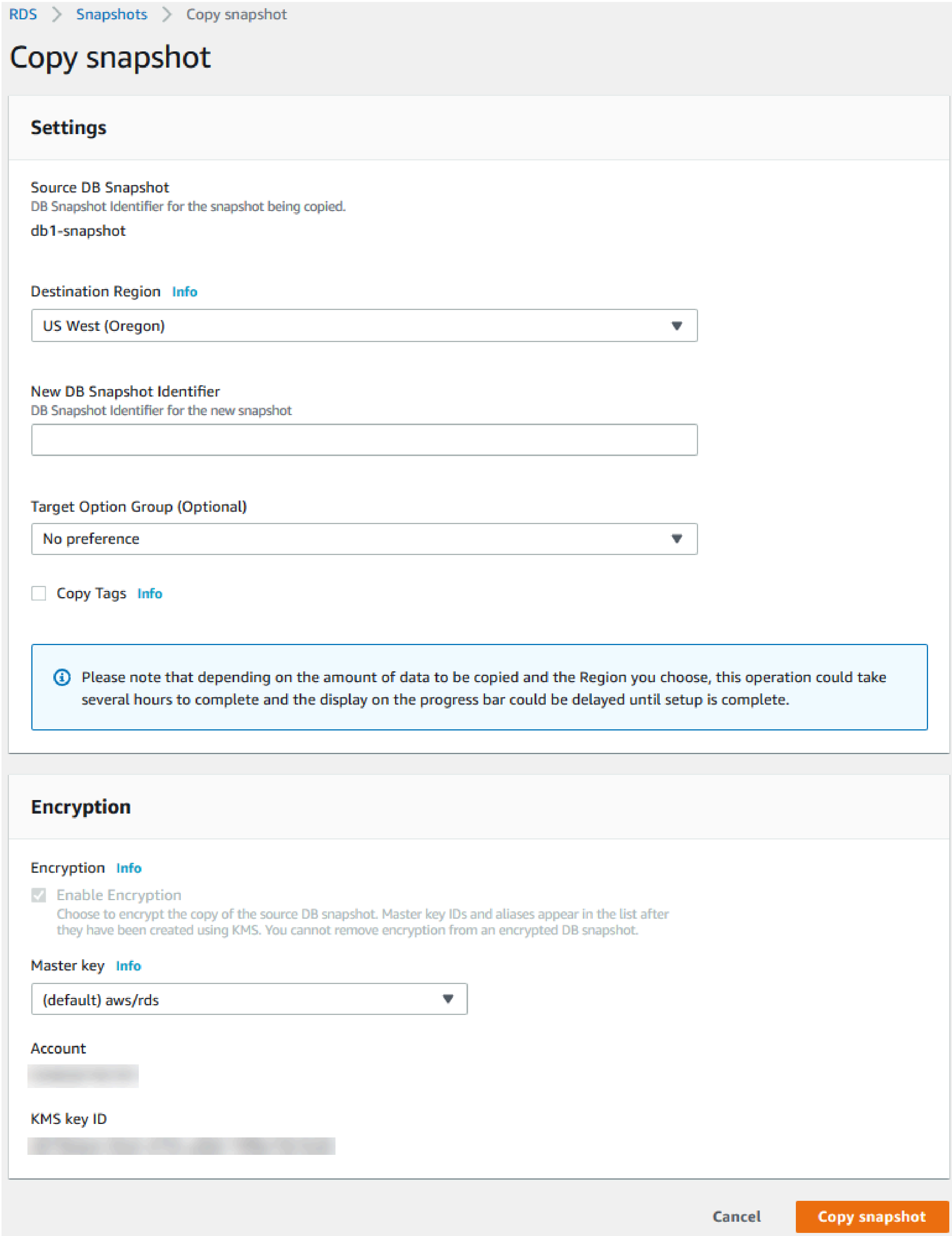
控制台

以下过程使用 Amazon Web Services Management Console，在同一 Amazon Web Services 区域中或跨区域复制加密或未加密数据库快照。

复制数据库快照

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择快照。
3. 选择要复制的数据库快照。
4. 对于 Actions (操作)，请选择 Copy snapshot (复制快照)。

   会显示 Copy snapshot (复制快照) 页面。



5. 对于 Target option group (optional) [目标选项组（可选）]，如果需要，则选择新的选项组。

如果将快照从一个 Amazon Web Services 区域复制到另一个区域，并且数据库实例使用非默认选项组，则指定该选项。

如果源数据库实例为 Oracle 或 Microsoft SQL Server 使用透明数据加密，则在跨区域复制时必须指定该选项。有关更多信息，请参阅 选项组的注意事项。

6. （可选）要将数据库快照复制到其他 Amazon Web Services 区域，请对 Destination Region（目标区域）选择新 Amazon Web Services 区域。

> **Note**
>
> 目标 Amazon Web Services 区域必须具有与源 Amazon Web Services 区域相同的可用数据库引擎版本。

7. 对于 New DB snapshot identifier（新数据库快照标识符），键入数据库快照副本的名称。

   可以创建自动备份或手动快照的多个副本，但每个副本必须具有唯一的标识符。
8. (可选) 选择 Copy Tags (复制标签) 将标签和值从快照复制到快照的副本。
9. （可选）对于加密，请执行以下操作：

   a. 如果数据库快照未加密，但需要加密副本，请选择 Enable Encryption (启用加密)。

   > **Note**
   >
   > 如果数据库快照已加密，则必须对副本进行加密，因此已选中复选框。

   b. 对于 Amazon KMS key，指定用于加密数据库快照副本的 KMS 密钥标识符。
10. 选择 Copy snapshot (复制快照)。

Amazon CLI

您可以使用 Amazon CLI 命令 copy-db-snapshot 复制数据库快照。如果您将快照复制到新 Amazon Web Services 区域，请在新 Amazon Web Services 区域中运行命令。

以下选项用于复制数据库快照。并非所有情况下都需要所有选项。使用以下说明和示例来确定要使用的选项。

- `--source-db-snapshot-identifier` – 源数据库快照的标识符。

  - 如果源快照与副本位于同一 Amazon Web Services 区域中，则指定有效的数据库快照标识符。例如，`rds:mysql-instance1-snapshot-20130805`。
  - 如果源快照与副本位于同一 Amazon Web Services 区域中，并且已与您的 Amazon Web Services 账户共享，则指定有效的数据库快照 ARN。例如，`arn:aws:rds:us-west-2:123456789012:snapshot:mysql-instance1-snapshot-20130805`。
  - 如果源快照与副本位于不同 Amazon Web Services 区域，则指定有效的数据库快照 ARN。例如：`arn:aws:rds:us-west-2:123456789012:snapshot:mysql-instance1-snapshot-20130805`。
  - 如果从共享的手动数据库快照进行复制，则该参数必须为共享的数据库快照的 Amazon Resource Name (ARN)。
  - 如果您复制加密快照，则此参数对于源 Amazon Web Services 区域必须为 ARN 格式，并且必须匹配 `PreSignedUrl` 参数中的 `SourceDBSnapshotIdentifier`。
- `--target-db-snapshot-identifier` – 加密数据库快照的新副本的标识符。
- `--copy-option-group` – 从已与您的 Amazon Web Services 账户 共享的快照中复制选项组。
- `--copy-tags` – 包括复制标签选项以将标签和值从快照复制到快照的副本。
- `--option-group-name` – 与快照的副本关联的选项组。

  如果将快照从一个 Amazon Web Services 区域复制到另一个区域，并且数据库实例使用非默认选项组，则指定该选项。

  如果源数据库实例为 Oracle 或 Microsoft SQL Server 使用透明数据加密，则在跨区域复制时必须指定该选项。有关更多信息，请参阅 选项组的注意事项。
- `--kms-key-id` – 加密的数据库快照的 KMS 密钥标识符。KMS 密钥标识符是 Amazon Resource Name (ARN)、密钥标识符或 KMS 密钥的密钥别名。
  - 如果您从 Amazon Web Services 账户复制加密的数据库快照，则可以为该参数指定值来使用新的 KMS 密钥加密副本。如果您不为该参数指定值，则使用与源数据库快照相同的 KMS 密钥来加密数据库快照的副本。
  - 如果您从其他 Amazon Web Services 账户复制共享的加密数据库快照，则必须为该参数指定值。
  - 如果您在复制未加密快照时指定该参数，则将加密副本。
  - 如果将加密快照复制到不同 Amazon Web Services 区域，必须为目标 Amazon Web Services 区域指定 KMS 密钥。KMS 密钥是特定于创建它们的 Amazon Web Services 区域，您无法将一个 Amazon Web Services 区域的加密密钥用于另一个 Amazon Web Services 区域。

Example 从未加密快照，到同一区域

以下代码使用新名称 mydbsnapshotcopy，在与源快照相同的 Amazon Web Services 区域中创建快照的副本。创建副本时，原始快照上的数据库选项组和标签都复制到快照副本。

对于 Linux、macOS 或 Unix：

```
aws rds copy-db-snapshot \
    --source-db-snapshot-identifier arn:aws:rds:us-west-2:123456789012:snapshot:mysql-
instance1-snapshot-20130805 \
    --target-db-snapshot-identifier mydbsnapshotcopy \
    --copy-option-group \
    --copy-tags
```

对于 Windows：

```
aws rds copy-db-snapshot ^
    --source-db-snapshot-identifier arn:aws:rds:us-west-2:123456789012:snapshot:mysql-
instance1-snapshot-20130805 ^
    --target-db-snapshot-identifier mydbsnapshotcopy ^
    --copy-option-group ^
    --copy-tags
```

Example 从未加密快照，跨区域

以下代码使用新名称 mydbsnapshotcopy，在运行命令的 Amazon Web Services 区域中创建快照的副本。

对于 Linux、macOS 或 Unix：

```
aws rds copy-db-snapshot \
    --source-db-snapshot-identifier arn:aws:rds:us-east-1:123456789012:snapshot:mysql-
instance1-snapshot-20130805 \
    --target-db-snapshot-identifier mydbsnapshotcopy
```

对于 Windows：

```
aws rds copy-db-snapshot ^
    --source-db-snapshot-identifier arn:aws:rds:us-east-1:123456789012:snapshot:mysql-
instance1-snapshot-20130805 ^
```

```
    --target-db-snapshot-identifier mydbsnapshotcopy
```

Example 从加密快照，跨区域

以下代码示例将加密的数据库快照从美国西部（俄勒冈州）区域复制到美国东部（弗吉尼亚州北部）。在目标 (us-east-1) 区域运行该命令。

对于 Linux、macOS 或 Unix：

```
aws rds copy-db-snapshot \
    --source-db-snapshot-identifier arn:aws:rds:us-west-2:123456789012:snapshot:mysql-instance1-snapshot-20161115 \
    --target-db-snapshot-identifier mydbsnapshotcopy \
    --kms-key-id my-us-east-1-key \
    --option-group-name custom-option-group-name
```

对于 Windows：

```
aws rds copy-db-snapshot ^
    --source-db-snapshot-identifier arn:aws:rds:us-west-2:123456789012:snapshot:mysql-instance1-snapshot-20161115 ^
    --target-db-snapshot-identifier mydbsnapshotcopy ^
    --kms-key-id my-us-east-1-key ^
    --option-group-name custom-option-group-name
```

RDS API

您可以使用 Amazon RDS API 操作 CopyDBSnapshot 复制数据库快照。如果您将快照复制到新 Amazon Web Services 区域，请在新 Amazon Web Services 区域中执行操作。

以下参数用于复制数据库快照。并非所有情况下都需要所有参数。使用以下说明和示例来确定要使用的参数。

- `SourceDBSnapshotIdentifier` – 源数据库快照的标识符。
  - 如果源快照与副本位于同一 Amazon Web Services 区域中，则指定有效的数据库快照标识符。例如，`rds:mysql-instance1-snapshot-20130805`。
  - 如果源快照与副本位于同一 Amazon Web Services 区域中，并且已与您的 Amazon Web Services 账户共享，则指定有效的数据库快照 ARN。例如，`arn:aws:rds:us-west-2:123456789012:snapshot:mysql-instance1-snapshot-20130805`。

  - 如果源快照与副本位于不同 Amazon Web Services 区域，则指定有效的数据库快照 ARN。例如：`arn:aws:rds:us-west-2:123456789012:snapshot:mysql-instance1-snapshot-20130805`。
  - 如果从共享的手动数据库快照进行复制，则该参数必须为共享的数据库快照的 Amazon Resource Name (ARN)。
  - 如果您复制加密快照，则此参数对于源 Amazon Web Services 区域必须为 ARN 格式，并且必须匹配 `PreSignedUrl` 参数中的 `SourceDBSnapshotIdentifier`。
- `TargetDBSnapshotIdentifier` – 加密数据库快照的新副本的标识符。
- `CopyOptionGroup` – 将该参数设置为 `true` 可将选项组从共享快照复制到快照的副本。默认为 `false`。
- `CopyTags` – 将该参数设置为 `true` 可将标签和值从快照复制到快照的副本。默认为 `false`。
- `OptionGroupName` – 与快照的副本关联的选项组。

  如果将快照从一个 Amazon Web Services 区域复制到另一个区域，并且数据库实例使用非默认选项组，则指定该参数。

  如果源数据库实例为 Oracle 或 Microsoft SQL Server 使用透明数据加密，则在跨区域复制时必须指定该参数。有关更多信息，请参阅 [选项组的注意事项]。
- `KmsKeyId` – 加密的数据库快照的 KMS 密钥标识符。KMS 密钥标识符是 Amazon Resource Name (ARN)、密钥标识符或 KMS 密钥的密钥别名。
  - 如果您从 Amazon Web Services 账户复制加密的数据库快照，则可以为该参数指定值来使用新的 KMS 密钥加密副本。如果您不为该参数指定值，则使用与源数据库快照相同的 KMS 密钥来加密数据库快照的副本。
  - 如果您从其他 Amazon Web Services 账户复制共享的加密数据库快照，则必须为该参数指定值。
  - 如果您在复制未加密快照时指定该参数，则将加密副本。
  - 如果将加密快照复制到不同 Amazon Web Services 区域，必须为目标 Amazon Web Services 区域指定 KMS 密钥。KMS 密钥特定于创建它们的 Amazon Web Services 区域，您无法将一个 Amazon Web Services 区域中的加密密钥用于另一个 Amazon Web Services 区域。
- `PreSignedUrl` – 在包含要复制的源数据库快照的源 Amazon Web Services 区域中，`CopyDBSnapshot` API 操作的包含签名版本 4 签名请求的 URL。

  使用 Amazon RDS API 从其他 Amazon Web Services 区域复制加密数据库快照时，请指定此参数。使用 Amazon CLI 从其他 Amazon Web Services 区域复制加密数据库快照时，必须指定源区域选项而非此参数。

预签名 URL 必须是对 CopyDBSnapshot API 操作的有效请求，该操作能够在包含要复制的加密数据库快照的源 Amazon Web Services 区域中运行。预签名 URL 请求必须包含以下参数值：

- `DestinationRegion` – 加密数据库快照将复制到的 Amazon Web Services 区域。此 Amazon Web Services 区域与调用包含此预签名 URL 的 `CopyDBSnapshot` 操作的区域为同一区域。

  例如，假设将加密的数据库快照从 us-west-2 区域复制到 us-east-1 区域。然后您在 us-east-1 区域中调用 `CopyDBSnapshot` 操作，并提供预签名 URL，该 URL 中包含在 us-west-2 区域中对 `CopyDBSnapshot` 操作的调用。对于本示例，预签名 URL 中的 `DestinationRegion` 必须设置为 us-east-1 区域。
- `KmsKeyId` – 要用于对目标 Amazon Web Services 区域中的数据库快照副本进行加密的密钥的 KMS 密钥标识符。这与在目标 `CopyDBSnapshot`中调用的 Amazon Web Services 区域 操作以及预签名 URL 中包含的操作所用的标识符相同。
- `SourceDBSnapshotIdentifier` – 复制的加密快照的数据库快照标识符。对于源 Amazon Web Services 区域，此标识符必须采用 Amazon 资源名称（ARN）格式。例如，如果您从 us-west-2 区域复制加密数据库快照，则 `SourceDBSnapshotIdentifier` 应类似于以下示例：`arn:aws:rds:us-west-2:123456789012:snapshot:mysql-instance1-snapshot-20161115`。

有关签名版本 4 签名请求的更多信息，请参阅：

- Amazon Simple Storage Service API 参考中的对请求进行身份验证：[使用查询参数（Amazon 签名版本 4）]
- Amazon Web Services 一般参考 中的[签名版本 4 签名流程]

Example 从未加密快照，到同一区域

以下代码使用新名称 `mydbsnapshotcopy`，在与源快照相同的 Amazon Web Services 区域中创建快照的副本。创建副本时，原始快照上的所有标签都将复制到快照副本。

```
https://rds.us-west-1.amazonaws.com/
 ?Action=CopyDBSnapshot
 &CopyTags=true
 &SignatureMethod=HmacSHA256
 &SignatureVersion=4
 &SourceDBSnapshotIdentifier=mysql-instance1-snapshot-20130805
 &TargetDBSnapshotIdentifier=mydbsnapshotcopy
 &Version=2013-09-09
 &X-Amz-Algorithm=AWS4-HMAC-SHA256
```

```
 &X-Amz-Credential=AKIADQKE4SARGYLE/20140429/us-west-1/rds/aws4_request
 &X-Amz-Date=20140429T175351Z
 &X-Amz-SignedHeaders=content-type;host;user-agent;x-amz-content-sha256;x-amz-date
 &X-Amz-Signature=9164337efa99caf850e874a1cb7ef62f3cea29d0b448b9e0e7c53b288ddffed2
```

Example 从未加密快照，跨区域

以下代码使用新名称 mydbsnapshotcopy，在 美国西部（加利福尼亚北部） 区域中创建快照的副本。

```
https://rds.us-west-1.amazonaws.com/
 ?Action=CopyDBSnapshot
 &SignatureMethod=HmacSHA256
 &SignatureVersion=4
 &SourceDBSnapshotIdentifier=arn%3Aaws%3Ards%3Aus-east-1%3A123456789012%3Asnapshot
%3Amysql-instance1-snapshot-20130805
 &TargetDBSnapshotIdentifier=mydbsnapshotcopy
 &Version=2013-09-09
 &X-Amz-Algorithm=AWS4-HMAC-SHA256
 &X-Amz-Credential=AKIADQKE4SARGYLE/20140429/us-west-1/rds/aws4_request
 &X-Amz-Date=20140429T175351Z
 &X-Amz-SignedHeaders=content-type;host;user-agent;x-amz-content-sha256;x-amz-date
 &X-Amz-Signature=9164337efa99caf850e874a1cb7ef62f3cea29d0b448b9e0e7c53b288ddffed2
```

Example 从加密快照，跨区域

以下代码使用新名称 mydbsnapshotcopy，在 US East (N. Virginia) 区域中创建快照的副本。

```
https://rds.us-east-1.amazonaws.com/
    ?Action=CopyDBSnapshot
    &KmsKeyId=my-us-east-1-key
    &OptionGroupName=custom-option-group-name
    &PreSignedUrl=https%253A%252F%252Frds.us-west-2.amazonaws.com%252F
         %253FAction%253DCopyDBSnapshot
         %2526DestinationRegion%253Dus-east-1
         %2526KmsKeyId%253Dmy-us-east-1-key
         %2526SourceDBSnapshotIdentifier%253Darn%25253Aaws%25253Ards%25253Aus-
west-2%25253A123456789012%25253Asnapshot%25253Amysql-instance1-snapshot-20161115
         %2526SignatureMethod%253DHmacSHA256
         %2526SignatureVersion%253D4
         %2526Version%253D2014-10-31
         %2526X-Amz-Algorithm%253DAWS4-HMAC-SHA256
```

```
        %2526X-Amz-Credential%253DAKIADQKE4SARGYLE%252F20161117%252Fus-west-2%252Frds
%252Faws4_request
        %2526X-Amz-Date%253D20161117T215409Z
        %2526X-Amz-Expires%253D3600
        %2526X-Amz-SignedHeaders%253Dcontent-type%253Bhost%253Buser-agent%253Bx-amz-
content-sha256%253Bx-amz-date
        %2526X-Amz-Signature
%253D255a0f17b4e717d3b67fad163c3ec26573b882c03a65523522cf890a67fca613
    &SignatureMethod=HmacSHA256
    &SignatureVersion=4
    &SourceDBSnapshotIdentifier=arn%3Aaws%3Ards%3Aus-west-2%3A123456789012%3Asnapshot
%3Amysql-instance1-snapshot-20161115
    &TargetDBSnapshotIdentifier=mydbsnapshotcopy
    &Version=2014-10-31
    &X-Amz-Algorithm=AWS4-HMAC-SHA256
    &X-Amz-Credential=AKIADQKE4SARGYLE/20161117/us-east-1/rds/aws4_request
    &X-Amz-Date=20161117T221704Z
    &X-Amz-SignedHeaders=content-type;host;user-agent;x-amz-content-sha256;x-amz-date
    &X-Amz-Signature=da4f2da66739d2e722c85fcfd225dc27bba7e2b8dbea8d8612434378e52adccf
```

限制

复制快照时，存在以下一些限制：

- 无法将快照复制到中国（北京）区域或 中国（宁夏）区域或从这两个区域复制快照。
- 您可以在 Amazon GovCloud（美国东部）和 Amazon GovCloud（美国西部）之间复制快照。但是，您不能在这些 GovCloud（美国）区域和非 GovCloud（美国）区域的区域之间复制快照。
- 如果您在目标快照可用之前删除了源快照，则快照复制会失败。在删除源快照之前，请确保目标快照的状态为 AVAILABLE。
- 每个账户最多可以同时进行到同一目标区域的 20 个快照复制请求。
- 当您为同一源数据库实例请求多个快照副本时，它们将在内部排队。在先前的快照副本完成之后，稍后请求的副本才会启动。有关更多信息，请参阅 Amazon 知识中心里的为什么我的 EC2 AMI 或 EBS 快照创建速度很慢？。
- 根据所涉及的 Amazon Web Services 区域和要复制的数据量，可能需要数小时才能完成跨区域快照复制。有时，某一给定的源区域可能会发出大量跨区域快照复制请求。在这种情况下，Amazon RDS 可能会将来自该源区域的新跨区域复制请求排入队列，直至某些区域正在进行的复制完成。当复制请求在队列中时，不显示有关这些复制请求的进度信息。复制开始后即显示进度信息。
- 如果某个副本在您启动另一个副本时仍处于挂起状态，则第二个副本在第一个副本完成后才开始。
- 您无法复制多可用区数据库集群的快照

- 您只能将使用 io2 卷的数据库实例的快照复制到有 io2 Block Express 卷可用的 Amazon Web Services 区域。有关更多信息，请参阅 Amazon RDS 数据库实例存储。

注意事项

有关复制数据库快照的注意事项，请参阅以下主题。

主题

- 快照保留
- 共享快照复制注意事项
- 加密快照复制的注意事项
- 增量快照复制的注意事项
- 跨区域快照复制的注意事项
- 选项组的注意事项
- 参数组的注意事项

快照保留

在以下几种情况下，Amazon RDS 会删除自动备份：

- 在保留期结束时。
- 对数据库实例禁用自动备份时。
- 删除数据库实例时。

如果要长期保留自动快照，则可复制它以创建一个手动快照。该快照在您删除之前将会一直保留。如果手动快照超出了默认存储空间，则可能会产生 Amazon RDS 存储成本。

共享快照复制注意事项

您可以复制其他 Amazon Web Services 账户与您共享的快照。在某些情况下，您可以从另一个 Amazon Web Services 账户复制共享的加密快照。在这些情况下，您必须有权访问用于加密快照的 Amazon KMS key。

> **Note**
>
> 对于您复制的共享快照，将向您收取 Amazon RDS 存储费用。Amazon RDS 可能会将源数据库实例的 ARN 附加到您复制的快照中。

您可以跨 Amazon Web Services 区域复制共享数据库快照，前提是该快照未加密。不过，如果共享数据库快照已加密，则您只能在同一区域中对其进行复制。

> **Note**
>
> 如果未加密或使用与初始完整快照相同的 KMS 密钥进行加密，则支持在同一 Amazon Web Services 区域中复制共享增量快照。如果您在复制后续快照时使用其他 KMS 密钥对其进行加密，则这些共享快照是完整快照。有关更多信息，请参阅 增量快照复制的注意事项。

加密快照复制的注意事项

您可以复制已使用 KMS 密钥加密的快照。如果您复制加密的快照，则此快照的副本也必须加密。如果在同一 Amazon Web Services 区域内复制加密的快照，可使用与原始快照相同的 KMS 密钥加密此副本。也可指定不同的 KMS 密钥。

如果跨区域复制加密快照，则必须指定在目标 Amazon Web Services 区域中有效的 KMS 密钥。该密钥可以是某个区域的专用 KMS 密钥，也可以是多区域密钥。有关多区域 KMS 密钥的更多信息，请参阅在 Amazon KMS 中使用多区域密钥。

源快照在复制过程中保持加密状态。有关更多信息，请参阅“Amazon RDS 加密的数据库实例的限制”。

您还可加密未加密快照的副本。这样，您可以快速为之前未加密的数据库实例增加加密功能。为此，您可在准备好加密数据库实例快照时创建该快照。然后创建该快照的副本并指定一个 KMS 密钥来加密该快照副本。之后您可从加密的快照还原加密的数据库实例。

有关 Amazon RDS 的 Amazon KMS 密钥管理的更多信息，请参阅 Amazon KMS key 管理。

增量快照复制的注意事项

增量快照仅包含在同一数据库实例的最近快照后发生更改的数据。与完整快照复制相比，增量快照复制速度更快，产生的存储成本更低。

快照副本是否为增量副本是由最近完成的快照副本和源快照决定的。如果删除了最新的快照副本，则下一个副本是完整副本，而不是增量副本。快照副本的类型将与源快照相同。如果源快照是增量快照，则快照副本将是增量快照。

在跨 Amazon Web Services 账户复制快照时，仅当满足以下所有条件时，副本才是增量副本：

- 最新的快照副本是同一个源数据库实例的副本，并且仍位于目标账户中。
- 目标账户中的所有快照副本均未加密，或者是使用同一 KMS 密钥加密的。
- 如果源数据库实例是多可用区实例，则自上次从该实例拍摄快照以来，它从未故障转移到其他可用区。

以下示例说明了完整快照和增量快照之间的区别。同样适用于共享快照和非共享快照。

快照	加密密钥	完整或增量
S1	K1	完整
S2	K1	S1 的增量
S3	K1	S2 的增量
S4	K1	S3 的增量
S1 (S1C) 的副本	K2	完整
S2 (S2C) 的副本	K3	完整
S3 (S3C) 的副本	K3	S2C 的增量
S4 (S4C) 的副本	K3	S3C 的增量
S4 (S4C2) 的副本 2	K4	完整

> **Note**
>
> 在这些示例中，仅当先前的快照仍然存在时，快照 S2、S3 和 S4 才是增量快照。
> 这同样适用于副本。仅当先前的副本仍然存在时，快照副本 S3C 和 S4C 才是增量副本。

有关跨 Amazon Web Services 区域复制增量快照的信息，请参阅完整和增量拷贝。

跨区域快照复制的注意事项

您可以跨 Amazon Web Services 区域复制数据库快照。但是，跨区域快照复制具有某些限制和注意事项。

请求跨区域数据库快照复制

要与源区域通信以请求跨区域数据库快照副复制，请求者（IAM 角色或 IAM 用户）必须有权访问源数据库快照和源区域。

请求者的 IAM 策略中的某些条件可能会导致请求失败。以下示例假定您要将数据库快照从 美国东部（俄亥俄州） 复制到 US East (N. Virginia)。这些示例显示了请求者的 IAM 策略中导致请求失败的条件：

- 请求者的策略对于 `aws:RequestedRegion` 具有条件限制。

  ```
  ...
  "Effect": "Allow",
  "Action": "rds:CopyDBSnapshot",
  "Resource": "*",
  "Condition": {
      "StringEquals": {
          "aws:RequestedRegion": "us-east-1"
      }
  }
  ```

  请求失败，因为该策略不允许访问源区域。要想请求成功，请指定源区域和目标区域。

  ```
  ...
  "Effect": "Allow",
  "Action": "rds:CopyDBSnapshot",
  "Resource": "*",
  "Condition": {
      "StringEquals": {
          "aws:RequestedRegion": [
              "us-east-1",
              "us-east-2"
          ]
      }
  }
  ```

- 请求者的策略不允许访问源数据库快照。

```
...
"Effect": "Allow",
"Action": "rds:CopyDBSnapshot",
"Resource": "arn:aws:rds:us-east-1:123456789012:snapshot:target-snapshot"
...
```

  要想请求成功，请指定源快照和目标快照。

```
...
"Effect": "Allow",
"Action": "rds:CopyDBSnapshot",
"Resource": [
    "arn:aws:rds:us-east-1:123456789012:snapshot:target-snapshot",
    "arn:aws:rds:us-east-2:123456789012:snapshot:source-snapshot"
]
...
```

- 请求者的政策拒绝 aws:ViaAWSService。

```
...
"Effect": "Allow",
"Action": "rds:CopyDBSnapshot",
"Resource": "*",
"Condition": {
    "Bool": {"aws:ViaAWSService": "false"}
}
```

  与源区域的通信由 RDS 代表请求者进行。要想请求成功，请不要拒绝 Amazon 服务发出的调用。

- 请求者的策略对于 aws:SourceVpc 或 aws:SourceVpce 具有条件限制。

  这些请求可能会失败，因为 RDS 对远程区域进行调用时，它并非来自指定的 VPC 或 VPC 端点。

如果您需要使用可能导致请求失败的上述某个条件，则可以在策略中包含第二个带有 aws:CalledVia 的语句，以便请求成功。例如，您可以使用带有 aws:CalledVia 的 aws:SourceVpce，如下所示：

```
...
"Effect": "Allow",
```

```
"Action": "rds:CopyDBSnapshot",
"Resource": "*",
"Condition": {
    "Condition" : {
        "ForAnyValue:StringEquals" : {
          "aws:SourceVpce": "vpce-1a2b3c4d"
        }
     }
},
{
    "Effect": "Allow",
    "Action": [
        "rds:CopyDBSnapshot"
    ],
    "Resource": "*",
    "Condition": {
        "ForAnyValue:StringEquals": {
            "aws:CalledVia": [
                "rds.amazonaws.com"
            ]
        }
    }
}
```

有关更多信息，请参阅 IAM 用户指南中的 IAM 中的策略和权限。

授权快照副本

跨区域数据库快照复制请求返回 `success` 后，RDS 将在后台启动副本。已创建 RDS 访问源快照的授权。此授权将源数据库快照链接到目标数据库快照，并允许 RDS 仅复制到指定的目标快照。

RDS 使用服务链接的 IAM 角色中的 `rds:CrossRegionCommunication` 权限对授权进行验证。如果副本获得授权，RDS 将与源区域通信并完成复制。

RDS 无权访问之前未经 `CopyDBSnapshot` 请求授权的数据库快照。复制完成后，将撤销授权。

RDS 使用服务链接的角色来验证源区域中的授权。如果在复制过程中删除了服务链接的角色，则复制将失败。

有关更多信息，请参阅 IAM 用户指南中的使用服务相关角色。

使用 Amazon Security Token Service 凭证

来自全局 Amazon Security Token Service (Amazon STS) 端点的会话令牌仅在默认启用的 Amazon Web Services 区域（商业区域）中有效。在 `assumeRole` 中使用来自 Amazon STS API 操作的凭证时， 如果源区域是选择加入区域，请使用区域端点。否则，请求将失败。发生这种情况的原因是，您的凭证必须在两个区域都有效，而在使用区域 Amazon STS 端点时，仅选择加入区域有效。

要使用全局端点，请确保在操作中为两个区域都启用了该端点。在 Amazon STS 账户设置中将全局端点设置为 `Valid in all Amazon Web Services ##`。

同样的规则也适用于预签名 URL 参数中的凭证。

有关更多信息，请参阅《IAM 用户指南》中的在 Amazon Web Services 区域中管理 Amazon STS。

延迟和多个复制请求

根据所涉及的 Amazon Web Services 区域和要复制的数据量，可能需要数小时才能完成跨区域快照复制。

有时，某一给定的源 Amazon Web Services 区域可能会发出大量跨区域快照复制请求。在这种情况下，Amazon RDS 可能会将来自该源 Amazon Web Services 区域的新跨区域复制请求排入队列，直至某些正在进行的复制完成。当复制请求在队列中时，不显示有关这些复制请求的进度信息。复制开始后即显示进度信息。

完整和增量拷贝

在向与源快照不同的 Amazon Web Services 区域复制快照时，即使复制增量快照，第一个副本也是完整快照副本。完整快照副本包含还原数据库实例需要的所有数据和元数据。在第一个快照副本后，可将同一数据库实例的增量快照复制到同一个 Amazon Web Services 账户内的相同目标区域。有关增量快照的更多信息，请参阅增量快照复制的注意事项。

已加密和未加密快照均支持跨 Amazon Web Services 区域的增量快照复制。

在跨 Amazon Web Services 区域复制快照时，如果满足以下条件，则副本为增量副本：

- 此前已将快照复制到目标区域。
- 最新的快照副本仍位于目标区域中。
- 目标区域中的所有快照副本均未加密，或者是使用同一 KMS 密钥加密的。

选项组的注意事项

数据库选项组特定于在其中创建它们的 Amazon Web Services 区域，您不能将一个 Amazon Web Services 区域中的选项组用于另一个 Amazon Web Services 区域。

对于 Oracle 数据库，您可以使用 Amazon CLI 或 RDS API 从与您的 Amazon Web Services 账户共享的快照中复制自定义数据库选项组。您只能在同一 Amazon Web Services 区域内复制选项组。如果选项组已复制到目标账户，并且在复制后未对其进行任何更改，则不会复制该选项组。如果之前已复制过源选项组，但在复制后发生了变化，则 RDS 会将新版本复制到目标账户。不会复制原定设置选项组。

在您跨区域复制快照时，可以为快照指定新选项组。我们建议您在复制快照之前准备新选项组。在目标 Amazon Web Services 区域，使用与原始数据库实例相同的设置创建选项组。如果新 Amazon Web Services 区域中已存在选项组，也可以使用它。

在某些情况下，您可以复制快照而不是为快照指定新的选项组。在这些情况下，还原快照时，数据库实例将获得默认选项组。要为新的数据库实例提供与原始数据库实例相同的选项，请执行以下操作：

1. 在目标 Amazon Web Services 区域，使用与原始数据库实例相同的设置创建选项组。如果新 Amazon Web Services 区域中已存在选项组，也可以使用它。
2. 在目标 Amazon Web Services 区域中还原快照之后，修改新数据库实例，并添加上一步中的新选项组或现有选项组。

参数组的注意事项

跨区域复制快照时，复制不包括由原始数据库实例使用的参数组。当您还原快照来创建新数据库实例时，该数据库实例会获得创建该实例的 Amazon Web Services 区域的默认参数组。要为新的数据库实例提供与源数据库实例相同的参数组，请执行以下操作：

1. 在目标 Amazon Web Services 区域中，使用与原始数据库实例相同的设置来创建数据库参数组。如果新 Amazon Web Services 区域中已存在选项组，也可以使用它。
2. 在目标 Amazon Web Services 区域中还原快照之后，修改新数据库实例，并添加新参数组或上一步中的现有参数组。

共享 Amazon RDS 的数据库快照

使用 Amazon RDS，您可以按以下方式共享手动数据库快照：

- 共享手动数据库快照（无论是否加密）可允许经授权的 Amazon Web Services 账户复制快照。
- 共享未加密的手动数据库快照可允许经授权的 Amazon Web Services 账户直接从快照还原数据库实例，而无需复制数据库实例再从中进行还原。但是，您无法从共享并且加密的数据库快照中恢复数据库实例。您可以改为创建数据库快照副本，并从该副本还原数据库实例。

> **Note**
>
> 要共享自动数据库快照，请通过复制自动快照来创建手动数据库快照，然后共享该副本。此过程也适用于 Amazon Backup 生成的资源。

有关复制快照的更多信息，请参阅复制 Amazon RDS 的数据库快照。有关从数据库快照还原数据库实例的更多信息，请参阅还原到数据库实例。

您可以与最多 20 个其他 Amazon Web Services 账户共享手动快照。

与其他 Amazon Web Services 账户共享手动快照时存在以下限制：

- 当使用 Amazon Command Line Interface (Amazon CLI) 或 Amazon RDS API 从共享的快照还原数据库实例时，您必须指定共享快照的 Amazon 资源名称（ARN）作为快照标识符。
- 如果数据库快照使用具有永久选项或持久选项的选项组，则不能共享，但具有 `Timezone` 和/或 `OLS` 选项的 Oracle 数据库实例除外。

  不能从选项组中移除永久选项。若为数据库实例指定了包含持续性选项的选项组，则不能从数据库实例中删除这些选项组。

  下表列出了永久和持续性选项及其相关的数据库引擎。

选项名称	持续的	永久的	数据库引擎
TDE	是	否	Microsoft SQL Server 企业版
TDE	是	是	Oracle Enterprise Edition
时区	是	是	

选项名称	持续的	永久的	数据库引擎
			Oracle Enterprise Edition Oracle Standard Edition Oracle Standard Edition One Oracle Standard Edition 2

对于 Oracle 数据库实例，您可以复制具有选项 `Timezone` 或 `OLS`（或两者）的共享数据库快照。为此，在复制数据库快照时，指定一个包含这些选项的目标选项组。OLS 选项仅对于运行 Oracle 12.2 或更高版本的 Oracle 数据库实例具有永久性和持久性。有关这些选项的更多信息，请参阅 Oracle 时区 和 Oracle Label Security。

- 您无法共享多可用区数据库集群的快照。

有关共享公有快照、共享加密快照和停止快照共享的信息，请参阅以下主题。

主题

- 共享 Amazon RDS 的公有快照
- 共享 Amazon RDS 的加密快照
- 停止 Amazon RDS 的快照共享

共享快照

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 共享数据库快照。

控制台

使用 Amazon RDS 控制台，您可以与最多 20 个 Amazon Web Services 账户共享手动数据库快照。您还可以使用该控制台停止与一个或多个账户共享手动快照。

使用 Amazon RDS 控制台共享手动数据库 快照

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。

2. 在导航窗格中，选择快照。
3. 选择要共享的手动快照。
4. 对于 Actions（操作），请选择 Share snapshot（共享快照）。
5. 为 DB snapshot visibility (数据库快照可见性) 选择以下一个选项。

   - 如果源未加密，选择公有以允许所有 Amazon 账户从您的手动数据库快照还原数据库实例，或选择私有以仅允许您指定的 Amazon Web Services 账户从手动数据库快照还原数据库实例。

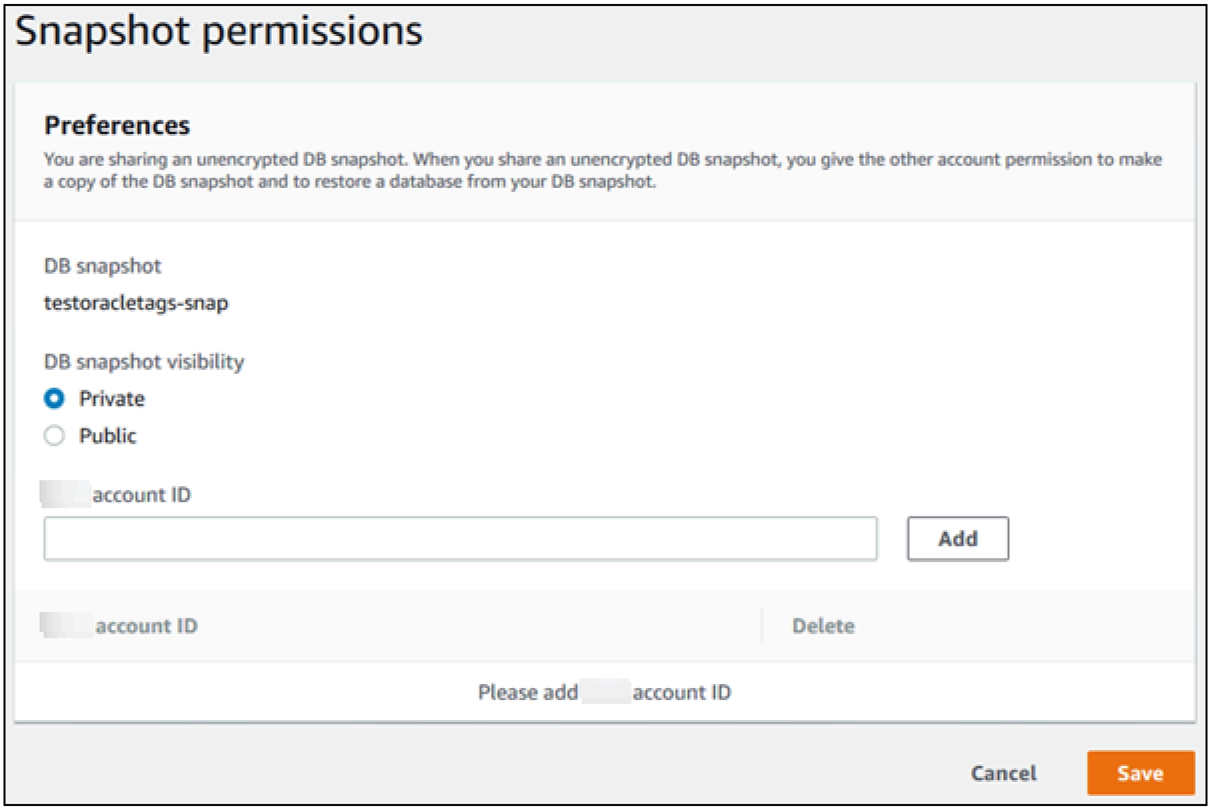
     > **Warning**
     >
     > 如果将数据库快照可见性设置为公有，则所有 Amazon Web Services 账户均可从您的手动数据库快照还原数据库实例，并且可访问您的数据。请勿将包含私有信息的任何手动数据库快照作为 Public 共享。
     > 有关更多信息，请参阅 [共享 Amazon RDS 的公有快照]。

   - 如果源已加密，由于已加密的快照无法公开共享，DB snapshot visibility (数据库快照可见性) 将设为 Private (私密)。

     > **Note**
     >
     > 无法共享已使用默认 Amazon KMS key 加密的快照。有关如何解决此问题的信息，请参阅[共享 Amazon RDS 的加密快照]。

6. 对于 Amazon 账户 ID，输入您想要允许从手动快照还原数据库实例的账户的 Amazon Web Services 账户标识符，然后选择添加。重复操作以加入其他 Amazon Web Services 账户标识符，最多可包含 20 个 Amazon Web Services 账户。

   如果您在许可账户列表中错加了某个 Amazon Web Services 账户标识符，可以选择错误 Amazon Web Services 账户标识符右侧的删除将其从列表中删除。



7. 为您想要允许还原手动快照的所有 Amazon Web Services 账户添加标识符以后，选择保存以保存您的更改。

Amazon CLI

要共享数据库快照，请使用 `aws rds modify-db-snapshot-attribute` 命令。使用 `--values-to-add` 参数添加有权还原手动快照的 Amazon Web Services 账户的 ID 列表。

Example 与单个账户共享快照

以下示例使 Amazon Web Services 账户标识符 `123456789012` 能够还原名为 `db7-snapshot` 的数据库快照。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-snapshot-attribute \
--db-snapshot-identifier db7-snapshot \
--attribute-name restore \
--values-to-add 123456789012
```

对于 Windows：

```
aws rds modify-db-snapshot-attribute ^
--db-snapshot-identifier db7-snapshot ^
--attribute-name restore ^
--values-to-add 123456789012
```

Example 与多个账户共享快照

以下示例使两个 Amazon Web Services 账户标识符（111122223333 和 444455556666）能够还原名为 manual-snapshot1 的数据库快照。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-snapshot-attribute \
--db-snapshot-identifier manual-snapshot1 \
--attribute-name restore \
--values-to-add {"111122223333","444455556666"}
```

对于 Windows：

```
aws rds modify-db-snapshot-attribute ^
--db-snapshot-identifier manual-snapshot1 ^
--attribute-name restore ^
--values-to-add "[\"111122223333\",\"444455556666\"]"
```

Note

使用 Windows 命令提示符时，必须在 JSON 代码中转义双引号 (")，方法是使用反斜杠 (\) 作为其前缀。

要列出能够用于还原快照的 Amazon Web Services 账户，请使用 describe-db-snapshot-attributes Amazon CLI 命令。

RDS API

您还可以使用 Amazon RDS API 与其他 Amazon Web Services 账户共享手动数据库快照。为此，请调用 ModifyDBSnapshotAttribute 操作。为 AttributeName 指定 restore，并使用 ValuesToAdd 参数添加有权还原手动快照的 Amazon Web Services 账户的 ID 列表。

要将手动快照设为公有并允许所有 Amazon Web Services 账户进行还原，请使用值 `all`。但请注意，若任何手动快照包含您不想向所有 Amazon Web Services 账户公开的私有信息，则不要添加 `all` 值。此外，由于此类快照不支持公开共享，请不要为已加密的快照指定 `all`。

要列出有权还原快照的所有 Amazon Web Services 账户，请使用 `DescribeDBSnapshotAttributes` API 操作。

共享 Amazon RDS 的公有快照

您可以将未加密的手动快照作为公有快照进行共享，这样所有 Amazon Web Services 账户均可使用此快照。当以公有快照形式共享快照时，确保不要将您的私有信息包含在公有快照之中。

公开共享快照时，它会授予所有 Amazon Web Services 账户复制快照并从中创建数据库实例的权限。

您不需要为其他账户拥有的公有快照的备份存储付费。您只需为您拥有的快照付费。

如果您复制公有快照，则您拥有该副本。您需要为快照副本的备份存储付费。如果您从公有快照创建数据库实例，则需要为该数据库实例付费。有关 Amazon RDS 定价信息，请参阅 [Amazon RDS 产品页]。

您只能删除您拥有的公有快照。要删除共享或公有快照，务必登录到拥有快照的 Amazon Web Services 账户。

查看其他 Amazon Web Services 账户拥有的公有快照

您可以在 Amazon RDS 控制台的 Snapshots (快照) 页面的 Public (公有) 选项卡上查看特定Amazon区域中其他账户拥有的公有快照。您的快照（由您账户拥有的快照）不会显示在此选项卡上。

要查看公有快照

1. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择快照。
3. 选择 Public (公有) 选项卡。

   此时将显示公有快照。您可以在 Owner (拥有者) 列中查看哪个账户拥有公有快照。

   > **Note**
   >
   > 您可能需要通过选择 Public snapshots (公有快照) 列表右上角的齿轮图标来修改页面首选项，才能看到此列。

查看您拥有的公有快照

您可以使用以下 Amazon CLI 命令（仅限 Unix）查看特定 Amazon 区域中由您的 Amazon Web Services 账户拥有的公有快照。

```
aws rds describe-db-snapshots --snapshot-type public --include-public |
 grep account_number
```

如果您有公共快照，则返回的输出与以下示例类似。

```
"DBSnapshotArn": "arn:aws:rds:us-east-1:123456789012:snapshot:mysnapshot1",
"DBSnapshotArn": "arn:aws:rds:us-east-1:123456789012:snapshot:mysnapshot2",
```

> Note
>
> 您可能会看到 `DBSnapshotIdentifier` 或 `SourceDBSnapshotIdentifier` 的重复条目。

共享来自已弃用数据库引擎版本的公有快照

不支持从已弃用的数据库引擎版本还原或复制公有快照。

RDS for Oracle 和 RDS for PostgreSQL 数据库引擎支持直接升级数据库快照引擎版本。您可以升级快照，然后公开重新共享它们。有关更多信息，请参阅下列内容：

- 升级 Oracle 数据库快照
- 升级 PostgreSQL 数据库快照引擎版本

对于其他数据库引擎，请执行以下步骤，使现有的不受支持的公有快照可供还原或复制：

1. 将快照标记为私有。
2. 还原快照。
3. 将还原的数据库实例升级到支持的引擎版本。
4. 创建新的快照。
5. 公开重新共享快照。

共享 Amazon RDS 的加密快照

您可共享使用 AES-256 加密算法“静态”加密的数据库快照，如加密 Amazon RDS 资源中所述。

以下限制适用于共享加密快照：

- 您无法公开共享加密的快照。
- 您无法共享使用透明数据加密 (TDE) 加密的 Oracle 或 Microsoft SQL Server 快照。
- 如果某个快照已使用共享该快照的 Amazon Web Services 账户的默认 KMS 密钥进行加密，则您无法共享该快照。

  有关 Amazon RDS 的 Amazon KMS 密钥管理的更多信息，请参阅 Amazon KMS key 管理。

要解决默认 KMS 密钥问题，请执行以下任务：

1. 创建客户自主管理型密钥并授予对它的访问权限.
2. 从源账户复制和共享快照.
3. 复制目标账户中的共享快照.

创建客户自主管理型密钥并授予对它的访问权限

首先，在与加密的数据库快照相同的 Amazon Web Services 区域中创建一个自定义 KMS 密钥。在创建客户自主管理型密钥时，您可以为另一个 Amazon Web Services 账户提供对它的访问权限。

> **Note**
> 当密钥策略授予对源账户和目标账户的访问权限时，您也可以使用其它 Amazon 账户中的 KMS 密钥。

创建客户自主管理型密钥并授予对它的访问权限

1. 从源 Amazon Web Services 账户登录 Amazon Web Services Management Console。
2. 从 https://console.aws.amazon.com/kms 打开 Amazon KMS 控制台。
3. 要更改 Amazon Web Services 区域，请使用页面右上角的区域选择器。
4. 在导航窗格中，选择客户托管密钥。
5. 选择创建密钥。

6. 在配置密钥页面上：

   a. 对于密钥类型，选择对称。
   b. 对于密钥用途，选择加密和解密。
   c. 展开 Advanced options (高级选项)。
   d. 对于密钥材料来源，选择 KMS。
   e. 对于区域性，请选择单区域密钥。
   f. 选择下一步。
7. 在添加标签页面上：

   a. 对于别名，输入您的 KMS 密钥的显示名称，例如 **`share-snapshot`**。
   b. （可选）为 KMS 密钥输入描述。
   c. （可选）向 KMS 密钥添加标签。
   d. 选择下一步。
8. 在定义密钥管理权限页面上，选择下一步。
9. 在定义密钥使用权限页面上：

   a. 对于其他 Amazon Web Services 账户，选择添加另一个 Amazon Web Services 账户。
   b. 输入您要向其授予访问权限的 Amazon Web Services 账户的 ID。

      您可以向多个 Amazon Web Services 账户授予访问权限。
   c. 选择下一步。
10. 查看您的 KMS 密钥，然后选择完成。

从源账户复制和共享快照

接下来，使用客户自主管理型密钥将源数据库快照复制到新快照。然后，您将与目标 Amazon Web Services 账户共享它。

复制和共享快照

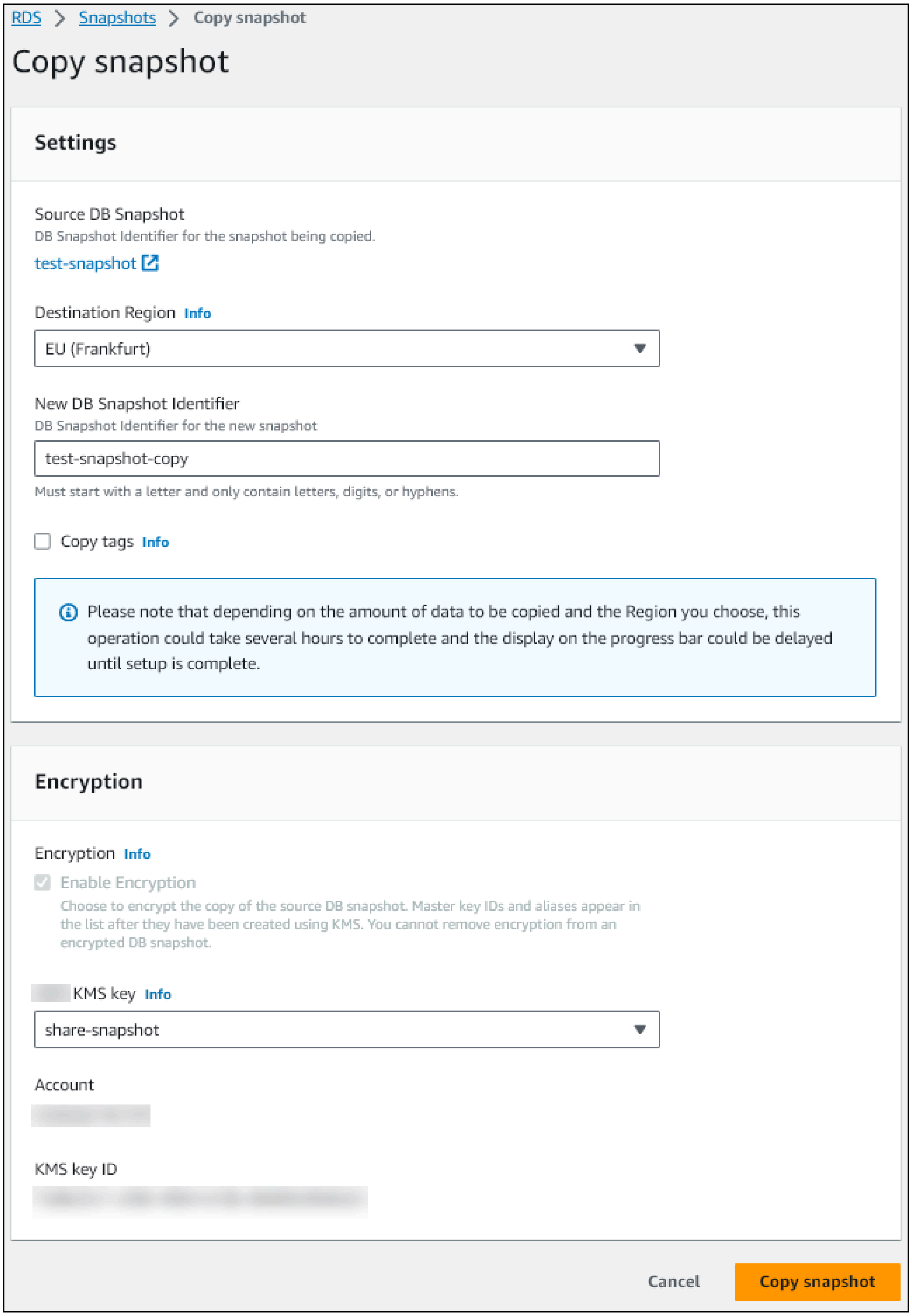
1. 从源 Amazon Web Services 账户登录 Amazon Web Services Management Console。
2. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/
3. 在导航窗格中，选择快照。
4. 选择要复制的数据库快照。

5. 对于 Actions (操作)，请选择 Copy snapshot (复制快照)。
6. 在复制快照页面上：

   a. 对于目标区域，选择您在上一个过程中创建客户自主管理型密钥的 Amazon Web Services 区域。

   b. 在新数据库快照标识符中输入数据库快照副本的名称。

   c. 对于 Amazon KMS key，选择您创建的客户自主管理型密钥。



d. 选择复制快照。

7. 当快照副本可用时，将其选中。

8. 对于 Actions（操作），请选择 Share snapshot（共享快照）。

9. 在快照权限页面上：

a. 输入您要与之共享快照副本的 Amazon Web Services 账户 ID，然后选择添加。

b. 选择保存。

共享此快照。

复制目标账户中的共享快照

现在，您可以复制目标 Amazon Web Services 账户中的共享快照。

复制共享快照

1. 从目标 Amazon Web Services 账户登录 Amazon Web Services Management Console。
2. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/
3. 在导航窗格中，选择快照。
4. 选择与我共享选项卡。
5. 选择共享快照。
6. 对于 Actions (操作)，请选择 Copy snapshot (复制快照)。
7. 按照前面的过程选择用于复制快照的设置，但要使用属于目标账户的 Amazon KMS key。

   选择复制快照。

停止 Amazon RDS 的快照共享

要停止共享数据库快照，请删除目标 Amazon Web Services 账户的权限。

控制台

停止与 Amazon Web Services 账户共享手动数据库快照

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择快照。
3. 选择要停止共享的手动快照。
4. 选择 Actions（操作），然后选择 Share snapshot（共享快照）。

5. 要移除 Amazon Web Services 账户的权限，请从授权账户列表中选择该账户的 Amazon 账户标识符所对应的删除。
6. 选择 保存 以保存您的更改。

CLI

要从列表中删除 Amazon Web Services 账户标识符，请使用 `--values-to-remove` 参数。

Example 停止共享快照

以下示例禁止 Amazon Web Services 账户 ID 444455556666 还原快照。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-snapshot-attribute \
--db-snapshot-identifier manual-snapshot1 \
--attribute-name restore \
--values-to-remove 444455556666
```

对于 Windows：

```
aws rds modify-db-snapshot-attribute ^
--db-snapshot-identifier manual-snapshot1 ^
--attribute-name restore ^
--values-to-remove 444455556666
```

RDS API

要移除 Amazon Web Services 账户的共享权限，请使用 `ModifyDBSnapshotAttribute` 操作（`AttributeName` 设置为 `restore`）和 `ValuesToRemove` 参数。要将手动快照标记为私有，请将值 `all` 从 `restore` 属性的值列表中删除。

将 Amazon RDS 的数据库快照数据导出到 Amazon S3

您可以将数据库快照数据导出到 Amazon S3 存储桶。导出过程在后台运行，而不会影响活动数据库的性能。

导出数据库快照时，Amazon RDS 从快照中提取数据并将其存储在 Amazon S3 存储桶中。数据以压缩和一致的 Apache Parquet 格式存储。

您可以导出所有类型的数据库快照 – 包括手动快照、自动系统快照和 Amazon Backup 服务创建的快照。默认情况下，将导出快照中的所有数据。但是，您可以选择导出特定的一组数据库、方案或表。

导出数据后，您可以通过 Amazon Athena 或 Amazon Redshift Spectrum 等工具直接分析导出的数据。有关使用 Athena 读取 Parque 数据的更多信息，请参阅 Amazon Athena 用户指南中的 Parquet SerDe。有关使用 Redshift Spectrum 读取 Parquet 数据的更多信息，请参阅《Amazon Redshift 数据库开发人员指南》中的从列式数据格式执行 COPY 操作。

> **Warning**
>
> 您无法将从 S3 导出的快照数据还原到新的数据库实例，也无法将快照数据从 S3 导入到现有的数据库实例。但是，您可以使用 Amazon Athena 或 Redshift Spectrum 处理数据来进行分析。此外，您可以使用 Amazon Glue 转换数据，然后使用诸如 Amazon DMS 之类的工具或自定义脚本将其导入 Amazon RDS。

有关将数据库快照导出到 Amazon S3 的更多信息，请参阅以下主题。

主题

导出快照数据概述

您可以使用以下过程将数据库快照数据导出到 Amazon S3 存储桶。有关更多详细信息，请参阅以下部分。

1. 确定要导出的快照。

   使用现有的自动快照或手动快照，或创建数据库实例或多可用区数据库集群的手动快照。

2. 设置对 Amazon S3 存储桶的访问权限。

   存储桶是 Amazon S3 对象或文件的容器。要提供访问存储桶的信息，请执行以下步骤：

   a. 标识要将快照导出到的 S3 存储桶。S3 存储桶必须与快照位于同一 Amazon 区域。有关更多信息，请参阅标识要导出到的 Amazon S3 存储桶。

   b. 创建一个 Amazon Identity and Access Management (IAM) 角色，用于授予快照导出任务对 S3 存储桶的访问权限。有关更多信息，请参阅使用 IAM 角色提供对 Amazon S3 存储桶的访问权限。

3. 创建对称加密 Amazon KMS key 以进行服务器端加密。快照导出任务使用 KMS 密钥在将导出数据写入 S3 时设置 Amazon KMS 服务器端加密。

   KMS 密钥策略必须同时包含 `kms:CreateGrant` 和 `kms:DescribeKey` 权限。有关在 Amazon RDS 中使用 KMS 密钥的更多信息，请参阅Amazon KMS key 管理。

   如果 KMS 密钥策略中有拒绝语句，则确保显式排除 Amazon 服务主体 `export.rds.amazonaws.com`。

   您可以在您的 Amazon 账户内使用 KMS 密钥，或者您可以使用跨账户 KMS 密钥。有关更多信息，请参阅使用跨账户 Amazon KMS key 加密 Amazon S3 导出的内容。

4. 使用控制台或 `start-export-task` CLI 命令将快照导出到 Amazon S3。有关更多信息，请参阅 将数据库快照导出到 Amazon S3 桶。

5. 要访问 Amazon S3 存储桶中导出的数据，请参阅 Amazon Simple Storage Service 用户指南中的上传、下载和管理对象。

设置 Amazon S3 存储桶的访问权限

要将数据库快照数据导出到 Amazon S3 文件，首先需要授予快照访问 Amazon S3 存储桶的权限。然后，您创建 IAM 角色以允许 Amazon RDS 服务写入 Amazon S3 存储桶。

主题

- 标识要导出到的 Amazon S3 存储桶
- 使用 IAM 角色提供对 Amazon S3 存储桶的访问权限
- 使用跨账户 Amazon S3 存储桶
- 使用跨账户 Amazon KMS key 加密 Amazon S3 导出的内容

标识要导出到的 Amazon S3 存储桶

标识要将数据库快照导出到的 Amazon S3 存储桶。使用现有 S3 存储桶或创建新的 S3 存储桶。

> **Note**
>
> 要导出到的 S3 存储桶必须与快照位于同一 Amazon 区域中。

有关使用 Amazon S3 存储桶的详细信息，请参阅 Amazon Simple Storage Service 用户指南中的以下主题：

- 如何查看 S3 存储桶的属性？
- 如何为 Amazon S3 存储桶启用默认加密？
- 如何创建 S3 存储桶？

使用 IAM 角色提供对 Amazon S3 存储桶的访问权限

将数据库快照数据导出到 Amazon S3 之前，请授予快照导出任务对 Amazon S3 存储桶的写入访问权限。

要授予此权限，请创建 IAM 策略以提供对桶的访问权限，然后创建一个 IAM 角色并将该策略附加到该角色。您稍后将此 IAM 角色分配给快照导出任务。

有关其他 Amazon S3 访问管理工具的信息，请参阅《Amazon S3 用户指南》中的 Amazon S3 中的访问控制。

> **Important**
>
> 如果计划使用Amazon Web Services Management Console导出快照，则可以选择在导出快照时自动创建 IAM 策略和角色。有关说明，请参阅 将数据库快照导出到 Amazon S3 桶。

授予数据库快照任务访问 Amazon S3 的权限

1. 创建一个 IAM 策略。此策略提供允许快照导出任务访问 Amazon S3 的存储桶和对象权限。

   在策略中包含以下必需操作，以允许将文件从 Amazon RDS 桶传输到 S3 桶：

   - `s3:PutObject*`
   - `s3:GetObject*`
   - `s3:ListBucket`
   - `s3:DeleteObject*`
   - `s3:GetBucketLocation`

   在策略中，包含以下资源以标识 S3 桶以及该桶中的对象。以下资源列表显示用于访问 Amazon S3 的 Amazon Resource Name (ARN) 格式。

   - `arn:aws:s3:::`*`amzn-s3-demo-bucket`*
   - `arn:aws:s3:::`*`amzn-s3-demo-bucket`*`/*`

   有关为 Amazon RDS 创建 IAM 策略的更多信息，请参阅创建和使用适用于 IAM 数据库访问的 IAM 策略。另请参阅 IAM 用户指南中的教程：创建和附加您的第一个客户托管策略。

   以下 Amazon CLI 命令使用这些选项创建一个名为 `ExportPolicy` 的 IAM 策略。该策略授予对名为 *`amzn-s3-demo-bucket`* 的存储桶的访问权限。

   > **Note**
   >
   > 创建策略后，请记下策略的 ARN。在将策略附加到 IAM 角色时，您在后面的步骤中需要使用 ARN。

   ```
   aws iam create-policy  --policy-name ExportPolicy --policy-document '{
       "Version": "2012-10-17",
       "Statement": [
           {
               "Sid": "ExportPolicy",
               "Effect": "Allow",
               "Action": [
                   "s3:PutObject*",
   ```

```
                "s3:ListBucket",
                "s3:GetObject*",
                "s3:DeleteObject*",
                "s3:GetBucketLocation"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        }
    ]
}'
```

2. 创建一个 IAM 角色，以便 Amazon RDS 可以代入该 IAM 角色，代表您访问 Amazon S3 桶。有关更多信息，请参阅 IAM 用户指南中的[创建向 IAM 用户委派权限的角色]。

   以下示例说明了如何使用 Amazon CLI 命令创建一个名为 `rds-s3-export-role` 的角色。

```
aws iam create-role  --role-name rds-s3-export-role  --assume-role-policy-document
 '{
     "Version": "2012-10-17",
     "Statement": [
       {
         "Effect": "Allow",
         "Principal": {
            "Service": "export.rds.amazonaws.com"
          },
         "Action": "sts:AssumeRole"
       }
     ]
   }'
```

3. 将您创建的 IAM 策略附加到您创建的 IAM 角色。

   以下 Amazon CLI 命令将之前创建的策略附加到名为 `rds-s3-export-role` 的角色。将 *`your-policy-arn`* 替换为您在先前步骤中记下的策略 ARN。

```
aws iam attach-role-policy  --policy-arn your-policy-arn  --role-name rds-s3-
export-role
```

使用跨账户 Amazon S3 存储桶

您可以跨 Amazon 账户使用 Amazon S3 存储桶。要使用跨账户存储桶，请添加存储桶策略以允许访问您用于 S3 导出的 IAM 角色。有关更多信息，请参阅示例 2：存储桶拥有者授予跨账户存储桶权限。

将存储桶策略附加到存储桶，如下面的示例所示。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::123456789012:role/Admin"
            },
            "Action": [
                "s3:PutObject*",
                "s3:ListBucket",
                "s3:GetObject*",
                "s3:DeleteObject*",
                "s3:GetBucketLocation"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-destination-bucket",
                "arn:aws:s3:::amzn-s3-demo-destination-bucket/*"
            ]
        }
    ]
}
```

使用跨账户 Amazon KMS key 加密 Amazon S3 导出的内容

您可以使用跨账户 Amazon KMS key 以加密 Amazon S3 导出的内容。首先，向本地账户添加密钥策略，然后在外部账户中添加 IAM 策略。有关更多信息，请参阅允许其他账户中的用户使用 KMS 密钥。

要使用跨账户 KMS 密钥

1. 向本地账户添加密钥策略。

   以下示例为外部账户 444455556666 中的 `ExampleRole` 和 `ExampleUser` 提供了内部账户 123456789012 中的权限。

```
{
    "Sid": "Allow an external account to use this KMS key",
    "Effect": "Allow",
    "Principal": {
        "AWS": [
            "arn:aws:iam::444455556666:role/ExampleRole",
            "arn:aws:iam::444455556666:user/ExampleUser"
        ]
    },
    "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:CreateGrant",
        "kms:DescribeKey",
        "kms:RetireGrant"
    ],
    "Resource": "*"
}
```

2. 在外部账户中添加 IAM 策略。

   以下示例 IAM 策略允许主体使用账户 123456789012 中的 KMS 密钥执行加密操作。要向账户 444455556666 中的 `ExampleRole` 和 `ExampleUser` 授予此权限，请将策略附加到该账户中的用户或角色。

```
{
    "Sid": "Allow use of KMS key in account 123456789012",
    "Effect": "Allow",
    "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:CreateGrant",
        "kms:DescribeKey",
        "kms:RetireGrant"
    ],
    "Resource": "arn:aws:kms:us-
west-2:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab"
```

```
}
```

将数据库快照导出到 Amazon S3 桶

每个 Amazon Web Services 账户 最多可以执行五个并发数据库快照导出任务。

> **Note**
>
> 导出 RDS 快照可能需要一段时间，具体取决于您的数据库类型和大小。导出任务首先还原并扩展整个数据库，然后再将数据提取到 Amazon S3。此阶段的任务进度显示为正在启动。当任务切换到将数据导出到 S3 时，进度显示为正在进行。
> 完成导出所需的时间取决于数据库中存储的数据。例如，具有分布良好的数字主键或索引列的表导出速度最快。不包含适用于分区的列的表，以及只有基于字符串的列上的一个索引的表将需要更长时间。导出时间之所以更长，是因为导出使用较慢的单线程进程。

您可以使用Amazon Web Services Management Console、Amazon CLI 或 RDS API 将数据库快照导出到 Amazon S3。要将数据库快照导出到跨账户 Amazon S3 存储桶，请使用 Amazon CLI 或 RDS API。

如果您使用 Lambda 函数导出快照，请将 `kms:DescribeKey` 操作添加到 Lambda 函数策略中。有关更多信息，请参阅 [Amazon Lambda 权限](#)。

控制台

仅为可导出到 Amazon S3 的快照显示导出到 Amazon S3 控制台选项。由于以下原因，快照可能无法导出：

- 此数据库引擎不支持 S3 导出。
- 此数据库引擎版本不支持 S3 导出。
- 创建了快照的 Amazon 区域不支持 S3 导出。

导出数据库快照

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择快照。

3. 从选项卡中，选择要导出的快照类型。
4. 在快照列表中，选择要导出的快照。
5. 对于 Actions (操作)，选择 Export to Amazon S3 (导出到 Amazon S3)。

   此时将显示 Export to Amazon S3 (导出到 Amazon S3) 窗口。
6. 对于 Export identifier (导出标识符)，输入用于标识导出任务的名称。此值也用于在 S3 存储桶中创建的文件的名称。
7. 选择要导出的数据：
   - 选择 All (全部) 可导出快照中的所有数据。
   - 选择 Partial (部分) 可导出快照的特定部分。如需标识要导出快照的哪些部分，请为 Identifiers (标识符)（以空格分隔）输入一个或多个数据库、架构或表。

     使用以下格式：

     ```
     database[.schema][.table] database2[.schema2][.table2] ... databasen[.scheman]
     [.tablen]
     ```

     例如：

     ```
     mydatabase mydatabase2.myschema1 mydatabase2.myschema2.mytable1
      mydatabase2.myschema2.mytable2
     ```
8. 对于 S3 bucket (S3 存储桶)，选择要导出到的存储桶。

   要将导出的数据分配给 S3 存储桶中的文件夹路径，请为 S3 prefix (S3 前缀) 输入可选路径。
9. 对于 IAM role (IAM 角色)，请选择一个角色以授予您对所选 S3 存储桶的写入访问权限，或创建新角色。
   - 如果您按照 使用 IAM 角色提供对 Amazon S3 存储桶的访问权限中的步骤创建了角色，请选择该角色。
   - 如果您没有创建授予您对所选 S3 桶的写入访问权限的角色，则选择 Create a new role（创建新角色）来自动创建该角色。接下来，在 IAM role name (IAM 角色名称) 中输入角色的名称。
10. 对于 Amazon KMS key，输入要用于加密导出数据的密钥的 ARN。
11. 选择 Export to Amazon S3 (导出到 Amazon S3)。

Amazon CLI

要使用 Amazon CLI 将数据库快照导出到 Amazon S3，请使用包含以下所需选项的 start-export-task 命令：

- `--export-task-identifier`
- `--source-arn`
- `--s3-bucket-name`
- `--iam-role-arn`
- `--kms-key-id`

在以下示例中，快照导出任务名为 *my-snapshot-export*，该任务将快照导出到名为 *amzn-s3-demo-bucket* 的 S3 存储桶。

Example

对于 Linux、macOS 或 Unix：

```
aws rds start-export-task \
    --export-task-identifier my-snapshot-export \
    --source-arn arn:aws:rds:Amazon_Region:123456789012:snapshot:snapshot-name \
    --s3-bucket-name amzn-s3-demo-bucket \
    --iam-role-arn iam-role \
    --kms-key-id my-key
```

对于 Windows：

```
aws rds start-export-task ^
    --export-task-identifier my-snapshot-export ^
    --source-arn arn:aws:rds:Amazon_Region:123456789012:snapshot:snapshot-name ^
    --s3-bucket-name amzn-s3-demo-bucket ^
    --iam-role-arn iam-role ^
    --kms-key-id my-key
```

示例输出如下。

```
{
    "Status": "STARTING",
```

```
    "IamRoleArn": "iam-role",
    "ExportTime": "2019-08-12T01:23:53.109Z",
    "S3Bucket": "my-export-bucket",
    "PercentProgress": 0,
    "KmsKeyId": "my-key",
    "ExportTaskIdentifier": "my-snapshot-export",
    "TotalExtractedDataInGB": 0,
    "TaskStartTime": "2019-11-13T19:46:00.173Z",
    "SourceArn": "arn:aws:rds:Amazon_Region:123456789012:snapshot:snapshot-name"
}
```

要在 S3 存储桶中为快照导出提供文件夹路径，请在 start-export-task 命令中包含 `--s3-prefix` 选项。

RDS API

要使用 Amazon RDS API 将数据库快照导出到 Amazon S3，请使用包含以下所需参数的 StartExportTask 操作：

- `ExportTaskIdentifier`
- `SourceArn`
- `S3BucketName`
- `IamRoleArn`
- `KmsKeyId`

区域和版本可用性

功能可用性和支持因每个数据库引擎的特定版本以及 Amazon Web Services 区域而异。有关将快照导出到 S3 的版本和区域可用性的更多信息，请参阅支持在 Amazon RDS 中将快照导出到 S3 的区域和数据库引擎。

限制

将数据库快照数据导出到 Amazon S3 有以下限制：

- 您不能同时为同一个数据库快照运行多个导出任务。这同时适用于完全导出和部分导出。
- 不支持从使用磁性存储的数据库导出快照。
- 导出到 S3 不支持包含冒号（:）的 S3 前缀。

- 在导出过程中，S3 文件路径中的以下字符将转换为下划线 (_)：

```
\ ` " (space)
```

- 如果数据库、架构或表的名称中包含以下字符以外的字符，则不支持部分导出。但是，您可以导出整个数据库快照。
  - 拉丁字母 (A–Z)
  - 数字 (0–9)
  - 美元符号 ($)
  - 下划线 (_)
- 数据库表列名不支持空格 ( ) 和某些字符。在导出过程中会跳过列名中包含以下字符的表：

```
, ; { } ( ) \n \t = (space)
```

- 在导出过程中会跳过其名称中包含斜杠 (/) 的表。
- 在导出期间，将跳过 RDS for PostgreSQL 临时表和未记录的表。
- 如果数据包含接近或大于 500MB 的大型对象（例如 BLOB 或 CLOB），则导出失败。
- 如果表中某个大行的大小接近或大于 2GB，则会在导出过程中略过该表。
- 对于部分导出，`ExportOnly` 列表的最大大小为 200 KB。
- 强烈建议您为每个导出任务使用唯一的名称。如果您没有使用唯一的任务名称，可能会收到以下错误消息：

  ExportTaskAlreadyExistsFault：调用 StartExportTask 操作时发生错误 (ExportTaskAlreadyExists)：ID 为 *xxxxx* 的导出任务已存在。
- 您可以在将快照数据导出到 S3 时删除快照，但是在导出任务完成之前，仍需支付该快照的存储成本。
- 您无法将从 S3 导出的快照数据还原到新的数据库实例，也无法将快照数据从 S3 导入到现有的数据库实例。
- 每个 Amazon Web Services 账户 最多可以执行五个并发数据库快照导出任务。
- 要将数据库快照导出到跨账户 Amazon S3 存储桶，必须使用 Amazon CLI 或 RDS API。
- 在 Amazon RDS 完成导出任务后，您可能需要稍等片刻才能从同一个数据库快照开始另一个导出任务。
- 您无法导出视图或实体化视图。
- RDS 导出到 S3 对于 GuardDuty Malware Protection for S3 不支持基于标签的访问控制。

监控 Amazon RDS 的快照导出

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 监控数据库快照导出。

控制台

监视数据库快照导出

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择快照。
3. 要查看快照导出的列表，请选择 Amazon S3 中的导出选项卡。
4. 要查看有关特定快照导出的信息，请选择导出任务。

Amazon CLI

要使用 Amazon CLI 监控数据库快照导出，请使用 describe-export-tasks 命令。

以下示例说明如何显示有关所有快照导出的当前信息。

Example

```
aws rds describe-export-tasks

{
    "ExportTasks": [
        {
            "Status": "CANCELED",
            "TaskEndTime": "2019-11-01T17:36:46.961Z",
            "S3Prefix": "something",
            "ExportTime": "2019-10-24T20:23:48.364Z",
            "S3Bucket": "amzn-s3-demo-bucket",
            "PercentProgress": 0,
            "KmsKeyId": "arn:aws:kms:Amazon_Region:123456789012:key/K7MDENG/
bPxRfiCYEXAMPLEKEY",
            "ExportTaskIdentifier": "anewtest",
            "IamRoleArn": "arn:aws:iam::123456789012:role/export-to-s3",
            "TotalExtractedDataInGB": 0,
            "TaskStartTime": "2019-10-25T19:10:58.885Z",
```

```
            "SourceArn": "arn:aws:rds:Amazon_Region:123456789012:snapshot:parameter-
groups-test"
        },
{
            "Status": "COMPLETE",
            "TaskEndTime": "2019-10-31T21:37:28.312Z",
            "WarningMessage": "{\"skippedTables\":[],\"skippedObjectives\":[],\"general
\":[{\"reason\":\"FAILED_TO_EXTRACT_TABLES_LIST_FOR_DATABASE\"}]}",
            "S3Prefix": "",
            "ExportTime": "2019-10-31T06:44:53.452Z",
            "S3Bucket": "amzn-s3-demo-bucket1",
            "PercentProgress": 100,
            "KmsKeyId": "arn:aws:kms:Amazon_Region:123456789012:key/2Zp9Utk/
h3yCo8nvbEXAMPLEKEY",
            "ExportTaskIdentifier": "thursday-events-test",
            "IamRoleArn": "arn:aws:iam::123456789012:role/export-to-s3",
            "TotalExtractedDataInGB": 263,
            "TaskStartTime": "2019-10-31T20:58:06.998Z",
            "SourceArn":
 "arn:aws:rds:Amazon_Region:123456789012:snapshot:rds:example-1-2019-10-31-06-44"
        },
        {
            "Status": "FAILED",
            "TaskEndTime": "2019-10-31T02:12:36.409Z",
            "FailureCause": "The S3 bucket edgcuc-export isn't located in the current
 Amazon Region. Please, review your S3 bucket name and retry the export.",
            "S3Prefix": "",
            "ExportTime": "2019-10-30T06:45:04.526Z",
            "S3Bucket": "amzn-s3-demo-bucket2",
            "PercentProgress": 0,
            "KmsKeyId": "arn:aws:kms:Amazon_Region:123456789012:key/2Zp9Utk/
h3yCo8nvbEXAMPLEKEY",
            "ExportTaskIdentifier": "wednesday-afternoon-test",
            "IamRoleArn": "arn:aws:iam::123456789012:role/export-to-s3",
            "TotalExtractedDataInGB": 0,
            "TaskStartTime": "2019-10-30T22:43:40.034Z",
            "SourceArn":
 "arn:aws:rds:Amazon_Region:123456789012:snapshot:rds:example-1-2019-10-30-06-45"
        }
    ]
}
```

要显示有关特定快照导出的信息，请在 `--export-task-identifier` 命令中包含 `describe-export-tasks` 选项。要筛选输出，请包括 `--Filters` 选项。有关更多选项，请参阅 describe-export-tasks 命令。

RDS API

要使用 Amazon RDS API 显示有关数据库快照导出的信息，请使用 DescribeExportTasks 操作。

要跟踪导出工作流的完成情况或启动其他工作流，您可以订阅 Amazon Simple Notification Service 主题。有关 Amazon SNS 的更多信息，请参阅 使用 Amazon RDS 事件通知。

取消 Amazon RDS 的快照导出任务

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 取消数据库快照导出任务。

> **Note**
>
> 取消快照导出任务不会删除导出到 Amazon S3 的任何数据。有关如何使用控制台删除数据的信息，请参阅如何从 S3 存储桶删除对象？ 要使用 CLI 删除数据，请使用 delete-object 命令。

控制台

取消快照导出任务

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择快照。
3. 选择 Exports in Amazon S3 (Amazon S3 中的导出) 选项卡。
4. 选择要取消的快照导出任务。
5. 选择 Cancel (取消)。
6. 在确认页面上选择 Cancel export task (取消导出任务)。

Amazon CLI

要使用 Amazon CLI 取消快照导出任务，请使用 cancel-export-task 命令。该命令需要 `--export-task-identifier` 选项。

Example

```
aws rds cancel-export-task --export-task-identifier my_export
{
    "Status": "CANCELING",
    "S3Prefix": "",
    "ExportTime": "2019-08-12T01:23:53.109Z",
    "S3Bucket": "amzn-s3-demo-bucket",
    "PercentProgress": 0,
    "KmsKeyId": "arn:aws:kms:Amazon_Region:123456789012:key/K7MDENG/
bPxRfiCYEXAMPLEKEY",
    "ExportTaskIdentifier": "my_export",
    "IamRoleArn": "arn:aws:iam::123456789012:role/export-to-s3",
    "TotalExtractedDataInGB": 0,
    "TaskStartTime": "2019-11-13T19:46:00.173Z",
    "SourceArn": "arn:aws:rds:Amazon_Region:123456789012:snapshot:export-example-1"
}
```

RDS API

要使用 Amazon RDS API 取消快照导出任务，请使用带 `ExportTaskIdentifier` 参数的 [CancelExportTask](https://docs.aws.amazon.com/AmazonRDS/latest/APIReference/API_CancelExportTask.html) 操作。

Amazon RDS 的 Amazon S3 导出任务的失败消息

下表描述了 Amazon S3 导出任务失败时返回的消息。

失败消息	描述
出现未知的内部错误。	由于未知错误、异常或故障导致任务失败。
将导出任务的元数据写入 S3 存储桶 [存储桶名称] 时出现未知的内部错误。	由于未知错误、异常或故障导致任务失败。
RDS 导出无法编写导出任务的元数据，因为它无法担任 IAM 角色 [角色 ARN]。	导出任务将担任您的 IAM 角色来验证是否允许向 S3 存储桶写入元数据。如果任务无法担任您的 IAM 角色，它将失败。
RDS 导出未能使用带有 KMS 密钥 [密钥 ID] 的 IAM 角色 [角色 ARN] 将导出任务	缺少一个或多个权限，因此导出任务无法访问 S3 存储桶。收到以下错误代码之一时，会引发此失败消息：

失败消息	描述
的元数据写入 S3 存储桶 [存储桶名称]。错误代码：[错误代码]	• 带有错误代码 `AccessDenied` 的 `AWSSecurityTokenServiceException` • 带有错误代码 `NoSuchBucket`、`AccessDenied`、`KMS.KMSInvalidStateException`、`403 Forbidden` 或 `KMS.DisabledException` 的 `AmazonS3Exception` 这些错误代码表示 IAM 角色、S3 桶或 KMS 密钥的设置出现配置错误。
IAM 角色 [角色 ARN] 无权在 S3 存储桶 [存储桶名称] 上调用 [S3 操作]。查看您的权限并重试导出。	IAM 策略配置错误。缺少对 S3 桶执行特定 S3 操作的权限，这会导致导出任务失败。
KMS 密钥检查失败。检查 KMS 密钥上的凭证然后重试。	KMS 密钥凭证检查失败。
S3 凭证检查失败。检查 S3 存储桶和 IAM 策略的权限。	S3 凭证检查失败。
S3 存储桶 [存储桶名称] 无效。它不在当前 Amazon 区域，或者它不存在。检查 S3 存储桶名称，然后重试导出。	S3 存储桶无效。
S3 存储桶 [存储桶名称] 不在当前 Amazon 区域。检查 S3 存储桶名称，然后重试导出。	S3 存储桶处于错误的 Amazon 区域。

排查 RDS for PostgreSQL 权限错误

将 PostgreSQL 数据库导出到 Amazon S3 时，您可能会看到 `PERMISSIONS_DO_NOT_EXIST` 错误，指出已跳过某些表。当您在创建数据库时指定的超级用户无权访问这些表时，通常会发生此错误。

要修复此错误，请运行以下命令：

```
GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA schema_name TO superuser_name
```

有关超级用户权限的更多信息，请参阅 主用户账户权限。

Amazon RDS 导出到 Amazon S3 时的文件命名约定

特定表的导出数据以 *base_prefix*/*files* 格式存储，基本前缀如下：

```
export_identifier/database_name/schema_name.table_name/
```

例如：

```
export-1234567890123-459/rdststdb/rdststdb.DataInsert_7ADB5D19965123A2/
```

文件的命名方式有两种约定。

- 当前约定：

  ```
  batch_index/part-partition_index-random_uuid.format-based_extension
  ```

  批量索引是一个序列号，表示从表中读取的一批数据。如果我们无法将您的表分区成小块以并行导出，则会有多个批量索引。如果表分区成多个表，也会发生同样的情况。这会出现多个批量索引，主表的每个表分区对应一个。

  如果我们可以将表分区成小块以并行读取，那么就只有批量索引 1 文件夹。

  在批量索引文件夹中，有一个或多个包含表数据的 Parquet 文件。Parquet 文件名的前缀是 `part-`*partition_index*。如果您的表已分区，则会有多个以分区索引 `00000` 开头的文件。

  分区索引序列中可能存在间隙。之所以发生这种情况，是因为每个分区都是从表中的范围查询中获得的。如果该分区的范围内没有数据，则跳过该序列号。

  例如，假设 `id` 列是表的主键，其最小值和最大值为 `100` 和 `1000`。当我们尝试导出这个带有九个分区的表时，我们会使用并行查询读取它，如下所示：

  ```
  SELECT * FROM table WHERE id <= 100 AND id < 200
  SELECT * FROM table WHERE id <= 200 AND id < 300
  ```

这应该生成九个文件，从 part-00000-*random_uuid*.gz.parquet 到 part-00008-*random_uuid*.gz.parquet。但是，如果没有 ID 介于 200 和 350 之间的行，则其中一个已完成的分区为空，并且不会为其创建任何文件。在前面的示例中，未创建 part-00001-*random_uuid*.gz.parquet。

- 较早的约定：

```
part-partition_index-random_uuid.format-based_extension
```

  这与当前约定相同，但没有 *batch_index* 前缀，例如：

```
part-00000-c5a881bb-58ff-4ee6-1111-b41ecff340a3-c000.gz.parquet
part-00001-d7a881cc-88cc-5ab7-2222-c41ecab340a4-c000.gz.parquet
part-00002-f5a991ab-59aa-7fa6-3333-d41eccd340a7-c000.gz.parquet
```

文件命名约定可能会更改。因此，在读取目标表时，我们建议您读取表的基本前缀内的所有内容。

Amazon RDS 导出到 Amazon S3 存储桶时的数据转换

将数据库快照导出到 Amazon S3 存储桶时，Amazon RDS 以 Parquet 格式转换数据、导出数据并存储数据。有关 Parquet 的更多信息，请参阅 Apache Parquet 网站。

Parquet 将所有数据存储为以下原始类型之一：

- BOOLEAN
- INT32
- INT64
- INT96
- FLOAT
- DOUBLE
- BYTE_ARRAY – 一个可变长度的字节数组，也称为二进制
- FIXED_LEN_BYTE_ARRAY – 当值具有恒定大小时使用的固定长度字节数组

Parquet 数据类型很少能减少读取和写入格式的复杂性。Parquet 提供了用于扩展原始类型的逻辑类型。逻辑类型实现为具有 `LogicalType` 元数据字段中数据的注释。逻辑类型注释说明如何解释原始类型。

当 STRING 逻辑类型注释 BYTE_ARRAY 类型时，它表示字节数组应被解释为 UTF-8 编码的字符串。导出任务完成后，如果发生了任何字符串转换，则 Amazon RDS 会通知您。导出的基础数据始终与源中的数据相同。但是，由于 UTF-8 中的编码差异，在工具（如 Athena）中读取时，某些字符可能会显示与源不同。

有关更多信息，请参阅 Parquet 文档中的 Parquet 逻辑类型定义。

主题

- MySQL 和 MariaDB 数据类型到 Parquet 的映射
- PostgreSQL 数据类型到 Parquet 的映射

MySQL 和 MariaDB 数据类型到 Parquet 的映射

下表显示在将数据转换并导出到 Amazon S3 时从 MySQL 和 MariaDB 数据类型到 Parquet 数据类型的映射。

源数据类型	Parquet 原始类型	逻辑类型注释	转换说明
数字数据类型			
BIGINT	INT64		
BIGINT UNSIGNED	FIXED_LEN _BYTE_ARRAY(9)	DECIMAL(20,0)	Parquet 仅支持签名类型，因此映射需要额外的字节（8 加 1）来存储 BIGINT_UN SIGNED 类型。
BIT	BYTE_ARRAY		
DECIMAL	INT32	DECIMAL (p,s)	如果源值小于 2^{31}，它其存储为 INT32。
	INT64	DECIMAL (p,s)	如果源值等于或大于 2^{31}，但小于 2^{63}，它将存储为 INT64。
	FIXED_LEN _BYTE_ARRAY(N)	DECIMAL (p,s)	如果源值等于或大于 2^{63}，则将它存

源数据类型	Parquet 原始类型	逻辑类型注释	转换说明
			储为 FIXED_LEN _BYTE_ARRAY(N)。
	BYTE_ARRAY	STRING	Parquet 不支持大于 38 的小数精度。十进制值转换为 BYTE_ARRAY 类型的字符串，并编码为 UTF8。
DOUBLE	DOUBLE		
FLOAT	DOUBLE		
INT	INT32		
INT UNSIGNED	INT64		
MEDIUMINT	INT32		
MEDIUMINT UNSIGNED	INT64		
NUMERIC	INT32	DECIMAL (p,s)	如果源值小于 2^{31}，它其存储为 INT32。
	INT64	DECIMAL (p,s)	如果源值等于或大于 2^{31}，但小于 2^{63}，它将存储为 INT64。
	FIXED_LEN _ARRAY(N)	DECIMAL (p,s)	如果源值等于或大于 2^{63}，则将它存储为 FIXED_LEN _BYTE_ARRAY(N)。

源数据类型	Parquet 原始类型	逻辑类型注释	转换说明
	BYTE_ARRAY	STRING	Parquet 不支持大于 38 的数值精度。此数值类型的值转换为 BYTE_ARRAY 类型的字符串，并编码为 UTF8。
SMALLINT	INT32		
SMALLINT UNSIGNED	INT32		
TINYINT	INT32		
TINYINT UNSIGNED	INT32	INT(16, true)	
字符串数据类型			
BINARY	BYTE_ARRAY		
BLOB	BYTE_ARRAY		
CHAR	BYTE_ARRAY		
ENUM	BYTE_ARRAY	STRING	
LINESTRING	BYTE_ARRAY		
LONGBLOB	BYTE_ARRAY		
LONGTEXT	BYTE_ARRAY	STRING	
MEDIUMBLOB	BYTE_ARRAY		
MEDIUMTEXT	BYTE_ARRAY	STRING	
MULTILINESTRING	BYTE_ARRAY		
SET	BYTE_ARRAY	STRING	

源数据类型	Parquet 原始类型	逻辑类型注释	转换说明
TEXT	BYTE_ARRAY	STRING	
TINYBLOB	BYTE_ARRAY		
TINYTEXT	BYTE_ARRAY	STRING	
VARBINARY	BYTE_ARRAY		
VARCHAR	BYTE_ARRAY	STRING	
日期和时间数据类型			
DATE	BYTE_ARRAY	STRING	日期将转换为 BYTE_ARRAY 类型的字符串，并编码为 UTF8。
DATETIME	INT64	TIMESTAMP_MICROS	
TIME	BYTE_ARRAY	STRING	TIME 类型转换为 BYTE_ARRAY 类型的字符串，并编码为 UTF8。
TIMESTAMP	INT64	TIMESTAMP_MICROS	
YEAR	INT32		
几何数据类型			
GEOMETRY	BYTE_ARRAY		
GEOMETRYCOLLECTION	BYTE_ARRAY		
MULTIPOINT	BYTE_ARRAY		

源数据类型	Parquet 原始类型	逻辑类型注释	转换说明
MULTIPOLYGON	BYTE_ARRAY		
POINT	BYTE_ARRAY		
POLYGON	BYTE_ARRAY		
JSON 数据类型			
JSON	BYTE_ARRAY	STRING	

PostgreSQL 数据类型到 Parquet 的映射

下表显示在将数据转换并导出到 Amazon S3 时从 PostgreSQL 数据类型到 Parquet 数据类型的映射。

PostgreSQL 数据类型	Parquet 原始类型	逻辑类型注释	映射注释
数字数据类型			
BIGINT	INT64		
BIGSERIAL	INT64		
DECIMAL	BYTE_ARRAY	STRING	DECIMAL 类型转换为 BYTE_ARRAY 类型的字符串，并编码为 UTF8。 此转换是为了避免由于数据精度和非数字 (NaN) 的数据值而引起的复杂性。
DOUBLE PRECISION	DOUBLE		
INTEGER	INT32		
MONEY	BYTE_ARRAY	STRING	

PostgreSQL 数据类型	Parquet 原始类型	逻辑类型注释	映射注释
REAL	FLOAT		
SERIAL	INT32		
SMALLINT	INT32	INT(16, true)	
SMALLSERIAL	INT32	INT(16, true)	
字符串和相关数据类型			
ARRAY	BYTE_ARRAY	STRING	数组转换为字符串并编码为 BINARY (UTF8)。 此转换是为了避免因数据精度、非数字 (NaN) 的数据值和时间数据值而产生的复杂性。
BIT	BYTE_ARRAY	STRING	
BIT VARYING	BYTE_ARRAY	STRING	
BYTEA	BINARY		
CHAR	BYTE_ARRAY	STRING	
CHAR(N)	BYTE_ARRAY	STRING	
ENUM	BYTE_ARRAY	STRING	
NAME	BYTE_ARRAY	STRING	
TEXT	BYTE_ARRAY	STRING	
文本搜索	BYTE_ARRAY	STRING	

PostgreSQL 数据类型	Parquet 原始类型	逻辑类型注释	映射注释
VARCHAR(N)	BYTE_ARRAY	STRING	
XML	BYTE_ARRAY	STRING	
日期和时间数据类型			
DATE	BYTE_ARRAY	STRING	
INTERVAL	BYTE_ARRAY	STRING	
TIME	BYTE_ARRAY	STRING	
带时区的时间	BYTE_ARRAY	STRING	
TIMESTAMP	BYTE_ARRAY	STRING	
TIMESTAMP（有时区）	BYTE_ARRAY	STRING	
几何数据类型			
BOX	BYTE_ARRAY	STRING	
CIRCLE	BYTE_ARRAY	STRING	
LINE	BYTE_ARRAY	STRING	
LINESEGMENT	BYTE_ARRAY	STRING	
路径	BYTE_ARRAY	STRING	
POINT	BYTE_ARRAY	STRING	
POLYGON	BYTE_ARRAY	STRING	
JSON 数据类型			
JSON	BYTE_ARRAY	STRING	
JSONB	BYTE_ARRAY	STRING	

PostgreSQL 数据类型	Parquet 原始类型	逻辑类型注释	映射注释
其他数据类型			
BOOLEAN	BOOLEAN		
CIDR	BYTE_ARRAY	STRING	网络数据类型
COMPOSITE	BYTE_ARRAY	STRING	
DOMAIN	BYTE_ARRAY	STRING	
INET	BYTE_ARRAY	STRING	网络数据类型
MACADDR	BYTE_ARRAY	STRING	
对象标识符	不适用		
PG_LSN	BYTE_ARRAY	STRING	
RANGE	BYTE_ARRAY	STRING	
UUID	BYTE_ARRAY	STRING	

使用 Amazon Backup 管理 Amazon RDS 的自动备份

Amazon Backup 是一项完全托管的备份服务，可在云中以及本地方便地集中管理和自动执行跨 Amazon 服务的数据备份。您可以在 Amazon Backup 中管理 Amazon RDS 数据库的备份。

> **Note**
>
> 由 Amazon Backup 管理的备份被视为手动数据库快照，但不计入 RDS 的数据库快照配额。使用 Amazon Backup 创建的备份的名称以 `awsbackup:`*`backup-job-number`* 结尾。

有关 Amazon Backup 的更多信息，请参阅 [Amazon Backup 开发人员指南](https://docs.aws.amazon.com/aws-backup/latest/devguide/)。

查看由 Amazon Backup 管理的备份

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择快照。
3. 选择 Backup service (备份服务) 选项卡。

   您的 Amazon Backup 备份位于 Backup service snapshots (备份服务快照) 之下。

监控 Amazon RDS 实例中的指标

在以下各节中，您可找到 Amazon RDS 监控概览以及关于如何访问指标的说明。要了解如何监控事件、日志和数据库活动流，请参阅 监控 Amazon RDS 数据库实例中的事件、日志和流。

主题

监控计划

在开始监控 Amazon RDS 之前，请创建监控计划。该计划应回答以下问题：

- 您的监控目标是什么？
- 您将监控哪些资源？
- 监控这些资源的频率如何？
- 您将使用哪些监控工具？
- 谁负责执行监控任务？
- 出现问题时应通知谁？

性能基准

为了实现您的监控目标，您需要建立一个基准。为此，请在 Amazon RDS 环境的不同时间、不同负载条件下测量性能。您可以监控如下指标：

- 网络吞吐量
- 客户端连接
- 用于读取、写入或元数据操作的 I/O
- 数据库实例的突增信用余额

我们建议您存储 Amazon RDS 的历史性能数据。使用存储的数据，您可以将当前性能与过去的趋势进行比较。您还可以区分正常性能模式与异常情况，并设计解决问题的方法。

性能准则

通常，性能指标的可接受值取决于应用程序相对于基准正在执行的操作。应调查相对于基准性能的一致或趋势性变化。以下指标通常是性能问题的根源：

- 高 CPU 或 RAM 消耗 – CPU 或 RAM 消耗值高可能是正常情况，前提是它们符合您的应用程序目标（如吞吐量或并发度）并且符合预期。
- 磁盘空间消耗 – 如果使用的空间始终不低于总磁盘空间的 85%，则应调查磁盘空间消耗。应查看是否可以从实例中删除数据或是将数据存档到其他系统以释放空间。
- 网络流量 – 对于网络流量，应与系统管理员进行讨论，以了解域网络和 Internet 连接的预期吞吐量。如果吞吐量始终低于预期，则应调查网络流量。
- 数据库连接 – 如果发现用户连接数较高，实例性能下降且响应时间延长，请考虑约束数据库连接。数据库实例的最佳用户连接数因您的实例类所执行操作的复杂性而异。要确定数据库连接的数量，请将数据库实例与参数组关联，其中 `User Connections` 参数设置为 0 以外的值（无限制）。您可以使用现有参数组或新建一个。有关更多信息，请参阅“Amazon RDS 的参数组”。
- IOPS 指标 – IOPS 指标的预期值取决于磁盘规格和服务器配置，因此，请使用您的基准来了解典型状况。调查值是否始终与您的基准不同。为获得最佳 IOPS 性能，请确保典型工作集与内存大小相适，以最大限度地减少读取和写入操作。

当性能超出已建立的基准时，您可能需要进行更改以优化数据库可用性，进而适应工作负载。例如，您可能需要更改数据库实例的实例类。或者，您可能需要更改可供客户端使用的数据库实例和只读副本的数量。

Amazon RDS 的监控工具

监控是保持 Amazon RDS 和您的其他 Amazon 解决方案的可靠性、可用性和性能的重要方面。Amazon 提供各种监控工具来监控 Amazon RDS、在出现错误时进行报告并适时自动采取措施。

主题

- 自动监控工具
- 手动监控工具

自动监控工具

建议您尽可能实现监控任务自动化。

主题

- Amazon RDS 实例状态和建议
- Amazon RDS 的 Amazon CloudWatch 指标
- Amazon RDS 性能详情和操作系统监控
- 集成服务

Amazon RDS 实例状态和建议

您可以使用以下自动化工具来监控 Amazon RDS，并在出现错误时进行报告：

- Amazon RDS 实例状态 – 通过使用 Amazon RDS 控制台、Amazon CLI 或 RDS API，查看有关您的实例当前状态的详细信息。
- Amazon RDS 建议 — 回应自动提供数据库资源（例如数据库实例、只读副本和数据库参数组）的建议。有关更多信息，请参阅 来自 Amazon RDS 的建议。

Amazon RDS 的 Amazon CloudWatch 指标

Amazon RDS 与 Amazon CloudWatch 集成以提供其他监控功能。

- Amazon CloudWatch – 此服务可实时监控您的Amazon资源以及您在Amazon上运行的应用程序。可以将以下 Amazon CloudWatch 功能用于 Amazon RDS：

- Amazon CloudWatch 指标 – Amazon RDS 每分钟自动向 CloudWatch 发送一次每个活动数据库的指标。对于 CloudWatch 中的 Amazon RDS 指标，您不会获得额外费用。有关更多信息，请参阅使用 Amazon CloudWatch 监控 Amazon RDS 指标。
- Amazon CloudWatch 警报 – 您可以在特定时间段内监控单个 Amazon RDS 指标。然后，您可以根据相对于您设置的阈值的指标值来执行一个或多个操作。有关更多信息，请参阅使用 Amazon CloudWatch 监控 Amazon RDS 指标。

Amazon RDS 性能详情和操作系统监控

您可以使用以下自动化工具来监控 Amazon RDS 性能：

- Amazon RDS Performance Insights - 可评估数据库上的负载，并确定执行操作的时间和位置。有关更多信息，请参阅 在 Amazon RDS 上使用性能详情监控数据库负载。
- Amazon RDS 增强监控 - 实时查看操作系统的指标。有关更多信息，请参阅 使用增强监控来监控操作系统指标。

集成服务

以下 Amazon 服务与 Amazon RDS 集成：

- Amazon EventBridge 是一种无服务器事件总线服务，可以轻松地将应用程序与来自各种来源的数据相连接。有关更多信息，请参阅 监控 Amazon RDS 事件。
- Amazon CloudWatch Logs 可让您监控、存储和访问来自 Amazon RDS 实例、CloudTrail 和其他来源的日志文件。有关更多信息，请参阅 监控 Amazon RDS 日志文件。
- Amazon CloudTrail 捕获由您的 Amazon Web Services 账户 或代表该账户发出的 API 调用和相关事件，并将日志文件传输到您指定的 Amazon S3 桶。有关更多信息，请参阅 监控 Amazon CloudTrail 中的 Amazon RDS API 调用。
- 数据库活动流是一项 Amazon RDS 功能，它提供 Oracle 数据库实例中近乎实时的活动流。有关更多信息，请参阅 使用数据库活动流监控 Amazon RDS。

手动监控工具

您需要手动监控那些 CloudWatch 警报未覆盖的项目。Amazon RDS、CloudWatch、Amazon Trusted Advisor 和其他Amazon控制台控制面板提供您的Amazon环境状态的概览视图。建议您还要查看数据库实例上的日志文件。

- 您可以从 Amazon RDS 控制台监控资源的以下项目：
  - 与数据库实例的连接数
  - 针对数据库实例的读写操作数量
  - 数据库实例当前使用的存储量
  - 为数据库实例使用的内存和 CPU 量
  - 流入和流出数据库实例的网络流量
- 您可以从 Trusted Advisor 控制面板查看以下成本优化、安全性、容错能力和性能改进检查：
  - Amazon RDS 闲置数据库实例
  - Amazon RDS 安全组访问风险
  - Amazon RDS 备份
  - Amazon RDS 多可用区

  有关这些检查的更多信息，请参阅 Trusted Advisor 最佳实践（检查）。
- CloudWatch 主页显示：
  - 当前告警和状态
  - 告警和资源图表
  - 服务运行状况

  此外，您还可以使用 CloudWatch 执行以下操作：
  - 创建自定义控制面板以监控您关注的服务。
  - 绘制指标数据图，以排除问题并弄清楚趋势。
  - 搜索并浏览您所有的 Amazon 资源指标。
  - 创建和编辑警报以接收有关问题的通知。

查看实例状态

使用 Amazon RDS 控制台，您可以快速访问数据库实例的状态。

主题

查看Amazon RDS 数据库实例状态

中的数据库实例的状态表示数据库实例的运行状况。您可以使用以下过程在 Amazon RDS 控制台、Amazon CLI 命令或 API 操作中查看数据库实例状态。

> **Note**
>
> Amazon RDS 还使用名为维护状态的另一种状态，此状态显示在 Amazon RDS 控制台的维护栏中。该值指示需要应用于数据库实例的任何维护修补程序的状态。维护状态独立于数据库实例状态。有关维护状态的更多信息，请参阅对数据库实例应用更新。

在下表中可找到数据库实例的可能状态值。此表还会显示是否对数据库实例和存储或者仅对存储向您计费，还是不向您计费。对于所有数据库实例状态，始终会针对备份使用向您计费。

数据库实例状态	已计费	描述
available	已计费	数据库实例正常且可用。
backing-up	已计费	当前正在备份数据库实例。
configuring-enhanced-monitoring	已计费	正在对此数据库实例启用或禁用增强监控。
configuring-iam-database-auth	已计费	正在对此数据库实例启用或禁用 Amazon Identity and Access Management（IAM）数据库身份验证。
configuring-log-exports	已计费	正在对此数据库实例启用或禁用向 Amazon CloudWatch Logs 发布日志文件。
converting-to-vpc	已计费	数据库实例正在从不在 Amazon Virtual Private Cloud (Amazon VPC) 中的数据库实例转换为在 Amazon VPC 中的数据库实例。
creating	不计费（非 PITR） 已计费（仅限 PITR）	正在创建数据库实例。无法访问正在创建的数据库实例。

数据库实例状态	已计费	描述
		如果您在时间点故障恢复（PITR）期间还原数据库，则会在数据库处于 creating 状态时对您计费。这是 creating 状态产生费用的唯一场景。
delete-precheck	不计费	Amazon RDS 正在验证只读副本是否正常运行且可以安全删除。
deleting	不计费	正在删除数据库实例。
failed	不计费	数据库实例已失败，Amazon RDS 无法恢复它。执行时间点还原，以还原至数据库实例的最近可还原时间，从而恢复数据。
inaccessible-encryption-credentials	不计费	无法访问或恢复用于加密或解密数据库实例的 Amazon KMS key。
inaccessible-encryption-credentials-recoverable	对存储计费	无法访问用于加密或解密数据库实例的 KMS 密钥。但是，如果 KMS 密钥处于活动状态，则重新启动数据库实例可以恢复它。 有关更多信息，请参阅 加密数据库实例。
incompatible-create	不计费	Amazon RDS 正尝试创建数据库实例，但无法执行此操作，因为资源与您的数据库实例不兼容。这种情况在一定条件下会发生，例如，当数据库实例的实例配置文件没有正确的权限时。
incompatible-network	不计费	Amazon RDS 正尝试对数据库实例执行恢复操作，但无法执行此操作，因为 VPC 正处于一种阻止此操作完成的状态。例如，如果子网中的所有可用 IP 地址都在使用中，并且 Amazon RDS 无法为数据库实例获取 IP 地址，就会出现此状态。

数据库实例状态	已计费	描述
incompatible-option-group	已计费	Amazon RDS 尝试应用选项组更改，但却无法执行，并且 Amazon RDS 无法回滚到选项组之前的状态。有关更多信息，请查看数据库实例的近期事件列表。例如，如果选项组包含一个诸如 TDE 的选项以及数据库实例不包含加密信息时，上述情况可能会发生。
incompatible-parameters	已计费	Amazon RDS 无法启动数据库实例，因为在数据库实例的数据库参数组中指定的参数与数据库实例不兼容。恢复参数更改或使这些更改与数据库实例相兼容以重新访问数据库实例。有关不兼容参数的更多信息，请查看数据库实例的近期事件列表。
incompatible-restore	不计费	Amazon RDS 无法执行时间点还原。此状态的常见原因包括使用临时表、 使用带 MySQL 的 MyISAM 表，或使用带 MariaDB 的 Aria 表。
insufficient-capacity	不计费	由于目前容量不足，Amazon RDS 无法创建实例。要在同一可用区中创建同一实例类型的数据库实例，请删除数据库实例，等待几个小时后，再尝试重新创建。或者，使用其他实例类或可用区创建新实例。
maintenance	已计费	Amazon RDS 正在对数据库实例应用维护更新。此状态用于 RDS 预先计划的实例级别的维护。
modifying	已计费	正在按照客户的请求修改数据库实例。
moving-to-vpc	已计费	数据库实例正移至新的 Amazon Virtual Private Cloud (Amazon VPC)。
rebooting	已计费	按照客户请求或需要重启数据库实例的 Amazon RDS 过程正在重启数据库实例。
resetting-master-credentials	已计费	正在按照客户请求重置数据库实例的主凭证。
renaming	已计费	正在按照客户请求重命名数据库实例。

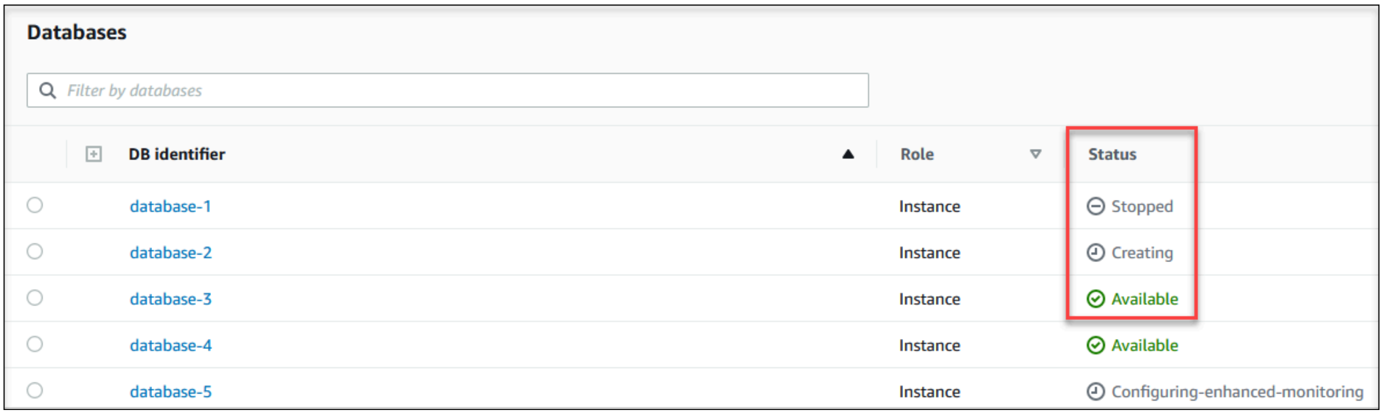
数据库实例状态	已计费	描述
restore-error	已计费	数据库实例在尝试还原到某个时间点或从快照还原时遇到错误。
starting	对存储计费	数据库实例正在启动。
stopped	对存储计费	数据库实例已停止。
stopping	对存储计费	正在停止数据库实例。
storage-config-upgrade	计费	正在升级数据库实例的存储文件系统配置。此状态仅适用于蓝绿部署中的绿色数据库或数据库实例只读副本。
storage-full	已计费	数据库实例达到了其存储分配容量。这是一种严重状态，我们推荐您立即修复该问题。为此，请通过修改数据库实例来扩展存储。要避免这种情况，请将 Amazon CloudWatch 警报设置为当存储空间逐渐减小时向您发出警告。
storage-initialization	计费	从快照还原后，数据库实例正在从 Amazon S3 加载数据块，以优化卷性能。它仍然可供操作，但在初始化完成之前，性能可能无法达到最佳。
storage-optimization	计费	Amazon RDS 正在优化数据库实例的存储。存储优化过程通常很短，但有时可能会达到甚至超过 24 小时。 在存储优化期间，数据库实例保持可用。存储优化进程在后台运行，不会影响实例的可用性。
upgrading	计费	数据库引擎或操作系统版本正在升级。

控制台

查看数据库实例的状态：

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。

   将显示 Databases（数据库）页面以及数据库实例的列表。对于的每个数据库实例，显示状态值。



CLI

要使用 Amazon CLI 查看数据库实例及其状态信息，请使用 describe-db-instances 命令。例如，以下 Amazon CLI 命令可列出所有数据库实例信息。

```
aws rds describe-db-instances
```

要查看特定数据库实例及其状态，请带以下选项调用 describe-db-instances 命令：

- `DBInstanceIdentifier` – 数据库实例的名称。

```
aws rds describe-db-instances --db-instance-identifier mydbinstance
```

要只是查看所有数据库集群的状态，请在 Amazon CLI 中使用以下查询。

```
aws rds describe-db-instances --query 'DBInstances[*].
[DBInstanceIdentifier,DBInstanceStatus]' --output table
```

API

要使用 Amazon RDS API 查看数据库实例的状态，请调用 DescribeDBInstances 操作。

来自 Amazon RDS 的建议

Amazon RDS 为数据库资源（例如数据库实例、、只读副本和数据库参数组）提供自动建议。这些建议通过分析数据库实例配置、使用和性能数据来提供最佳实践准则。

Amazon RDS 性能详情可监控特定指标，并自动通过分析指定资源可能存在问题的级别来创建阈值。当新的指标值在给定时间段内超过预定义的阈值时，性能详情会生成主动建议。此建议有助于防止数据库性能将来受到影响。例如，当连接到数据库的会话没有执行活动的工作，但可以保持阻止数据库资源时，将为 RDS for PostgreSQL 实例生成“空闲事务”建议。要获得主动建议，您必须开启性能详情，并设置付费套餐保留期。有关开启性能详情的信息，请参阅为 Amazon RDS 打开和关闭性能详情。有关性能详情的定价和数据留存的更多信息，请参阅性能详情的定价和数据留存。

DevOps Guru for RDS 会监控某些指标，以检测指标的行为何时变得高度不寻常或异常。这些异常被报告为带有建议的被动见解。例如，DevOps Guru for RDS 可能会建议您考虑增加 CPU 容量或调查导致数据库负载的等待事件。DevOps Guru for RDS 还提供基于阈值的主动建议。对于这些建议，您必须开启 DevOps Guru for RDS。有关开启 DevOps Guru for RDS 的信息，请参阅打开 DevOps Guru 并指定资源覆盖范围。

建议将处于以下任何状态：活动、已忽略、待处理或已解决。已解决的建议有效期为 365 天。

您可以查看或取消建议。您可以立即应用基于配置的活动建议，将其安排在下一个维护时段，或将其忽略。对于基于阈值的主动建议和基于机器学习的被动建议，您需要查看建议的问题原因，然后执行建议的操作来修复问题。

在以下 Amazon Web Services 区域中支持建议：

- 美国东部（俄亥俄州）
- 美国东部（弗吉尼亚州北部）
- 美国西部（加利福尼亚北部）
- 美国西部（俄勒冈州）
- 亚太地区（孟买）
- 亚太地区（首尔）
- 亚太地区（新加坡）
- 亚太地区（悉尼）
- 亚太地区（东京）
- 加拿大（中部）
- 欧洲地区（法兰克福）

- 欧洲地区（爱尔兰）
- 欧洲地区（伦敦）
- 欧洲地区（巴黎）
- 欧洲地区（斯德哥尔摩）
- 南美洲（圣保罗）

在以下各节中，学习查看、应用、忽略和修改来自 Amazon RDS 的建议。

主题

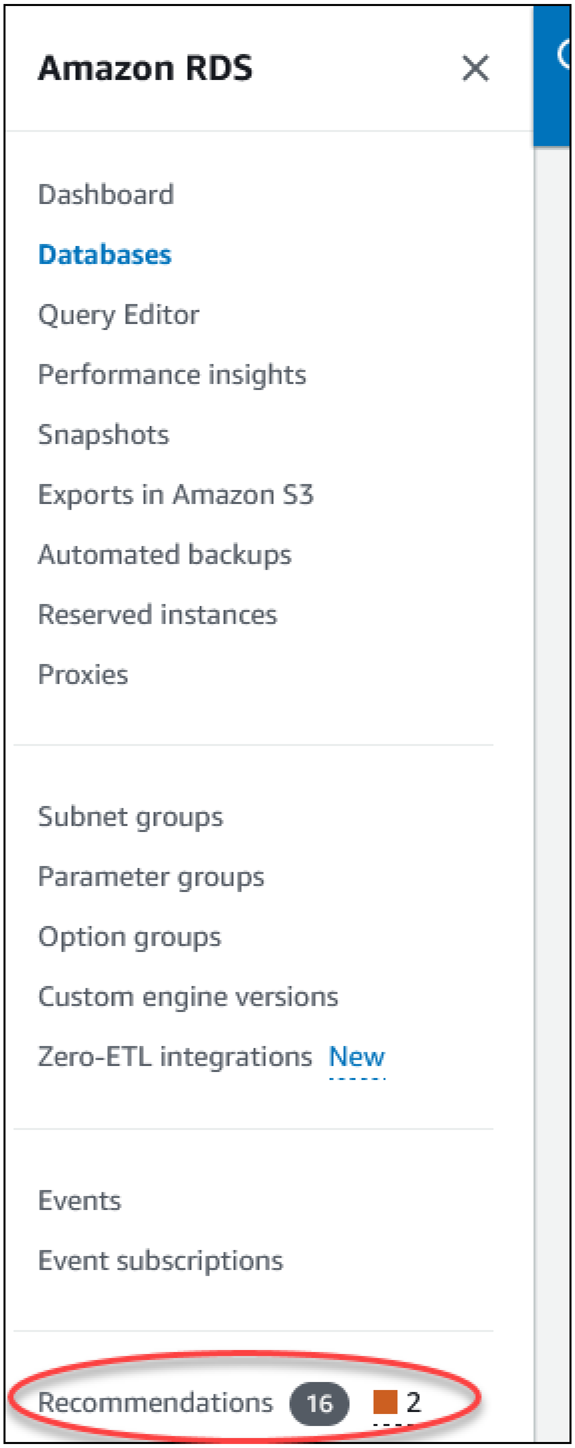
查看 Amazon RDS 建议

使用 Amazon RDS 控制台，您可以查看 Amazon RDS 针对您的数据库资源提出的建议。

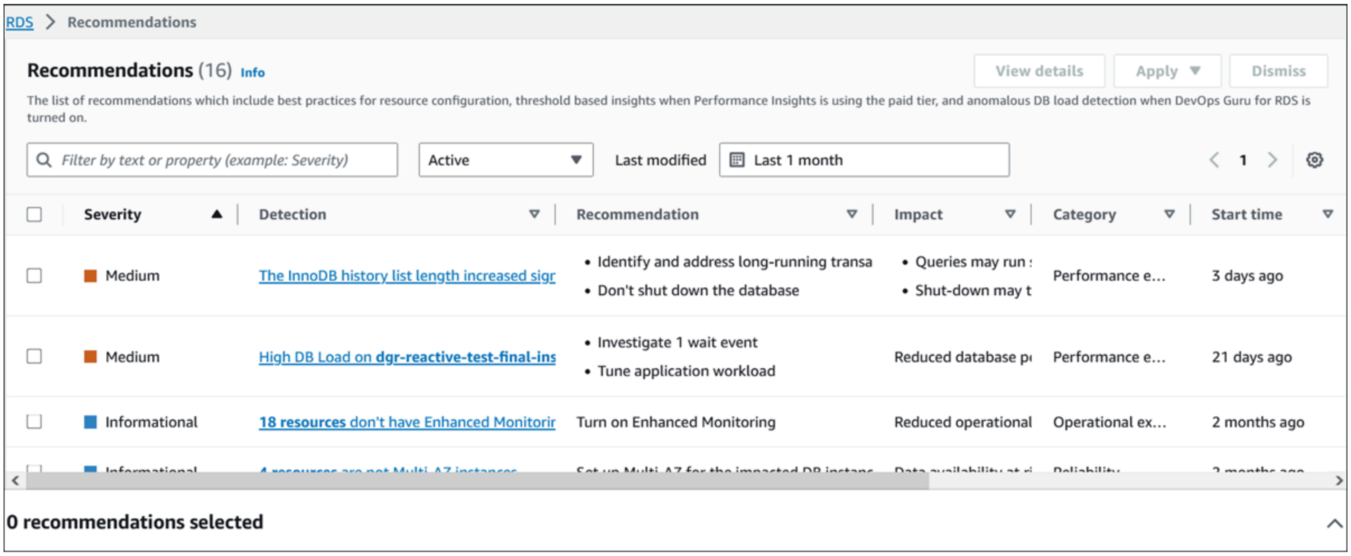
控制台

查看 Amazon RDS 建议

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，执行以下任何一项操作：
   - 选择建议。建议旁边提供了针对您的资源的有效建议数量和上个月生成的严重性最高的建议数量。要查找每个严重性的有效建议数量，请选择显示最高严重性的数字。

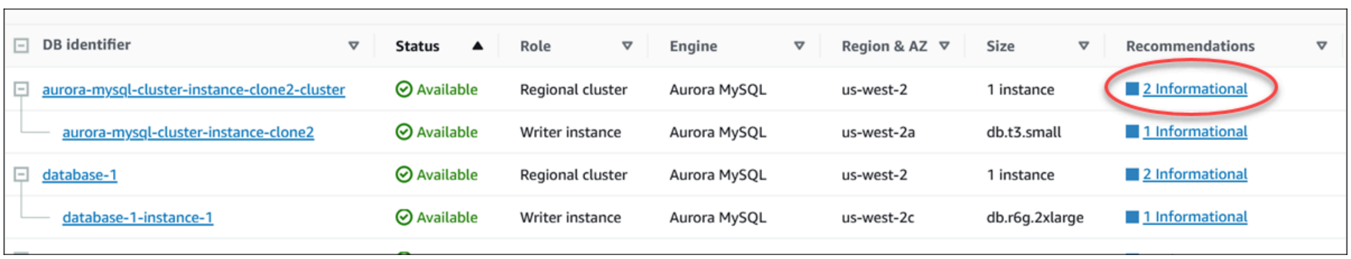


默认情况下，建议页面显示上个月的新建议列表。Amazon RDS 会针对您账户中的所有资源提供建议，并按其严重性对建议进行排序。

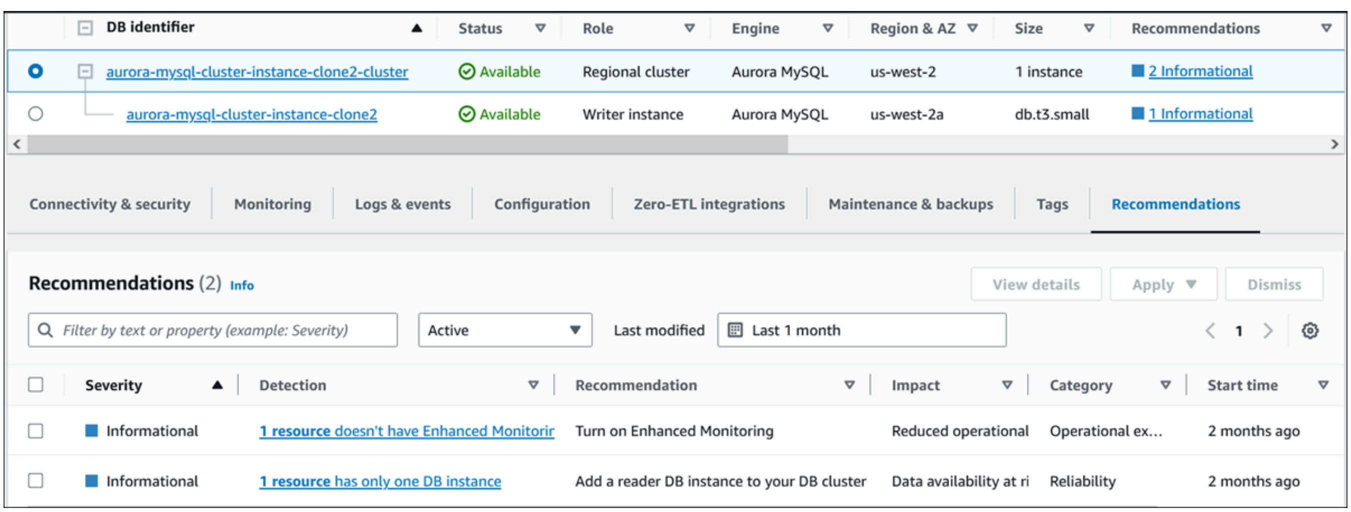


您可以选择建议以查看页面底部的部分，其中包含受影响的资源以及如何应用该建议的详细信息。

- 在数据库页中，为资源选择建议。



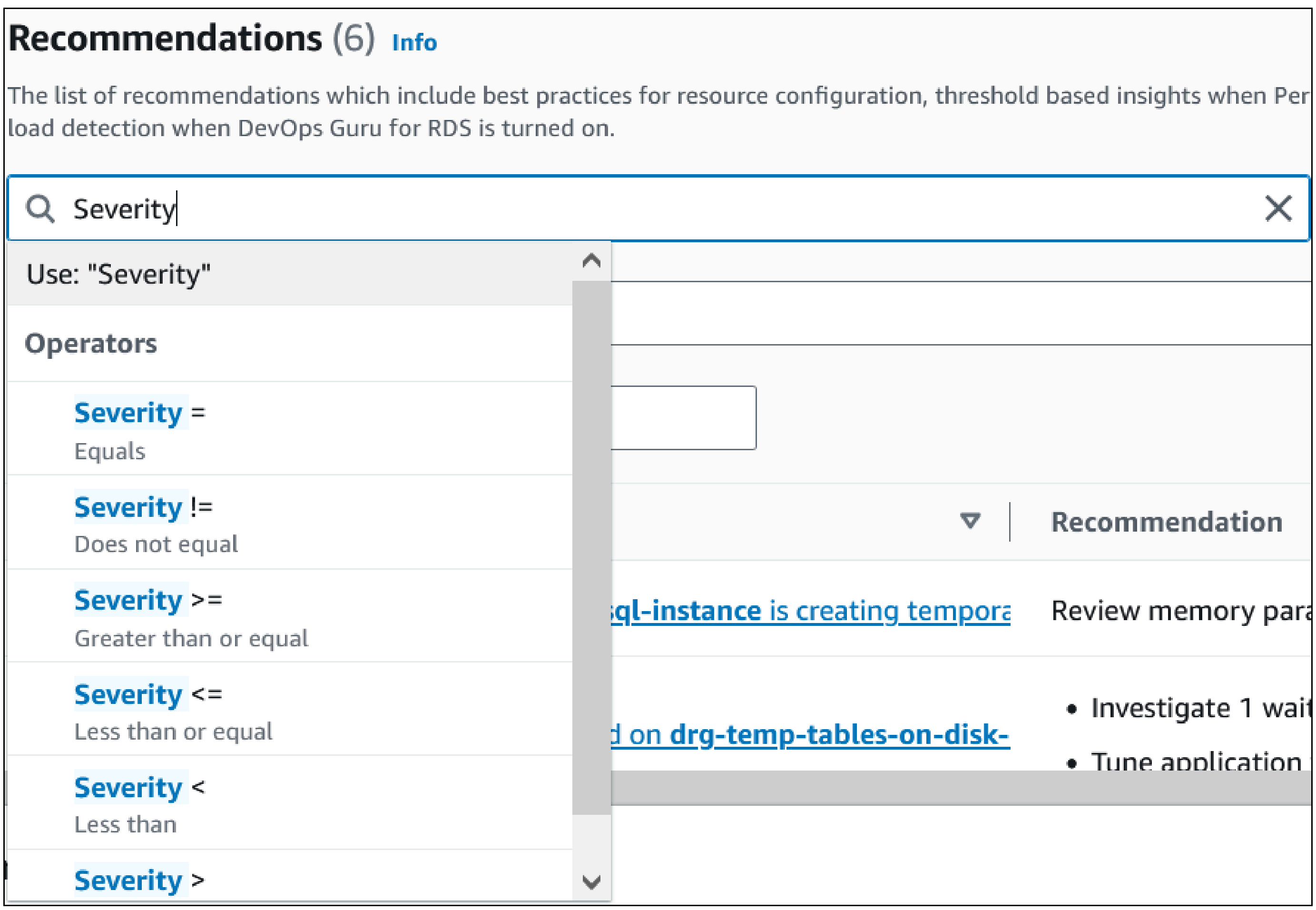
建议选项卡显示所选资源的建议及其详细信息。



以下详细信息可用于建议：

- 严重性 - 问题的含义级别。严重性级别为高、中、低和信息。
- 检测 - 受影响资源的数量和问题的简短描述。选择此链接可查看建议和分析详细信息。
- 建议 – 要应用的建议操作的简短描述。
- 影响 – 对不应用建议时可能产生的影响的简短描述。
- 类别 - 建议的类型。这些类别包括性能效率、安全性、可靠性、成本优化、卓越运营和可持续性。
- 状态 – 建议的当前状态。可能的状态为全部、活动、已忽略、已解决和待处理。
- 开始时间 – 问题开始的时间。例如，18 小时前。
- 上次修改时间 - 由于严重性更改而导致系统上次更新建议的时间，或者您对建议做出回应的时间。例如，10 小时前。
- 结束时间 - 问题结束的时间。对于任何持续存在的问题，不会显示此时间。
- 资源标识符 - 一个或多个资源的名称。

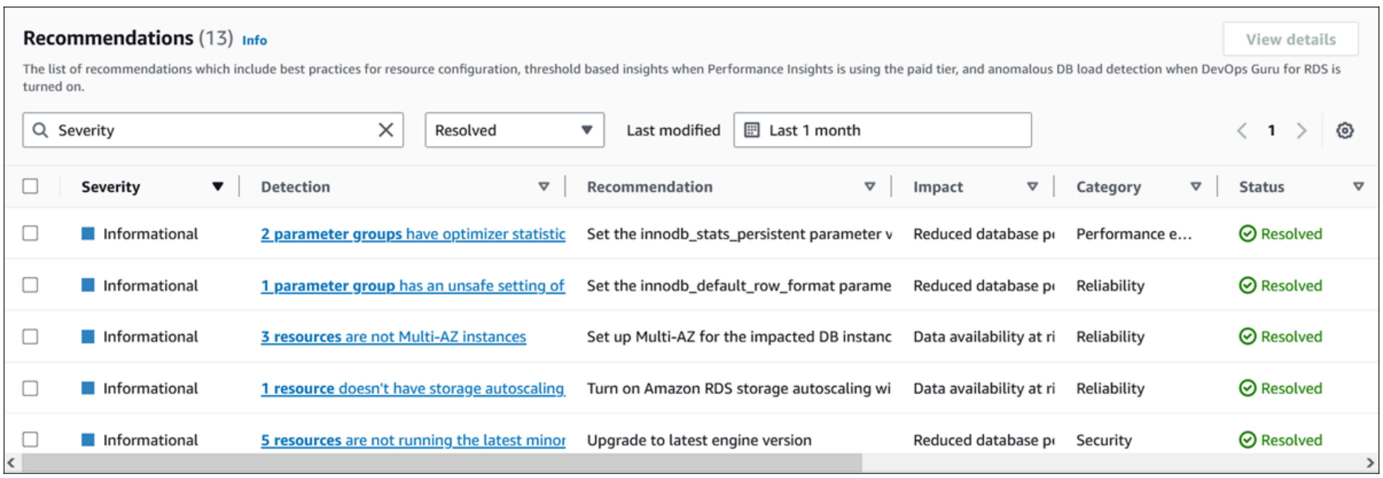
3. （可选）在字段中选择严重性或类别运算符以筛选建议列表。



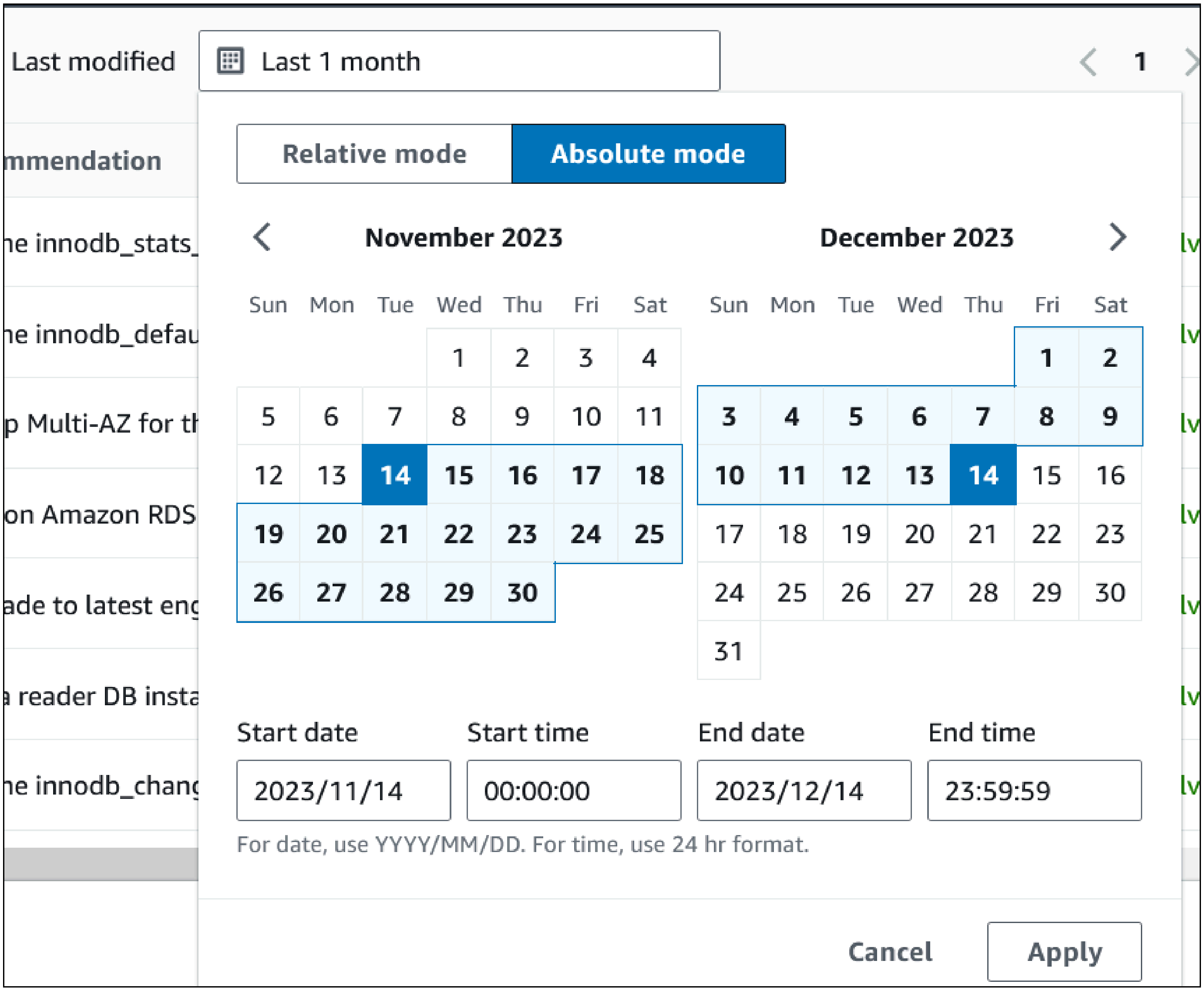
此时将显示所选操作的建议。

4. （可选）选择以下任何一个建议状态：

   - 活动（默认）– 显示您可以应用、计划在下一个维护时段应用或忽略的当前建议。
   - 全部 - 显示当前状态的所有建议。
   - 已忽略 - 显示已忽略的建议。
   - 已解决 - 显示已解决的建议。
   - 待处理 - 显示其建议的操作正在进行中或计划在下一个维护时段应用的建议。



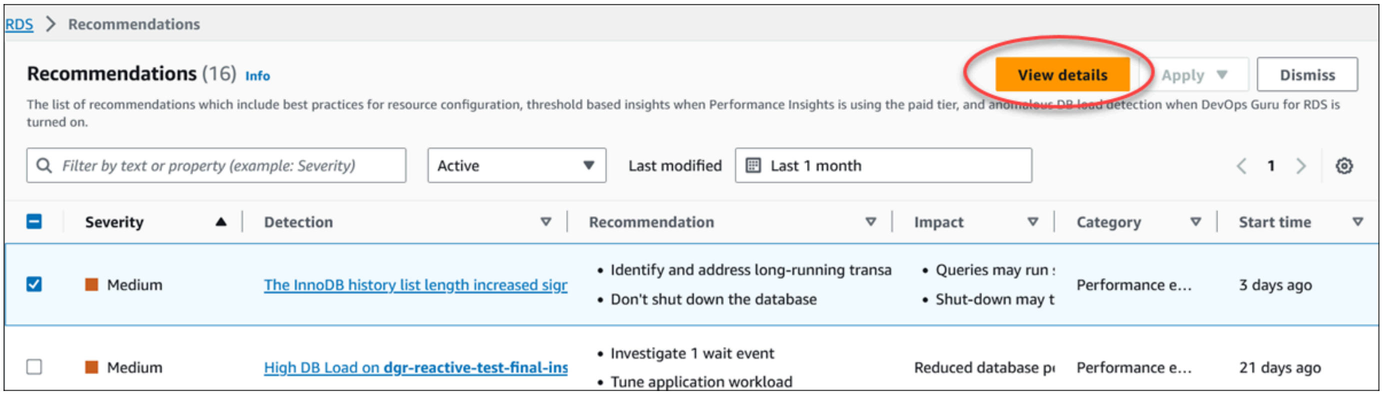
5. （可选）在上次修改时间中选择相对模式或绝对模式，以修改时间段。建议页面显示该时间段内生成的建议。默认时间段为最近 1 个月。在绝对模式下，您可以选择时间段，或者在开始日期和结束日期字段中输入时间。



将显示设定时间段的建议。

请注意，将范围设置为全部，即可查看您账户中资源的所有相关建议。

6. （可选）选择右侧的首选项以自定义要显示的详细信息。您可以选择页面大小、对文本换行以及允许或隐藏列。
7. （可选）选择建议，然后选择查看详细信息。



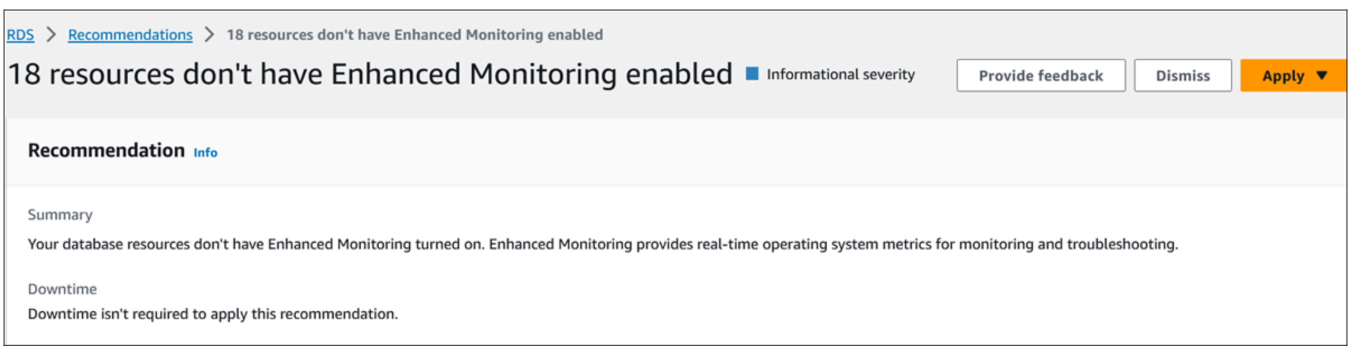
此时会显示建议详细信息页面。标题提供了检测到问题的资源总数和严重性。

有关基于异常的被动建议的详细信息页面上组件的信息，请参阅《Amazon DevOps Guru User Guide》中的 [Viewing reactive anomalies]。

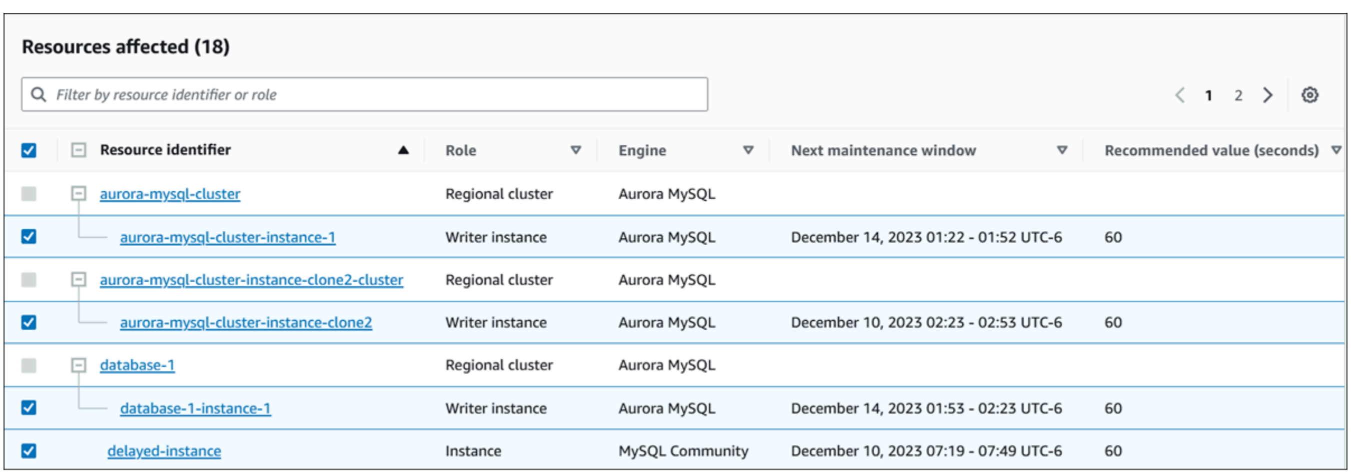
有关基于阈值的主动建议的详细信息页面上组件的信息，请参阅[查看性能详情主动建议]。

其它自动建议在建议详细信息页面上显示以下组件：

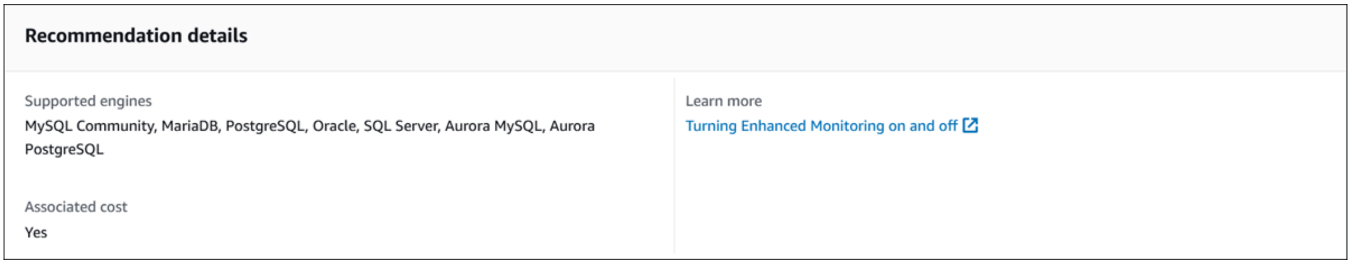
- 建议 – 建议的摘要以及是否需要停机才能应用建议。



- 受影响的资源 - 受影响资源的详细信息。



- 建议详细信息 – 支持的引擎信息、应用建议所需的任何相关费用以及用于了解更多信息的文档链接。

CLI

要查看数据库实例的 Amazon RDS 建议，请在 Amazon CLI 中使用以下命令。

```
aws rds describe-db-recommendations
```

RDS API

要使用 Amazon RDS API 查看 Amazon RDS 建议，请使用 DescribeDBRecommendations 操作。

应用 Amazon RDS 建议

要使用 Amazon RDS 控制台应用 Amazon RDS 建议，请在详细信息页面中选择基于配置的建议或受影响的资源。然后，选择立即应用该建议，或将其安排在下一个维护时段。资源可能需要重启才能使更改生效。对于一些数据库参数组建议，您可能需要重启资源。

基于阈值的主动建议或基于异常的被动建议不具有应用选项，可能需要进一步审查。

控制台

应用基于配置的建议

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，执行以下任一操作：

   - 选择建议。

     此时将出现建议页面，其中包含所有建议的列表。
   - 选择数据库，然后在数据库页中为资源选择建议。

详细信息显示在所选建议的建议选项卡中。

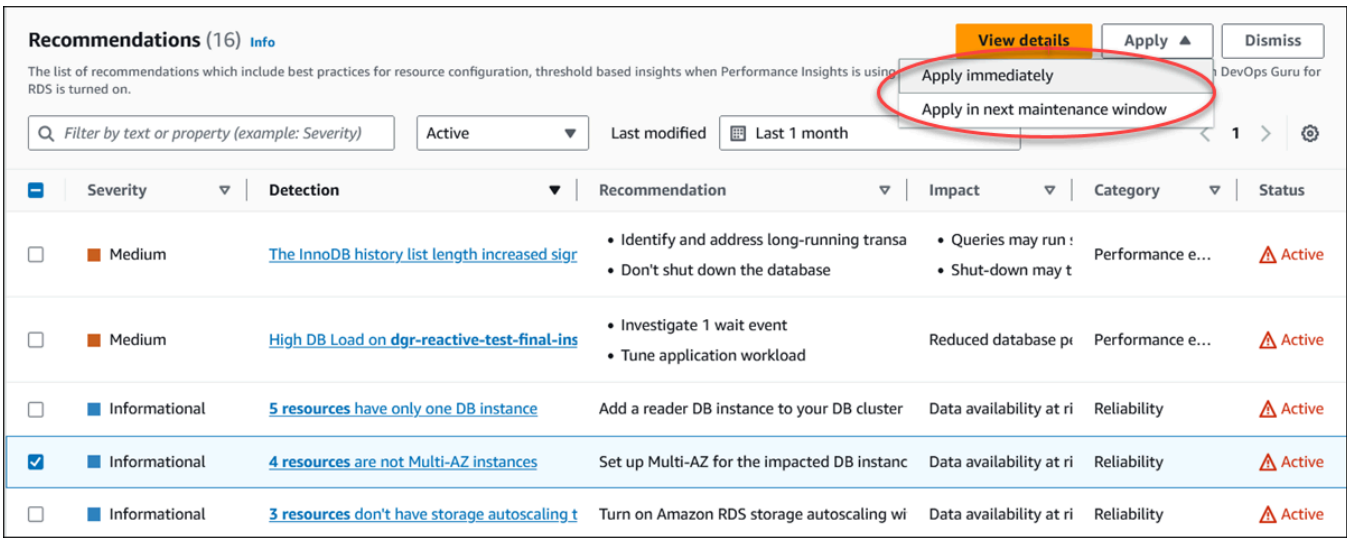
- 在建议页或数据库页的建议选项卡中，为活动的建议选择检测。

  此时会显示建议详细信息页面。

3. 在建议详细信息页面中选择一个建议或一个或多个受影响的资源，然后执行以下任一操作：

   - 选择应用，然后选择立即应用以立即应用建议。
   - 选择应用，然后选择在下一个维护时段内应用以计划在下一个维护时段中应用。

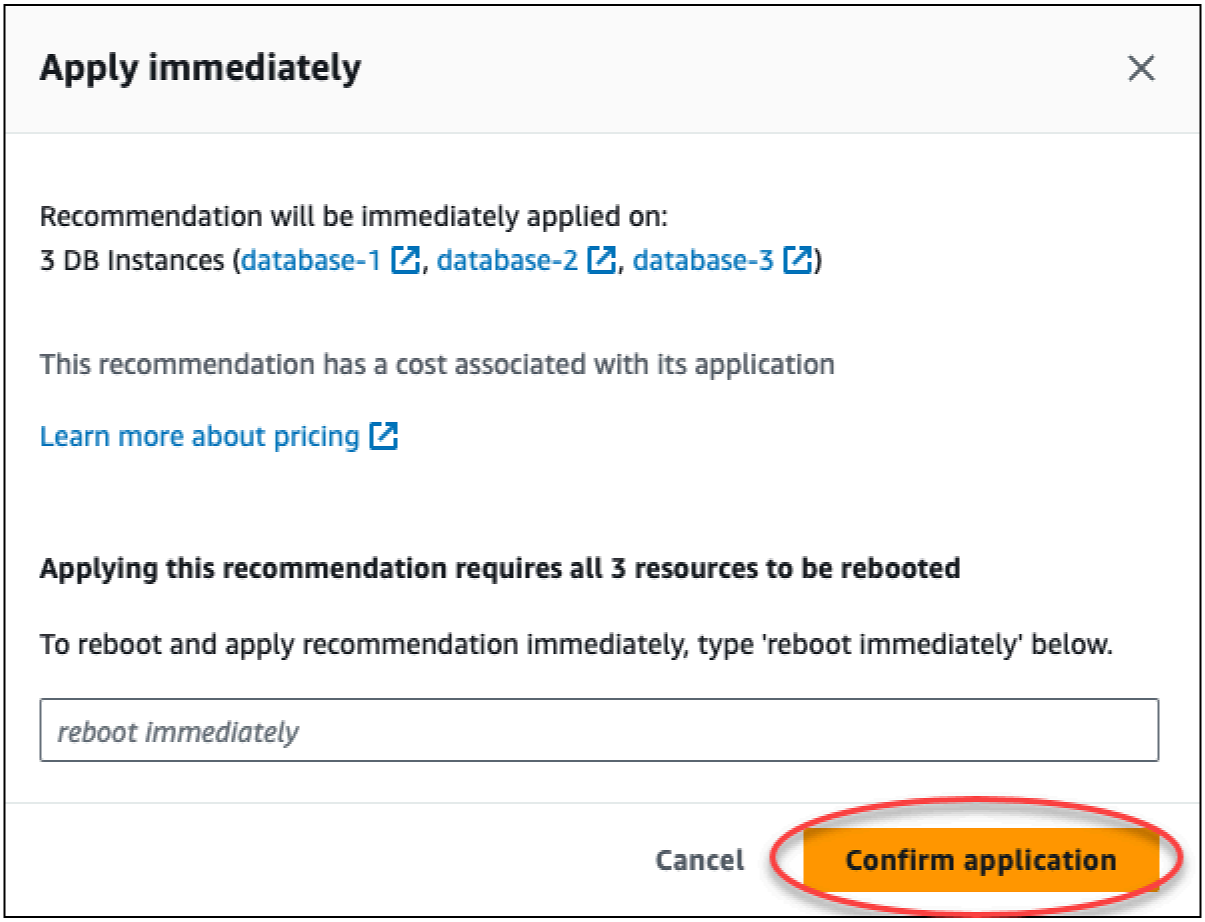
     在到达下一个维护时段之前，所选建议的状态将更新为待定。



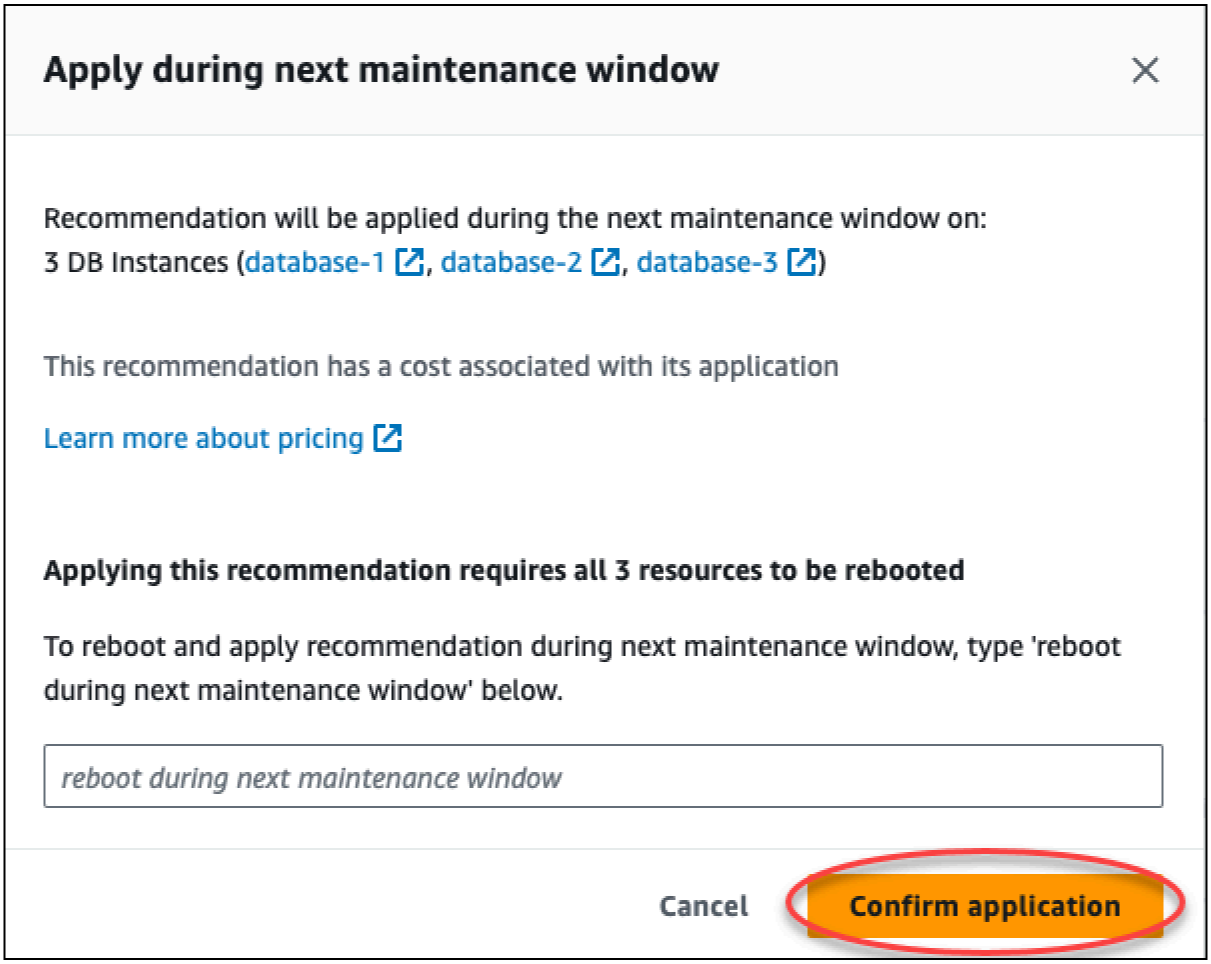
   将显示一个确认窗口。

4. 选择确认应用以应用建议。此窗口将确认资源是需要自动重启还是手动重启才能使更改生效。

   以下示例显示了立即应用建议的确认窗口。



以下示例显示了计划在下一个维护时段中应用建议的确认窗口。



当应用的建议成功或失败时，横幅会显示一条消息。

以下示例显示了带有成功消息的横幅。

以下示例显示了带有失败消息的横幅。

RDS API

使用 Amazon RDS API 应用基于配置的 RDS 建议

1. 使用 DescribeDBRecommendations 操作。输出中的 `RecommendedActions` 可以有一个或多个建议的操作。
2. 对步骤 1 中的每个建议操作使用 RecommendedAction 对象。输出包含 `Operation` 和 `Parameters`。

   以下示例显示了带有一个建议操作的输出。

```
    "RecommendedActions": [
        {
            "ActionId": "0b19ed15-840f-463c-a200-b10af1b552e3",
            "Title": "Turn on auto backup", // localized
            "Description": "Turn on auto backup for my-mysql-instance-1", //
 localized
            "Operation": "ModifyDbInstance",
            "Parameters": [
                {
                    "Key": "DbInstanceIdentifier",
                    "Value": "my-mysql-instance-1"
                },
                {
                    "Key": "BackupRetentionPeriod",
                    "Value": "7"
                }
            ],
            "ApplyModes": ["immediately", "next-maintenance-window"],
            "Status": "applied"
        },
        ... // several others
    ],
```

3. 对步骤 2 的输出中的每个建议操作使用 `operation` 并输入 `Parameters` 值。
4. 步骤 2 中的操作成功后，使用 ModifyDBRecommendation 操作修改建议状态。

忽略 Amazon RDS 建议

可以使用 Amazon RDS 控制台、Amazon CLI 或 Amazon RDS API 忽略一个或多个 Amazon RDS 建议。

控制台

忽略一个或多个建议

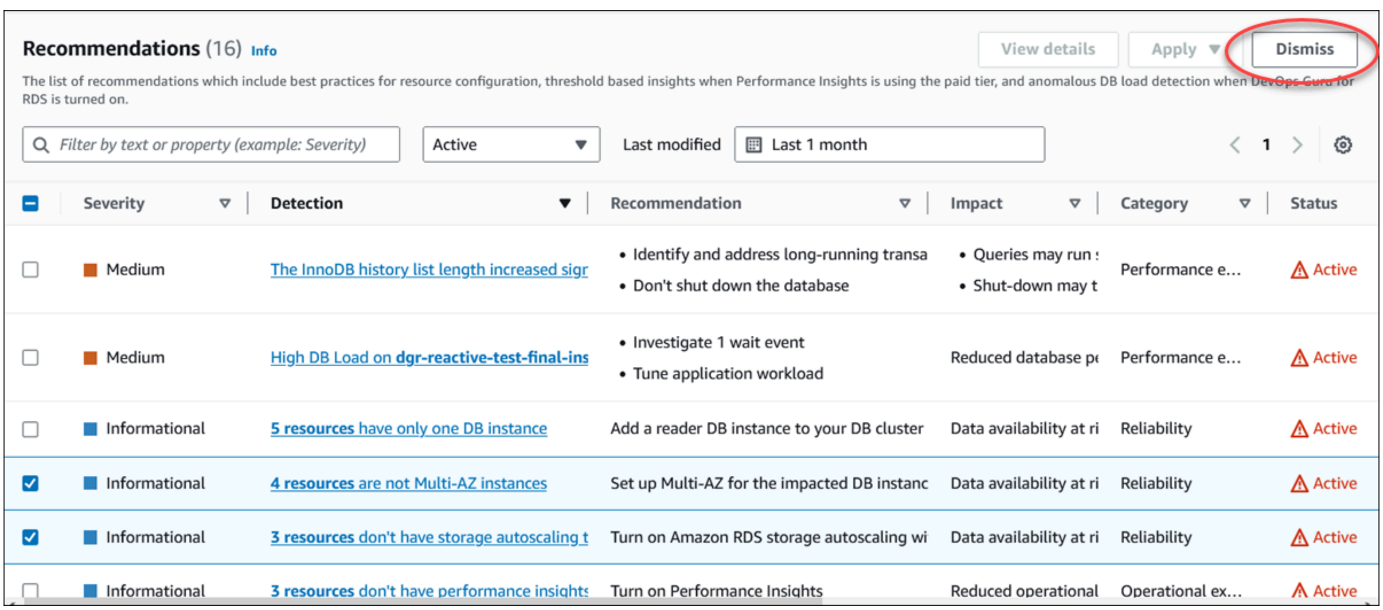
1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，执行以下任一操作：
   - 选择建议。

     此时将出现建议页面，其中包含所有建议的列表。
   - 选择数据库，然后在数据库页中为资源选择建议。

     详细信息显示在所选建议的建议选项卡中。
   - 在建议页或数据库页的建议选项卡中，为活动的建议选择检测。

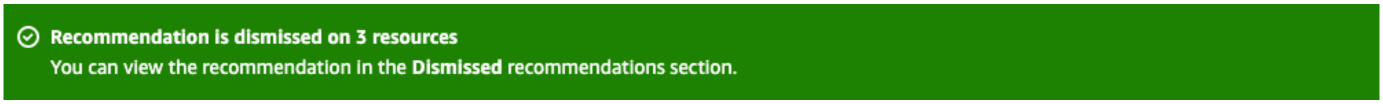
     建议详细信息页面显示受影响资源的列表。
3. 在建议详细信息页面中选择一个或多个建议，或者选择一个或多个受影响的资源，然后选择忽略。

   以下示例显示了建议页面，其中选定了多个要取消的活动建议。



当选定的一个或多个建议被忽略时，横幅会显示一条消息。

以下示例显示了带有成功消息的横幅。



以下示例显示了带有失败消息的横幅。

CLI

使用 Amazon CLI 忽略 RDS 建议

1. 运行 aws rds describe-db-recommendations --filters "Name=status,Values=active" 命令。

   输出提供了处于 active 状态的建议列表。
2. 查找要从步骤 1 中忽略的建议的 recommendationId。
3. 从步骤 2 中使用 recommendationId 运行命令 >aws rds modify-db-recommendation --status dismissed --recommendationId <ID> 以忽略建议。

RDS API

要使用 Amazon RDS API 忽略 RDS 建议，请使用 [ModifyDBRecommendation](#) 操作。

将已忽略的 Amazon RDS 建议修改为活动的建议

可以使用 Amazon RDS 控制台、Amazon CLI 或 Amazon RDS API，将一个或多个已忽略的 Amazon RDS 建议移至活动的建议。

控制台

将一个或多个已被忽略的建议移至活动的建议

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。

2. 在导航窗格中，执行以下任一操作：

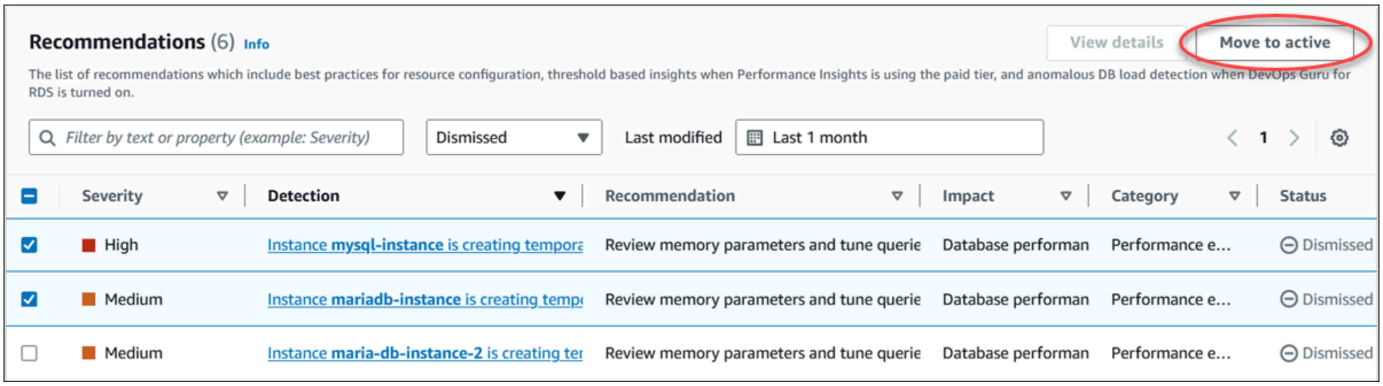
   • 选择建议。

     建议页面显示按您账户中所有资源的严重性排序的建议列表。

   • 选择数据库，然后在数据库页中为资源选择建议。

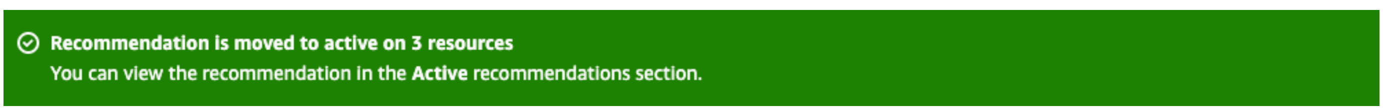
     建议选项卡显示所选资源的建议及其详细信息。

3. 从列表中选择一个或多个已忽略的建议，然后选择移至活动状态。



   当将所选建议从已忽略状态移至活动状态时，横幅会显示一条成功或失败消息。

   以下示例显示了带有成功消息的横幅。



   以下示例显示了带有失败消息的横幅。

CLI

使用 Amazon CLI 将已忽略的 RDS 建议更改为活动建议

1. 运行 `aws rds describe-db-recommendations --filters "Name=status,Values=dismissed"` 命令。

   输出提供了处于 `dismissed` 状态的建议列表。

2. 查找要从步骤 1 中更改状态的建议的 `recommendationId`。

3. 从步骤 2 中使用 `recommendationId` 运行命令 `>aws rds modify-db-recommendation --status active --recommendationId <ID>` 以更改为活动建议。

RDS API

要使用 Amazon RDS API 将已忽略的 RDS 建议更改为活动建议，请使用 ModifyDBRecommendation 操作。

Amazon RDS 中的建议参考

Amazon RDS 在创建或修改资源时，为资源生成建议。可以在下表中找到 Amazon RDS 中的建议示例。

类型	描述	建议	需要停机	其他信息
磁性介质卷正在使用中	您的数据库实例在使用磁性介质存储。对于大多数数据库实例，不建议使用磁性介质存储。选择其它存储类型：通用型（SSD）或预调配 IOPS。	选择其它存储类型：通用型（SSD）或预调配 IOPS。	是	Amazon EC2 文档中的上一代卷。
资源自动备份已关闭	您的数据库实例没有开启自动备份。建议使用自动备份，因为它们可以实现数据库实例的时间点恢复。	开启自动备份，保留期最长为 14 天。	是	启用自动备份 Amazon 数据库博客上的 Demystifying Amazon RDS backup storage costs
需要引擎次要版本升级	您的数据库资源未运行最新次要数据库引擎版本。最新的次要版本包含最新的安全修复和其它改进。	升级到最新的引擎版本。	是	升级数据库实例引擎版本

类型	描述	建议	需要停机	其他信息
增强监控已关闭	您的数据库资源未开启增强监控。增强监控提供用于监控和故障排除的实时操作系统指标。	打开增强监控。	否	使用增强监控来监控操作系统指标
存储加密已关闭	Amazon RDS 支持使用您在 Amazon Key Management Service（Amazon KMS）中管理的密钥对所有数据库引擎进行静态加密。在采用 Amazon RDS 加密的活动数据库实例上，静态存储在存储中的数据会加密，类似于自动备份、只读副本和快照。 如果在创建数据库实例时未开启加密，则需要先创建并还原数据库实例的解密快照的加密副本，然后再开启加密。	为您的数据库实例开启静态数据加密。	是	Amazon RDS 中的安全性 复制 Amazon RDS 的数据库快照
性能详情已关闭	性能详情用于监控数据库实例负载，有助于您分析和解决数据库性能问题。建议您开启性能详情。	开启性能详情。	否	在 Amazon RDS 上使用性能详情监控数据库负载

类型	描述	建议	需要停机	其他信息
数据库实例已关闭存储自动扩缩	您的数据库实例没有开启存储自动扩缩功能。当数据库工作负载增加时，RDS 存储自动扩缩会自动扩展存储容量，且无停机时间。	使用指定的最大存储阈值启用 Amazon RDS 存储自动扩缩	否	使用 Amazon RDS 存储自动扩展功能自动管理容量
需要更新 RDS 资源主要版本	不支持使用数据库引擎当前主要版本的数据库。建议您升级到包含新功能和增强功能的最新主要版本。	升级至数据库引擎的最新主要版本。	是	升级数据库实例引擎版本 使用 Amazon RDS 蓝绿部署进行数据库更新
需要更新 RDS 资源实例类	您的数据库实例正在运行更早代次的数据库实例类。我们已经将更早代次的数据库实例类替换为成本和/或性能更高的数据库实例类。建议您使用更新代次的数据库实例类运行数据库实例。	升级数据库实例类。	是	数据库实例类支持的数据库引擎
RDS 资源使用的是随附许可证的终止支持引擎版本	建议您将主要版本升级到 Amazon RDS 支持的最新引擎版本，以继续使用当前的许可证支持。当前许可证不支持数据库的引擎版本。	建议您将数据库升级到 Amazon RDS 中支持的最新版本，以便继续使用许可模型。	是	Oracle 主要版本升级

类型	描述	建议	需要停机	其他信息
数据库实例未使用多可用区部署	建议您使用多可用区部署。多可用区部署可增强数据库实例的可用性和持久性。	为受影响的数据库实例设置多可用区	否 在此更改期间，不会停机。不过，可能会对性能产生影响。有关更多信息，请参阅 将 Amazon RDS 的数据库实例转换为多可用区部署	Amazon RDS Multi-AZ 的定价

类型	描述	建议	需要停机	其他信息
数据库内存参数与默认值不同	数据库实例的内存参数与默认值明显不同。这些设置可能会影响性能并导致错误。 建议您将数据库实例的自定义内存参数重置为数据库参数组中的默认值。	将内存参数重置为其默认值。	否	Amazon 数据库博客上的 [Best practices for configuring performance parameters for Amazon RDS for MySQL](#)
`InnoDB_Change_Buffering` 参数使用的值小于最佳值	更改缓冲将允许 MySQL 数据库实例推迟维护二级索引所需的一些写入操作。此功能在磁盘速度较慢的环境中非常有用。更改缓冲配置稍微提高了数据库性能，但在升级期间导致了崩溃恢复延迟和长时间关机。在 MySQL 版本 8.4 中默认设置为 `OFF`。	在数据库参数组中将 `InnoDB_Change_Buffering` 参数值设置为 `NONE`。	否	Amazon 数据库博客上的 [Best practices for configuring performance parameters for Amazon RDS for MySQL](#)
查询缓存参数已开启	当更改要求清除查询缓存时，您的数据库实例将显示为停滞状态。大多数工作负载不会受益于查询缓存。查询缓存已从 MySQL 8.0 及更高版本中删除。建议您将 query_cache_type 参数设置为 0。	在数据库参数组中将 `query_cache_type` 参数值设置为 `0`。	是	Amazon 数据库博客上的 [Best practices for configuring performance parameters for Amazon RDS for MySQL](#)

类型	描述	建议	需要停机	其他信息
log_output 参数设置为表	如果 log_output 设置为 TABLE，则使用的存储比 log_output 设置为 FILE 时更多。建议您将此参数设置为 FILE，以避免达到存储大小限制。在 MySQL 8.4 及更高版本中默认设置为 FILE。	在数据库参数组中将 log_output 参数值设置为 FILE。	否	MySQL 数据库日志文件
参数组不使用大页	大页可以提高数据库的可扩展性，但您的数据库实例不使用大页。建议您在数据库实例的数据库参数组中将 use_large_pages 参数值设置为 ONLY。	在数据库参数组中将 use_large_pages 参数值设置为 ONLY。	是	为 RDS for Oracle 实例开启大页
autovacuum 参数已关闭	数据库实例 的 autovacuum 参数已关闭. 关闭 autovacuum 会增加表和索引膨胀并影响性能。 建议您在数据库参数组中开启 autovacuum。	在数据库参数组中开启 autovacuum 参数。	否	Amazon 数据库博客上的 Understanding autovacuum in Amazon RDS for PostgreSQL environments

类型	描述	建议	需要停机	其他信息
synchronous_commit 参数已关闭	关闭 synchronous_commit 参数后，数据库崩溃可能会导致数据丢失。数据库的持久性受到威胁。 建议您开启此 synchronous_commit 参数。	在数据库参数组中开启 synchronous_commit 参数。	是	Amazon 数据库博客上的 Amazon Aurora PostgreSQL parameters: Replication, security, and logging。
track_counts 参数已关闭	当 track_counts 参数处于关闭状态时，数据库不会收集数据库活动统计数据。Autovacuum 需要这些统计信息才能正常工作。 建议您将 track_counts 参数设置为 1。	将 track_counts 数据设置为 1。	否	PostgreSQL 的运行时统计数据
enable_indexonlyscan 参数已关闭	关闭仅限索引的扫描计划类型时，查询计划程序或优化程序无法使用该类型。 建议您将 enable_indexonlyscan 参数值设置为 1。	将 enable_indexonlyscan 参数值设置为 1。	否	PostgreSQL 的计划程序方法配置

类型	描述	建议	需要停机	其他信息
enable_indexscan 参数已关闭	关闭索引扫描计划类型时，查询计划程序或优化程序无法使用该类型。 建议您将 enable_indexscan 值设置为 1。	将 enable_indexscan 参数值设置为 1。	否	[PostgreSQL 的计划程序方法配置]()
innodb_flush_log_at_trx 参数已关闭	您的数据库实例的 innodb_flush_log_at_trx 参数值不是安全的值。此参数控制向磁盘提交操作的持久性。 建议您将 innodb_flush_log_at_trx 参数设置为 1。	将 innodb_flush_log_at_trx 参数值设置为 1。	否	Amazon 数据库博客上的 [Best practices for configuring performance parameters for Amazon RDS for MySQL]()
sync_binlog 参数已关闭	在数据库实例中确认事务提交之前，不会强制将二进制日志同步到磁盘。 建议您将 sync_binlog 参数值设置为 1。	将 sync_binlog 参数值设置为 1。	否	Amazon 数据库博客上的 [Best practices for configuring replication parameters for Amazon RDS for MySQL]()

类型	描述	建议	需要停机	其他信息
innodb_stats_persistent 参数已关闭	您的数据库实例未配置为将 InnoDB 统计信息持久保存到磁盘上。如果不存储统计数据，则每次实例重启和访问表时都会重新计算统计数据。这会导致查询执行计划的差异。您可以在表级别修改此全局参数的值。 建议您将 innodb_stats_persistent 参数值设置为 ON。	将 innodb_stats_persistent 参数值设置为 ON。	否	Amazon 数据库博客上的 [Best practices for configuring performance parameters for Amazon RDS for MySQL]
innodb_open_files 参数较低	innodb_open_files 参数控制 InnoDB 一次可打开的文件数量。当 mysqld 运行时，InnoDB 会打开所有日志和系统表空间文件。 对于您的数据库实例，InnoDB 一次可打开的最大文件数的值较低。建议您将 innodb_open_files 参数设置为最小值 65。	将 innodb_open_files 参数设置为最小值 65。	是	[InnoDB 对于 MySQL 打开的文件]

类型	描述	建议	需要停机	其他信息
max_user_connections 参数较低	对于您的数据库实例，每个数据库账户的最大同时连接数的值较低。 建议您将 max_user_connections 参数设置为大于 5 的数字。	将 max_user_connections 参数的值增加到大于 5 的数字。	是	设置 MySQL 的账户资源限制
只读副本在可写模式下打开	您的数据库实例的只读副本处于可写模式，这允许来自客户端的更新。 建议您将 read_only 参数设置为 TrueIfReplica ，这样只读副本就不会处于可写模式。	将 read_only 参数值设置为 TrueIfReplica 。	否	Amazon 数据库博客上的 Best practices for configuring replication parameters for Amazon RDS for MySQL

类型	描述	建议	需要停机	其他信息
innodb_default_row_format 参数设置不安全	您的数据库实例遇到一个已知问题：当索引超过 767 字节时，在低于 8.0.26 的 MySQL 版本中创建且 row_format 设置为 COMPACT 或 REDUNDANT 的表将不可访问且无法恢复。 建议您将 innodb_default_row_format 参数值设置为 DYNAMIC。	将 innodb_default_row_format 参数值设置为 DYNAMIC。	否	[MySQL 8.0.26 中的变化](#)
general_logging 参数已开启	您的数据库实例已开启常规日志记录。在排查数据库问题时，此设置非常有用。但是，开启常规日志记录会增加 I/O 操作量和分配的存储空间量，这可能导致争用和性能降级。 请检查您对常规日志记录使用情况的要求。建议您将 general_logging 参数值设置为 0。	请检查您对常规日志记录使用情况的要求。如果它不是强制性的，建议您将 general_logging 参数值设置为 0。	否	[RDS for MySQL 数据库日志概览](#)

类型	描述	建议	需要停机	其他信息
RDS 实例的系统内存容量预调配不足	建议您调整查询以使用更少的内存，或者使用所分配内存更高的数据库实例类型。当实例内存不足时，数据库性能就会受到影响。	使用内存容量更高的数据库实例	是	Amazon 数据库博客上的 Scaling Your Amazon RDS Instance Vertically and Horizontally Amazon RDS 实例类型 Amazon RDS 定价
RDS 实例的系统 CPU 容量预调配不足	建议您调整查询来使用更少的 CPU，或修改数据库实例来使用所分配的 vCPU 更高的数据库实例类。当运行数据库实例的 CPU 容量不足时，数据库性能可能会下降。	使用 CPU 容量更高的数据库实例	是	Amazon 数据库博客上的 Scaling Your Amazon RDS Instance Vertically and Horizontally Amazon RDS 实例类型 Amazon RDS 定价
RDS 资源未正确地利用连接池	建议您启用 Amazon RDS 代理，以便高效地池化和共享现有数据库连接。如果您已经在为数据库使用代理，请正确地配置代理，来改善多个数据库实例之间的连接池和负载均衡。RDS 代理有助于降低连接耗尽和停机的风险，同时提高可用性和可扩展性。	启用 RDS 代理或修改现有的代理配置	否	Amazon 数据库博客上的 Scaling Your Amazon RDS Instance Vertically and Horizontally 的 Amazon RDS 代理 Amazon RDS 代理定价

类型	描述	建议	需要停机	其他信息
RDS 实例正在创建过多的临时对象	建议您调整工作负载以防止创建过多的临时对象，或者切换到支持优化读取的 RDS 实例类。RDS 优化型读取功能可提高涉及大量临时对象和/或大型临时对象的工作负载的数据库性能。评估您的工作负载，以确定使用具有 RDS 优化型读取功能的实例是否有利于您的数据库工作负载。	使用带 RDS 优化型读取功能的数据库实例类型	是	Amazon RDS 实例类型 使用 Amazon RDS 优化型读取功能提高 RDS for MySQL 的查询性能 使用 Amazon RDS 优化型读取功能提高 RDS for MariaDB 的查询性能 使用 Amazon RDS 优化型读取功能提高 RDS for PostgreSQL 的查询性能
RDS 实例的系统 IOPS 容量预调配不足	我们建议调整数据库工作负载以降低 IOPS，或者将数据库实例纵向扩展到具有更高的默认 IOPS 限制的类型。当前数据库实例无法支持预调配 IOPS，或者数据库工作负载具有较高的 IOPS 利用率。	使用默认 IOPS 限制更高的数据库实例类型	是	Amazon RDS 实例类型 Amazon RDS 数据库实例存储 数据库负载

类型	描述	建议	需要停机	其他信息
RDS 实例具有预调配不足的 Amazon EBS 卷	我们建议调整数据库工作负载以降低 IOPS 或增加数据库的预调配 IOPS。当 IOPS 利用率接近预调配 IOPS 时，数据库性能可能会下降。	为数据库实例预调配更多 IOPS	是	Amazon RDS 实例类型 Amazon RDS 数据库实例存储 数据库负载
RDS 实例的吞吐能力预调配不足	我们建议调整数据库工作负载以降低吞吐量或增加数据库的预调配吞吐量。当吞吐量利用率接近预调配吞吐量时，数据库性能可能会受到影响。	为数据库实例预调配更多吞吐量	是	Amazon RDS 实例类型 Amazon RDS 数据库实例存储 数据库负载
RDS 实例的 EBS I/O 预置不足	我们建议调整数据库工作负载来减少 I/O 操作，或者修改数据库实例来使用 Amazon RDS io2 Block Express 卷，这些卷专为需要高性能、高吞吐量和低延迟的数据库工作负载而设计。在当前工作负载下，数据库可能无法以所需的速率处理 I/O 操作，这可能导致性能下降。	将 Amazon RDS io2 Block Express 卷用于 RDS 实例	否	Amazon RDS 数据库实例存储 Amazon RDS 的 Amazon CloudWatch 指标 《Amazon EBS 用户指南》中的 Provisioned IOPS SSD volumes

在 Amazon RDS 控制台中查看指标

Amazon RDS 与 Amazon CloudWatch 集成以显示 RDS 控制台中的各种 RDS 数据库实例指标。有关这些指标的说明，请参阅 Amazon RDS 的指标参考。

对于数据库实例，将监控以下类别的指标：

- CloudWatch – 显示您可以在 RDS 控制台中访问的 RDS 的 Amazon CloudWatch 指标。您也可以在 CloudWatch 控制台中访问这些指标。每个指标均包括一个图形，显示特定时间范围内监控的指标。有关 CloudWatch 指标的列表，请参阅 Amazon RDS 的 Amazon CloudWatch 指标。
- 增强监控 - 显示 RDS 数据库实例打开增强监控时操作系统指标的摘要。RDS 将增强监控中的指标传输到您的 Amazon CloudWatch Logs 账户。每个操作系统指标均包括一个图形，显示特定时间范围内监控的指标。有关概述，请参阅使用增强监控来监控操作系统指标。有关增强监控指标的列表，请参阅 增强监控中的操作系统指标。
- OS 进程列表 – 显示您的数据库实例中运行的每个进程的详细信息。
- Performance Insights - 打开 中数据库实例的 Amazon RDS Performance Insights 控制面板。有关 Performance Insights 概述，请参阅 在 Amazon RDS 上使用性能详情监控数据库负载。有关 Performance Insights 指标的列表，请参阅 Amazon RDS 性能详情的 Amazon CloudWatch 指标。

Amazon RDS 现在在性能详情控制面板中提供性能详情和 CloudWatch 指标的合并视图。必须为您的数据库实例启用性能详情，才能使用此视图。您可以在监控选项卡中选择新的监控视图，也可以在导航窗格中选择性能详情。要查看有关选择此视图的说明，请参阅使用性能详情控制面板查看组合指标。

使用性能详情控制面板查看组合指标

> **Important**
>
> Amazon 已宣布性能详情的生命周期终止日期为：2025 年 11 月 30 日。在此日期之后，Amazon RDS 将不再支持性能详情控制台体验、灵活的保留期（1-24 个月）及其关联定价。性能详情 API 将继续存在，定价不变。性能详情 API 的费用将与 CloudWatch 数据库洞察的费用一起显示在您的 Amazon 账单中。
>
> 我们建议您在 2025 年 11 月 30 日之前，将使用性能详情付费套餐的所有数据库实例升级到数据库洞察的高级模式。有关升级到数据库洞察的高级模式的信息，请参阅开启适用于 Amazon RDS 的数据库洞察的高级模式。
>
> 如果您不采取任何行动，则使用性能详情的数据库实例将默认为使用数据库洞察的标准模式。在数据库洞察的标准模式下，您可能会在 7 天后无法访问性能数据历史记录，并且可能无法在 Amazon RDS 控制台中使用执行计划和按需分析功能。在 2025 年 11 月 30 日之后，只有数据库洞察的高级模式将支持执行计划和按需分析。
>
> 借助 CloudWatch 数据库洞察，您可以监控数据库实例集的数据库负载，并对性能进行大规模分析和故障排除。有关数据库洞察的更多信息，请参阅使用 CloudWatch 数据库洞察监控 Amazon RDS 数据库。有关定价信息，请参阅 Amazon CloudWatch 定价。

Amazon RDS 现在在性能详情控制面板中提供数据库实例的性能详情和 CloudWatch 指标的合并视图。您可以使用预配置的控制面板或创建自定义控制面板。预配置的控制面板提供最常用的指标，以帮助诊断数据库引擎的性能问题。或者，您可以创建一个自定义控制面板，其中包含数据库引擎的可满足分析需求的指标。然后，将此控制面板用于您的 Amazon 账户中该数据库引擎类型的所有数据库实例。

您可以在监控选项卡中选择监控视图，也可以在导航窗格中选择性能详情。

必须为数据库实例开启性能详情，才能在性能详情控制面板中查看组合指标。有关开启性能详情的更多信息，请参阅为 Amazon RDS 打开和关闭性能详情。

在以下各节中，您可以了解使用新版和旧版监控视图来显示性能详情和 CloudWatch 指标。

主题

从“监控”选项卡中选择新版监控视图

Important

Amazon 已宣布性能详情的生命周期终止日期为：2025 年 11 月 30 日。在此日期之后，Amazon RDS 将不再支持性能详情控制台体验、灵活的保留期（1-24 个月）及其关联定价。性能详情 API 将继续存在，定价不变。性能详情 API 的费用将与 CloudWatch 数据库洞察的费用一起显示在您的 Amazon 账单中。

我们建议您在 2025 年 11 月 30 日之前，将使用性能详情付费套餐的所有数据库实例升级到数据库洞察的高级模式。有关升级到数据库洞察的高级模式的信息，请参阅开启适用于 Amazon RDS 的数据库洞察的高级模式。

如果您不采取任何行动，则使用性能详情的数据库实例将默认为使用数据库洞察的标准模式。在数据库洞察的标准模式下，您可能会在 7 天后无法访问性能数据历史记录，并且可能无法在 Amazon RDS 控制台中使用执行计划和按需分析功能。在 2025 年 11 月 30 日之后，只有数据库洞察的高级模式将支持执行计划和按需分析。

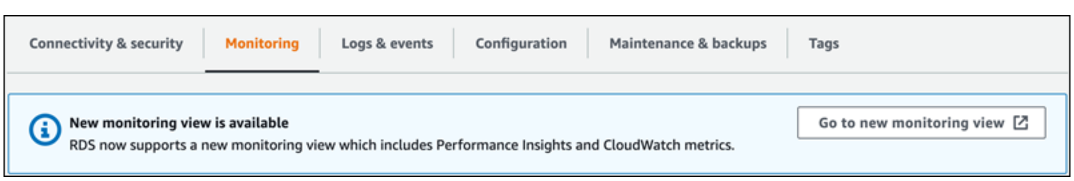
借助 CloudWatch 数据库洞察，您可以监控数据库实例集的数据库负载，并对性能进行大规模分析和故障排除。有关数据库洞察的更多信息，请参阅使用 CloudWatch 数据库洞察监控 Amazon RDS 数据库。有关定价信息，请参阅 Amazon CloudWatch 定价。

从 Amazon RDS 控制台中，您可以选择新版监控视图来查看数据库实例的性能详情和 CloudWatch 指标。

在“监控”选项卡中选择新版监控视图

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在左侧导航窗格中，选择数据库。
3. 选择要监控的数据库实例的名称。
4. 向下滚动并选择监控选项卡。

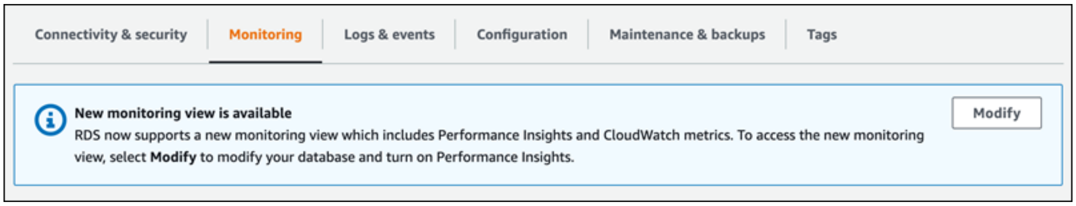
   此时会出现一条横幅，其中包含选择新监控视图的选项。以下示例显示了选择新监控视图的横幅。



5. 选择转到新的监控视图以打开性能详情控制面板，其中包含数据库实例的性能详情和 CloudWatch 指标。

6. （可选）如果对数据库实例关闭了性能详情，则会出现一个横幅，其中包含修改数据库集群和开启性能详情的选项。

   以下示例显示了在监控选项卡中修改数据库集群的横幅。

   选择修改以修改数据库集群并开启性能详情。有关开启性能详情的更多信息，请参阅为 Amazon RDS 打开和关闭性能详情

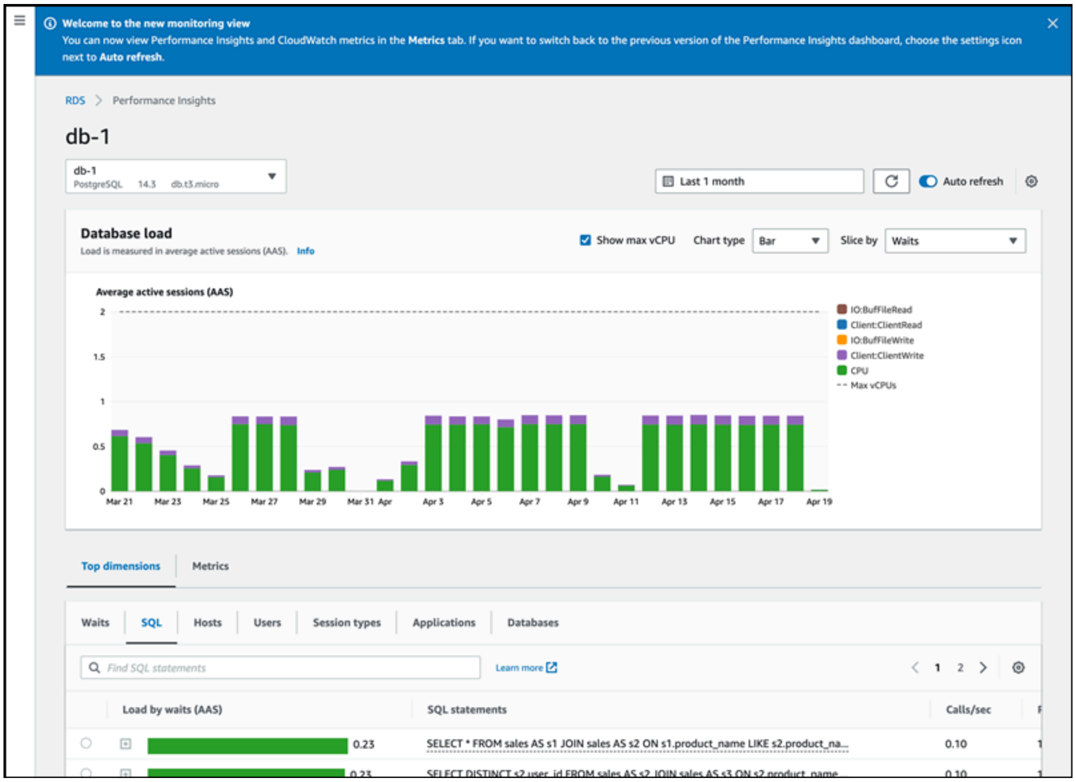
从性能详情页面中选择新版监控视图

> **Important**
>
> Amazon 已宣布性能详情的生命周期终止日期为：2025 年 11 月 30 日。在此日期之后，Amazon RDS 将不再支持性能详情控制台体验、灵活的保留期（1-24 个月）及其关联定价。性能详情 API 将继续存在，定价不变。性能详情 API 的费用将与 CloudWatch 数据库洞察的费用一起显示在您的 Amazon 账单中。
>
> 我们建议您在 2025 年 11 月 30 日之前，将使用性能详情付费套餐的所有数据库实例升级到数据库洞察的高级模式。有关升级到数据库洞察的高级模式的信息，请参阅开启适用于 Amazon RDS 的数据库洞察的高级模式。
>
> 如果您不采取任何行动，则使用性能详情的数据库实例将默认为使用数据库洞察的标准模式。在数据库洞察的标准模式下，您可能会在 7 天后无法访问性能数据历史记录，并且可能无法在 Amazon RDS 控制台中使用执行计划和按需分析功能。在 2025 年 11 月 30 日之后，只有数据库洞察的高级模式将支持执行计划和按需分析。
>
> 借助 CloudWatch 数据库洞察，您可以监控数据库实例集的数据库负载，并对性能进行大规模分析和故障排除。有关数据库洞察的更多信息，请参阅使用 CloudWatch 数据库洞察监控 Amazon RDS 数据库。有关定价信息，请参阅 Amazon CloudWatch 定价。

从 Amazon RDS 控制台中，您可以选择新版监控视图来查看数据库实例的性能详情和 CloudWatch 指标。

使用导航窗格中的性能详情选择新版监控视图

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在左侧导航窗格中，选择性能详情。
3. 选择一个数据库实例，以便查看性能详情控制面板，其中显示数据库实例的性能详情和 CloudWatch 指标。



使用性能详情创建自定义控制面板

> **Important**
>
> Amazon 已宣布性能详情的生命周期终止日期为：2025 年 11 月 30 日。在此日期之后，Amazon RDS 将不再支持性能详情控制台体验、灵活的保留期（1-24 个月）及其关联定价。性能详情 API 将继续存在，定价不变。性能详情 API 的费用将与 CloudWatch 数据库洞察的费用一起显示在您的 Amazon 账单中。
>
> 我们建议您在 2025 年 11 月 30 日之前，将使用性能详情付费套餐的所有数据库实例升级到数据库洞察的高级模式。有关升级到数据库洞察的高级模式的信息，请参阅开启适用于 Amazon RDS 的数据库洞察的高级模式。

如果您不采取任何行动，则使用性能详情的数据库实例将默认为使用数据库洞察的标准模式。在数据库洞察的标准模式下，您可能会在 7 天后无法访问性能数据历史记录，并且可能无法在 Amazon RDS 控制台中使用执行计划和按需分析功能。在 2025 年 11 月 30 日之后，只有数据库洞察的高级模式将支持执行计划和按需分析。

借助 CloudWatch 数据库洞察，您可以监控数据库实例集的数据库负载，并对性能进行大规模分析和故障排除。有关数据库洞察的更多信息，请参阅使用 CloudWatch 数据库洞察监控 Amazon RDS 数据库。有关定价信息，请参阅 Amazon CloudWatch 定价。

在新的监控视图中，您可以创建自定义控制面板，其中具有满足分析要求所需的指标。

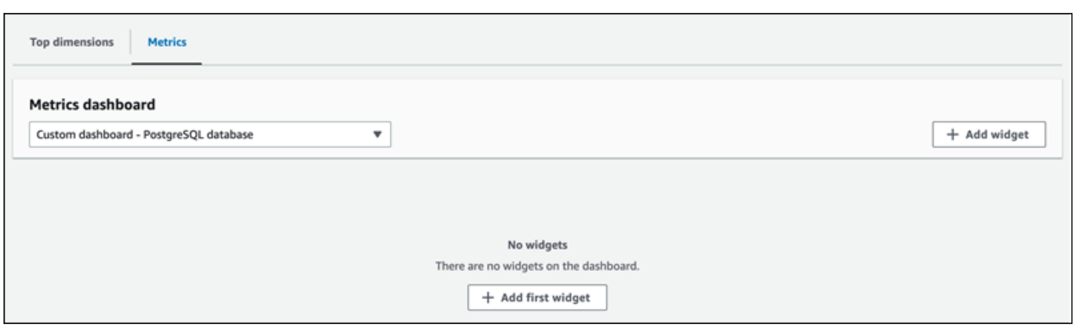
您可以通过为数据库实例选择性能详情和 CloudWatch 指标来创建自定义控制面板。您可以将此自定义控制面板用于 Amazon 账户中属于相同数据库引擎类型的其他数据库实例。

> **Note**
>
> 自定义控制面板可支持多达 50 个指标。

使用小组件设置菜单编辑或删除控制面板，并移动小组件窗口或调整其大小。

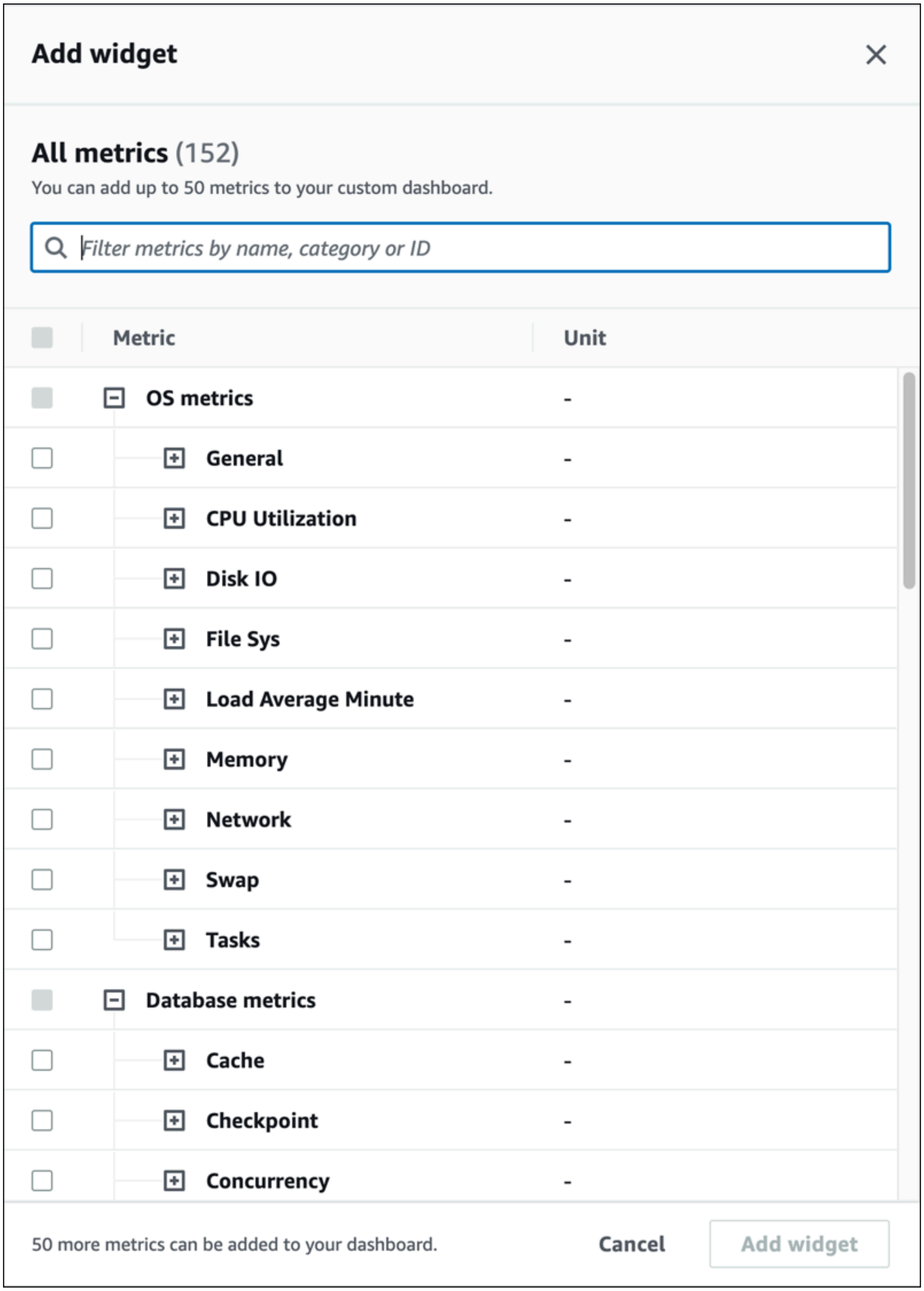
使用导航窗格中的性能详情创建自定义控制面板

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在左侧导航窗格中，选择性能详情。
3. 选择一个数据库实例。
4. 向下滚动到窗口中的指标选项卡。
5. 从下拉列表中选择自定义控制面板。以下显示了自定义控制面板的创建过程。



6. 选择添加小组件以打开添加小组件窗口。您可以在此窗口中打开和查看可用的操作系统（OS）指标、数据库指标和 CloudWatch 指标。

   以下示例显示了添加小组件窗口以及指标。



7. 选择要在控制面板中查看的指标，然后选择添加小组件。您可以使用搜索字段来查找特定的指标。

所选指标显示在您的控制面板上。

8. （可选）如果您要修改或删除控制面板，请选择小组件右上角的设置图标，然后在菜单中选择以下操作之一。

   - 编辑 - 修改窗口中的指标列表。为控制面板选择指标后，选择更新小组件。
   - 删除 - 删除小组件。在确认窗口中选择删除。

使用性能详情选择预配置控制面板

Important

Amazon 已宣布性能详情的生命周期终止日期为：2025 年 11 月 30 日。在此日期之后，Amazon RDS 将不再支持性能详情控制台体验、灵活的保留期（1-24 个月）及其关联定价。性能详情 API 将继续存在，定价不变。性能详情 API 的费用将与 CloudWatch 数据库洞察的费用一起显示在您的 Amazon 账单中。

我们建议您在 2025 年 11 月 30 日之前，将使用性能详情付费套餐的所有数据库实例升级到数据库洞察的高级模式。有关升级到数据库洞察的高级模式的信息，请参阅开启适用于 Amazon RDS 的数据库洞察的高级模式。

如果您不采取任何行动，则使用性能详情的数据库实例将默认为使用数据库洞察的标准模式。在数据库洞察的标准模式下，您可能会在 7 天后无法访问性能数据历史记录，并且可能无法在 Amazon RDS 控制台中使用执行计划和按需分析功能。在 2025 年 11 月 30 日之后，只有数据库洞察的高级模式将支持执行计划和按需分析。

借助 CloudWatch 数据库洞察，您可以监控数据库实例集的数据库负载，并对性能进行大规模分析和故障排除。有关数据库洞察的更多信息，请参阅使用 CloudWatch 数据库洞察监控 Amazon RDS 数据库。有关定价信息，请参阅 Amazon CloudWatch 定价。

您可以使用预配置控制面板查看最常用的指标。此控制面板有助于诊断数据库引擎的性能问题，并将平均恢复时间从几小时缩短到几分钟。

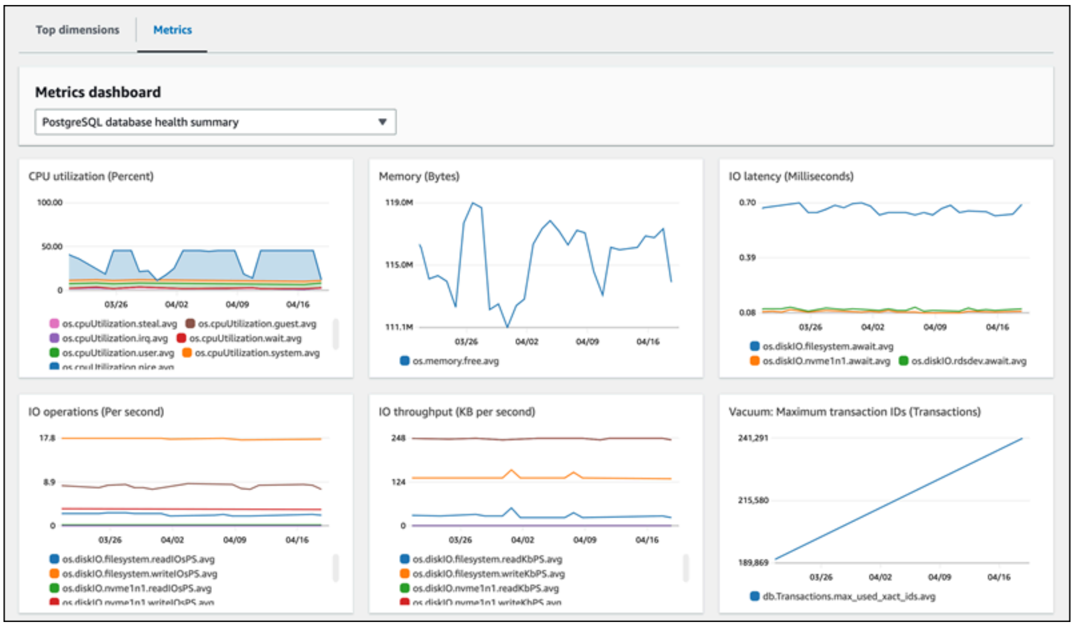
Note

无法编辑此控制面板。

使用导航窗格中的性能详情选择预配置控制面板

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在左侧导航窗格中，选择性能详情。
3. 选择一个数据库实例。
4. 向下滚动到窗口中的指标选项卡
5. 从下拉列表中选择预配置的控制面板。

   您可以在控制面板中查看数据库实例的指标。以下示例显示了一个预配置指标控制面板。



使用 Amazon CloudWatch 监控 Amazon RDS 指标

Amazon CloudWatch 是一个指标存储库。此存储库可从 Amazon RDS 收集原始数据，并将数据处理为易读的近乎实时的指标。有关发送到 CloudWatch 的 Amazon RDS 指标的完整列表，请参阅 Amazon RDS 的指标参考。

要大规模分析数据库性能并对其进行故障排除，请使用 CloudWatch 数据库洞察。

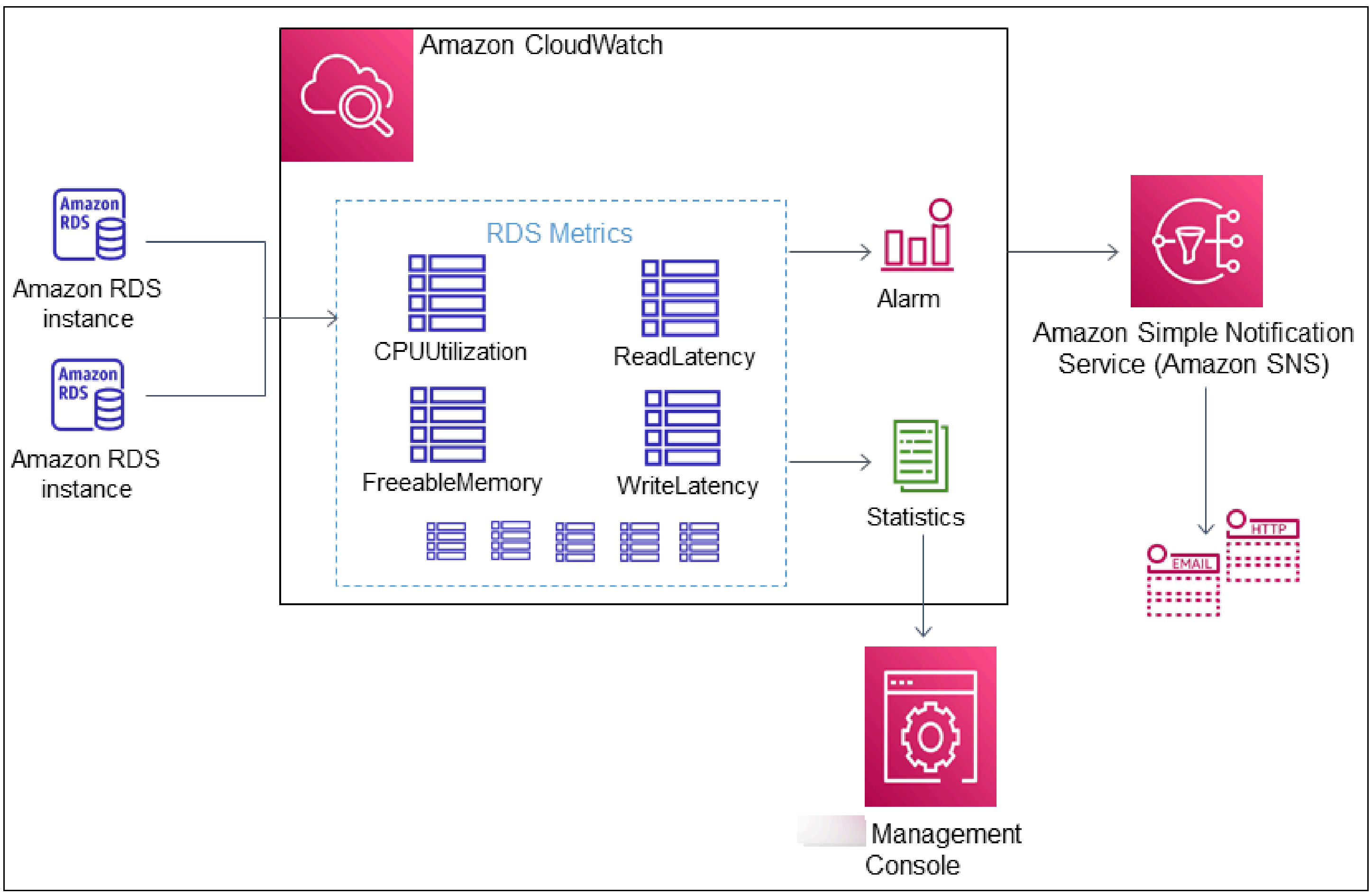
主题

Amazon RDS 和 Amazon CloudWatch 概述

默认情况下，以 1 分钟为间隔自动将 Amazon RDS 指标数据发送到 CloudWatch。例如，`CPUUtilization` 指标记录了一段时间内数据库实例的 CPU 使用率百分比。时间段为 60 秒（1 分钟）的数据点可用 15 天。这意味着您能够访问历史信息，了解您的 Web 应用程序或服务的执行情况。

现在，您可以将 Performance Insights 指标控制面板从 Amazon RDS 导出到 Amazon CloudWatch。您可以将预配置或自定义的指标控制面板导出为新的控制面板，也可以将其添加到现有的 CloudWatch 控制面板中。导出的控制面板可在 CloudWatch 控制台中查看。有关如何将 Performance Insights 指标控制面板导出到 CloudWatch 的更多信息，请参阅将 Performance Insights 指标导出到 CloudWatch。

如下图所示，您可以为 CloudWatch 指标设置警报。例如，您可能会创建一个警报，在实例的 CPU 使用率超过 70% 时发出信号。您可以配置 Amazon Simple Notification Service 以在超过阈值时向您发送电子邮件。



Amazon RDS 向 Amazon CloudWatch 发布以下类型的指标：

- RDS 数据库实例的指标

  有关这些指标的表格，请参阅 Amazon RDS 的 Amazon CloudWatch 指标。
- Performance Insights 指标

  有关这些指标的表格，请参阅 Amazon RDS 性能详情的 Amazon CloudWatch 指标 和 性能详情计数器指标。
- 增强监控指标（发布到 Amazon CloudWatch Logs）

  有关这些指标的表格，请参阅 增强监控中的操作系统指标。
- 您的 Amazon Web Services 账户中 Amazon RDS 服务配额的用量指标

  有关这些指标的表格，请参阅 Amazon RDS 的 Amazon CloudWatch 用量指标。有关 Amazon RDS 配额的更多信息，请参阅 Amazon RDS 的配额和限制。

有关 CloudWatch 的更多信息，请参阅 Amazon CloudWatch 用户指南 中的什么是 Amazon CloudWatch？。有关 CloudWatch 指标保留的更多信息，请参阅指标保留。

在 CloudWatch 控制台和 Amazon CLI 中查看数据库实例指标

在下文中，您可以了解有关如何使用 CloudWatch 查看数据库实例指标的详细信息。有关使用 CloudWatch Logs 实时监控数据库实例的操作系统指标的信息，请参阅使用增强监控来监控操作系统指标。

在使用 Amazon RDS 资源时，Amazon RDS 每分钟向 Amazon CloudWatch 发送一次指标和维度。

现在，您可以将 Performance Insights 指标控制面板从 Amazon RDS 导出到 Amazon CloudWatch，然后在 CloudWatch 控制台中查看这些指标。有关如何将 Performance Insights 指标控制面板导出到 CloudWatch 的更多信息，请参阅将 Performance Insights 指标导出到 CloudWatch。

使用以下过程在 CloudWatch 控制台和 CLI 中查看 Amazon RDS 的指标。

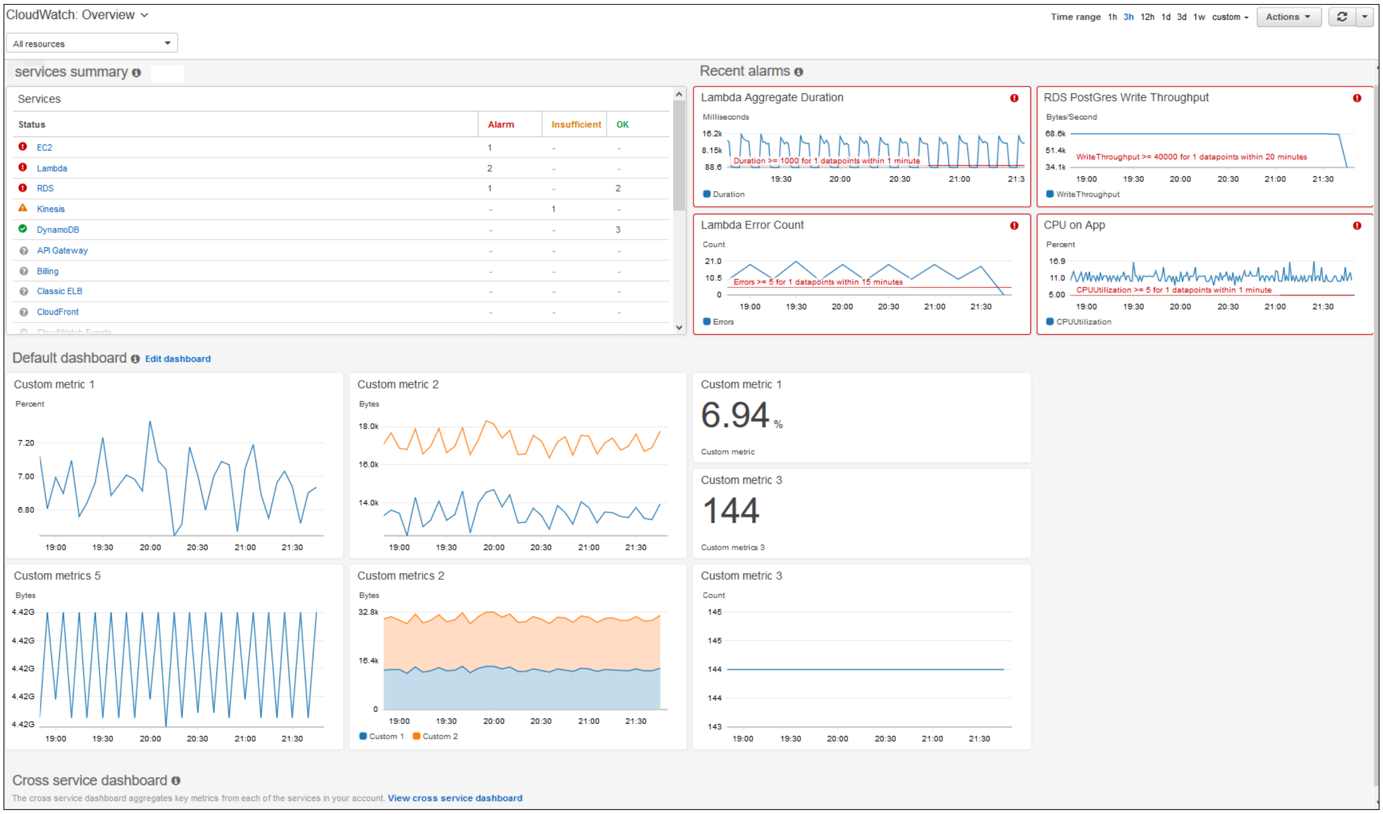
控制台

使用 Amazon CloudWatch 控制台查看指标

指标的分组首先依据服务命名空间，然后依据每个命名空间内的各种维度组合。

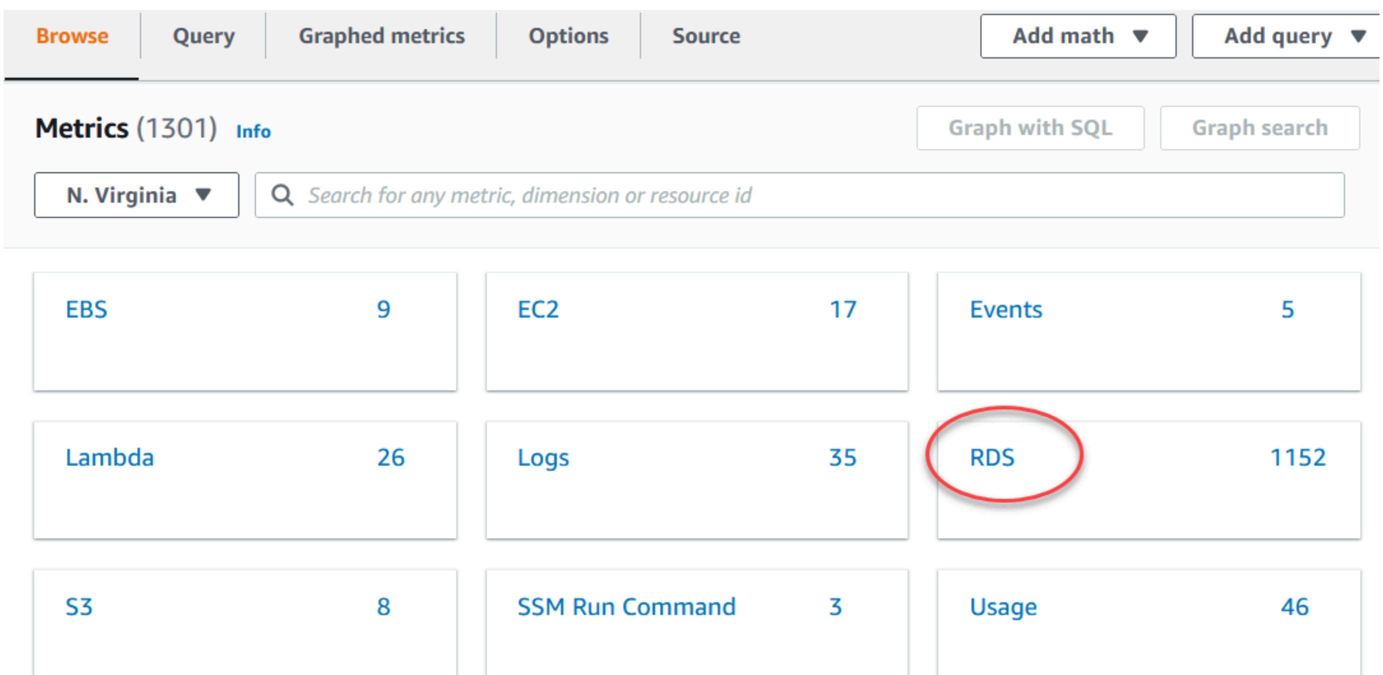
1. 访问 https://console.aws.amazon.com/cloudwatch/ 打开 CloudWatch 控制台。

   将显示 CloudWatch 概览主页。



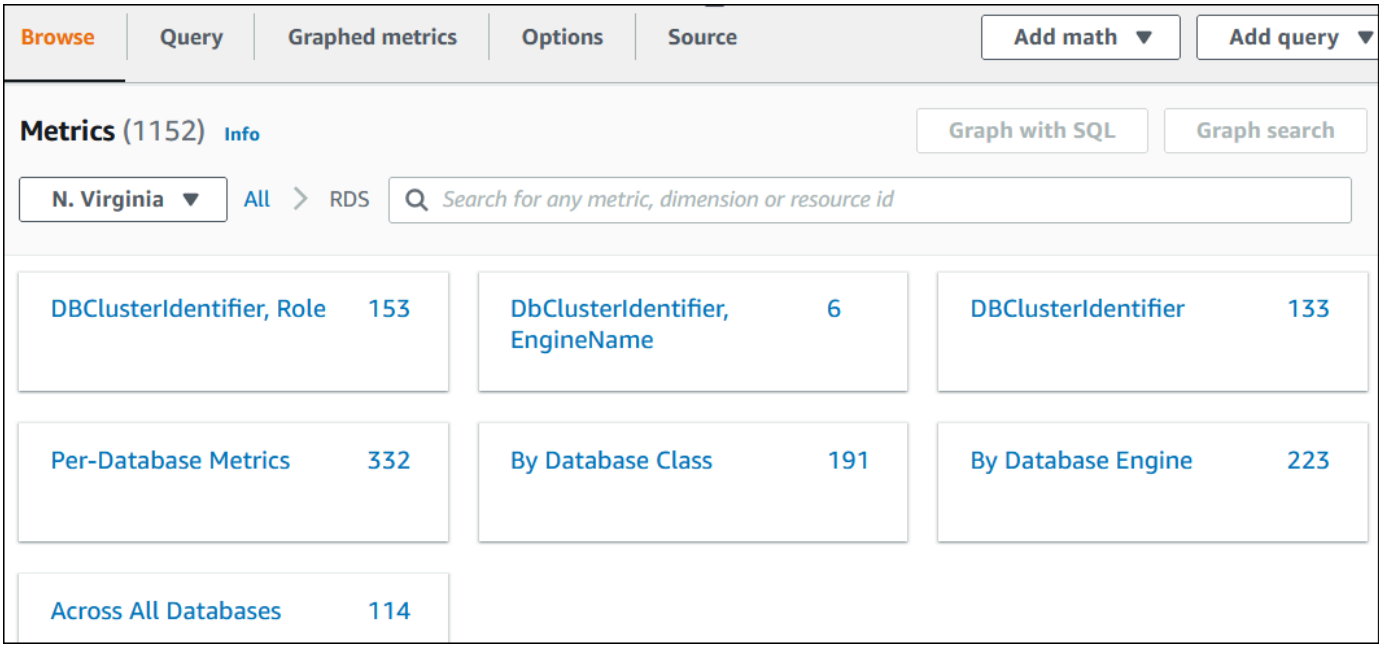
2. 如果需要，更改 Amazon Web Services 区域。从导航栏中，选择您的 Amazon 资源所在的 Amazon Web Services 区域。有关更多信息，请参阅[区域和端点](#)。

3. 在导航窗格中，选择 Metrics（指标），然后选择 All metrics（所有指标）。

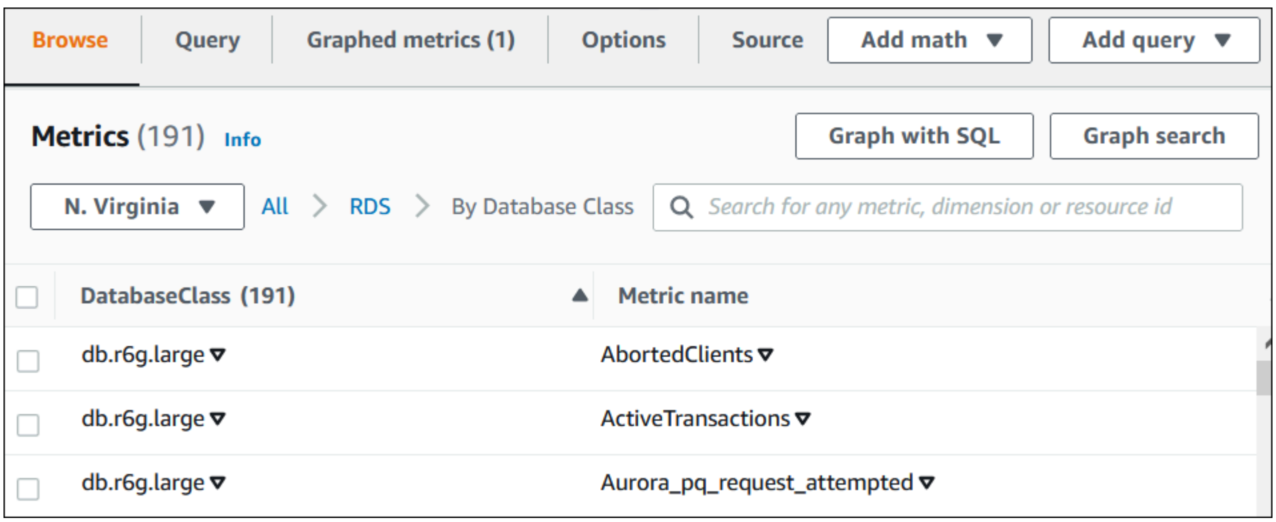


4. 向下滚动并选择 RDS 指标命名空间。

该页面将显示 Amazon RDS 维度。有关这些维度的说明，请参阅 Amazon RDS 的 Amazon CloudWatch 维度。



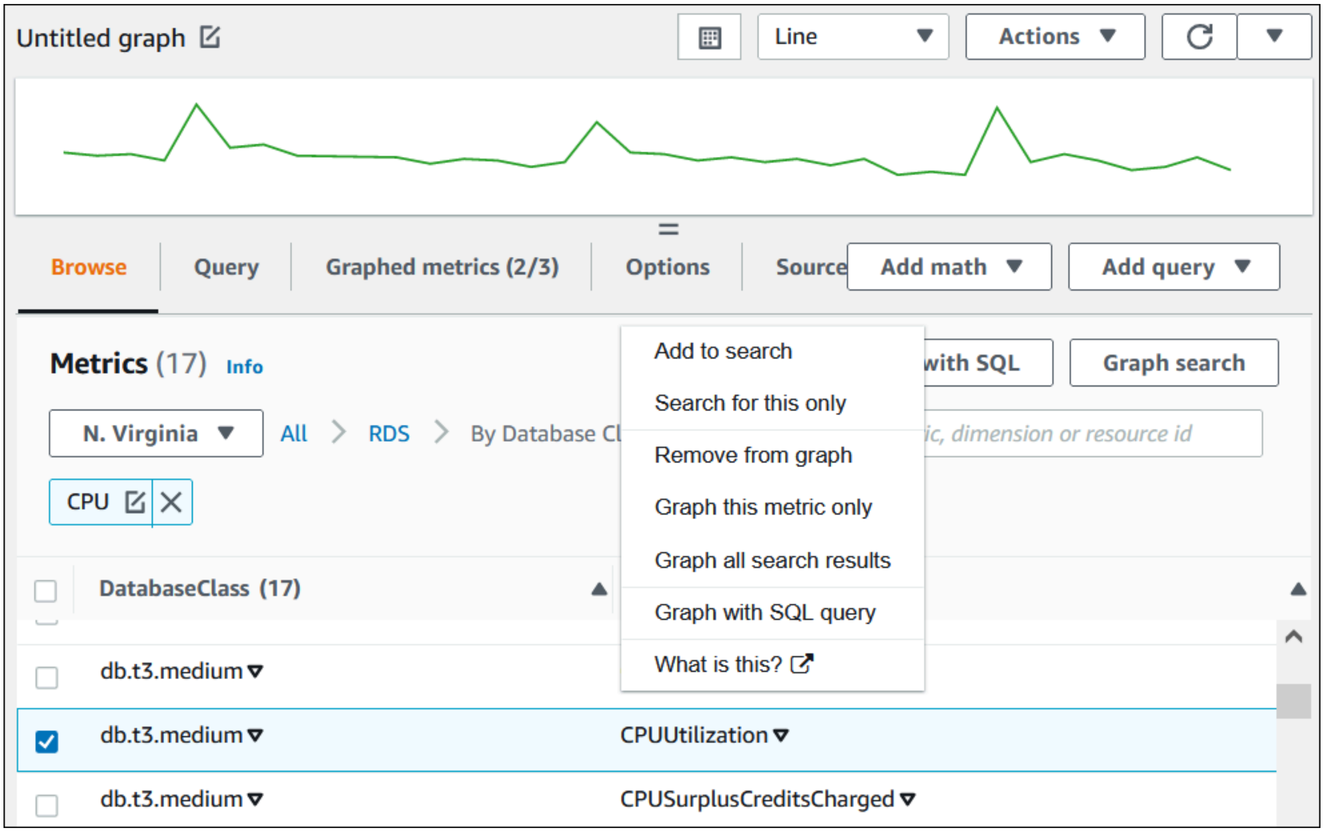
5. 选择指标维度，例如，按数据库类。



6. 执行以下任一操作：

- 要对指标进行排序，请使用列标题。
- 要为指标绘制图表，请选中该指标旁的复选框。
- 要按资源进行筛选，请选择资源 ID，然后选择添加到搜索。
- 要按指标进行筛选，请选择指标名称，然后选择添加到搜索。

以下示例筛选 db.t3.medium 类并绘制 CPUUtilization 指标。



Amazon CLI

若要通过使用 Amazon CLI 获取指标信息，请使用 CloudWatch 命令 `list-metrics`。在以下示例中，您将列出 AWS/RDS 命名空间中的所有指标。

```
aws cloudwatch list-metrics --namespace AWS/RDS
```

要获取指标数据，请使用命令 `get-metric-data`。

以下示例以 5 分钟为间隔，获取实例 `my-instance` 在特定 24 小时时段内的 `CPUUtilization` 统计数据。

使用以下内容创建 JSON 文件 `CPU_metric.json`。

```
{
   "StartTime" : "2023-12-25T00:00:00Z",
   "EndTime" : "2023-12-26T00:00:00Z",
   "MetricDataQueries" : [{
```

```
      "Id" : "cpu",
      "MetricStat" : {
     "Metric" : {
         "Namespace" : "AWS/RDS",
         "MetricName" : "CPUUtilization",
         "Dimensions" : [{ "Name" : "DBInstanceIdentifier" , "Value" : my-instance}]
     },
        "Period" : 360,
        "Stat" : "Minimum"
      }
    }]
}
```

Example

对于 Linux、macOS 或 Unix：

```
aws cloudwatch get-metric-data \
    --cli-input-json file://CPU_metric.json
```

对于 Windows：

```
aws cloudwatch get-metric-data ^
     --cli-input-json file://CPU_metric.json
```

示例输出如下：

```
{
    "MetricDataResults": [
        {
            "Id": "cpu",
            "Label": "CPUUtilization",
            "Timestamps": [
                "2023-12-15T23:48:00+00:00",
                "2023-12-15T23:42:00+00:00",
                "2023-12-15T23:30:00+00:00",
                "2023-12-15T23:24:00+00:00",
                ...
            ],
            "Values": [
                13.299778337027714,
                13.677507543049558,
                14.24976250395827,
```

```
                13.02521708695145,
                ...
            ],
            "StatusCode": "Complete"
        }
    ],
    "Messages": []
}
```

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的获取指标统计数据。

将 Performance Insights 指标导出到 CloudWatch

Performance Insights 允许您将数据库实例的预配置或自定义指标控制面板导出到 Amazon CloudWatch。您可以将指标控制面板导出为新的控制面板，也可以将其添加到现有的 CloudWatch 控制面板中。当您选择将控制面板添加到现有 CloudWatch 控制面板时，可以创建标题标签，以便指标单独显示在 CloudWatch 控制面板的某个部分中。

您还可以在 CloudWatch 控制台中查看导出的指标控制面板。如果您在导出 Performance Insights 指标控制面板后向其添加了新指标，则必须再次导出此控制面板，然后才能在 CloudWatch 控制台中查看新指标。

您还可以在 Performance Insights 控制面板中选择一个指标小组件，然后在 CloudWatch 控制台中查看指标数据。

有关在 CloudWatch 控制台中查看指标的更多信息，请参阅 在 CloudWatch 控制台和 Amazon CLI 中查看数据库实例指标。

在以下各节中，将性能详情指标导出到 CloudWatch 作为新的或现有控制面板，并在 CloudWatch 中查看性能详情指标。

主题

- 将 Performance Insights 指标作为新的控制面板导出到 CloudWatch
- 向现有的 CloudWatch 控制面板添加 Performance Insights 指标
- 在 CloudWatch 中查看 Performance Insights 指标小组件

将 Performance Insights 指标作为新的控制面板导出到 CloudWatch

从 Performance Insights 控制面板中选择预配置或自定义的指标控制面板，然后将其作为新控制面板导出到 CloudWatch。您可以在 CloudWatch 控制台中查看这些导出的控制面板。

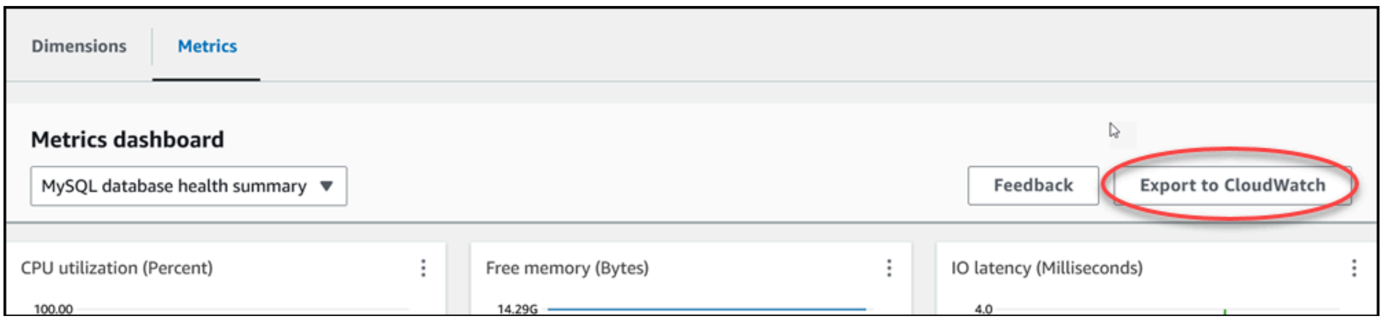
将 Performance Insights 指标控制面板作为新控制面板导出到 CloudWatch

1. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在左侧导航窗格中，选择 Performance Insights。
3. 选择一个数据库实例。

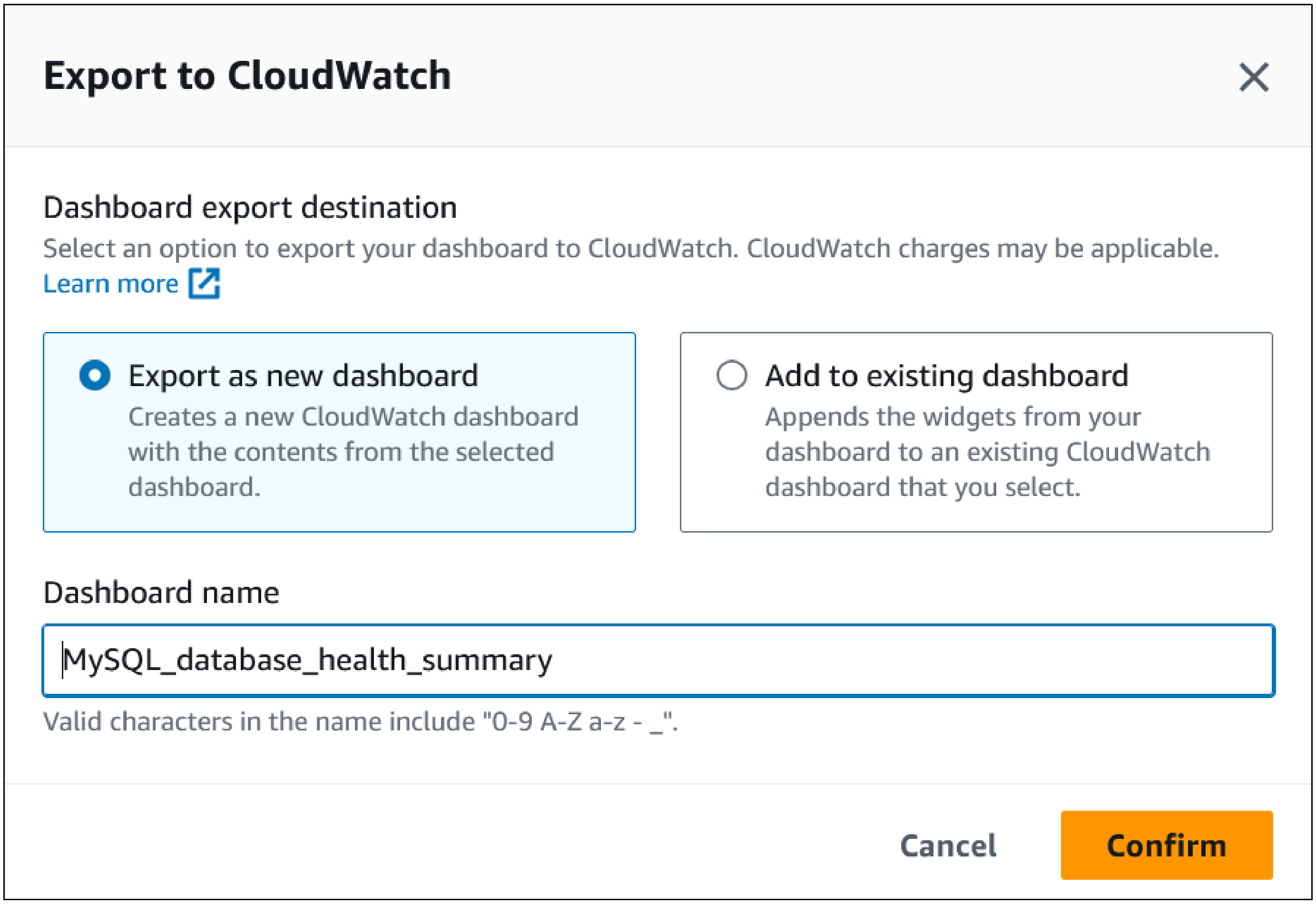
   将显示该数据库实例的 Performance Insights 控制面板。
4. 向下滚动并选择指标。

   默认情况下，会显示包含 Performance Insights 指标的预配置控制面板。
5. 选择预配置或自定义控制面板，然后选择导出到 CloudWatch。

   随后会显示导出到 CloudWatch 窗口。

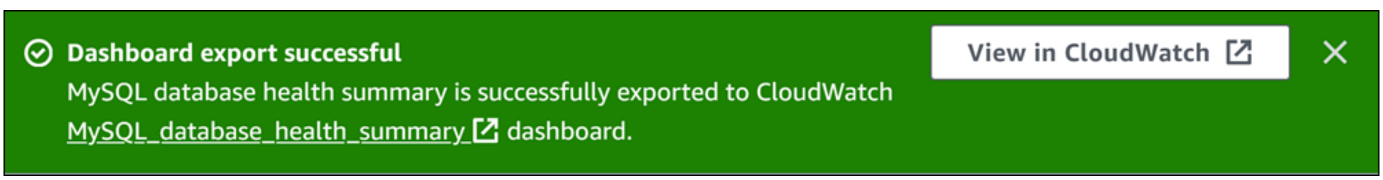


6. 选择导出为新控制面板。



7. 在控制板名称字段中输入新控制面板的名称，然后选择确认。

   控制面板导出成功后，横幅会显示一条消息。



8. 选择横幅中的链接或在 CloudWatch 中查看，即可在 CloudWatch 控制台中查看指标控制面板。

向现有的 CloudWatch 控制面板添加 Performance Insights 指标

向现有的 CloudWatch 控制面板添加预配置或自定义指标控制面板。您可以向指标控制面板添加标签，使其单独显示在 CloudWatch 控制面板的某个部分中。

将指标导出到现有的 CloudWatch 控制面板

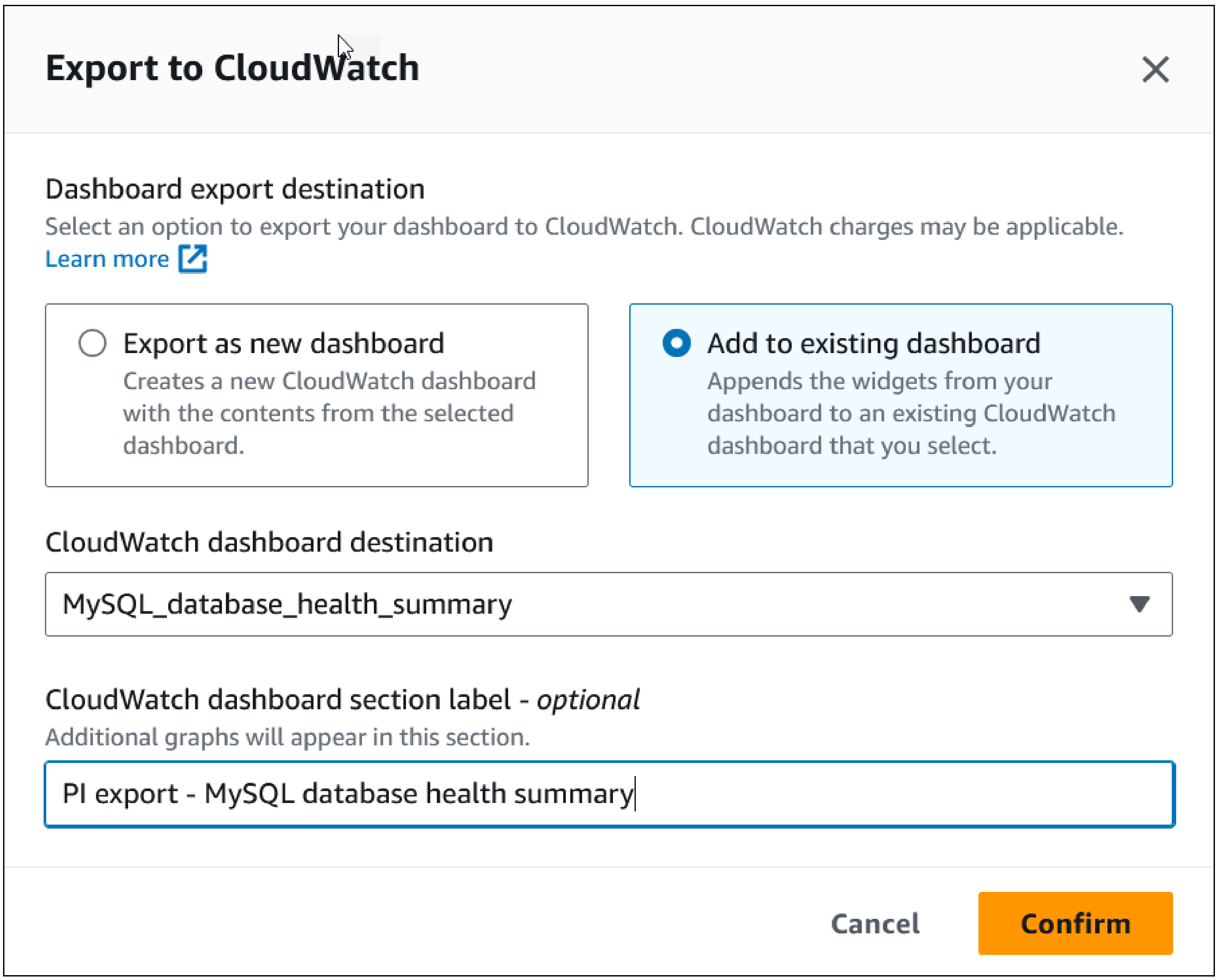
1. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。

2. 在左侧导航窗格中，选择 Performance Insights。
3. 选择一个数据库实例。

   将显示该数据库实例的 Performance Insights 控制面板。
4. 向下滚动并选择指标。

   默认情况下，会显示包含 Performance Insights 指标的预配置控制面板。
5. 选择预配置或自定义控制面板，然后选择导出到 CloudWatch。

   随后会显示导出到 CloudWatch 窗口。
6. 选择添加到现有控制面板。



7. 指定控制面板目标和标签，然后选择确认。

- CloudWatch 控制面板目标 - 选择现有的 CloudWatch 控制面板。
- CloudWatch 控制面板部分标签 - 可选 - 输入要在 CloudWatch 控制面板的此部分中显示的 Performance Insights 指标的名称。

   控制面板导出成功后，横幅会显示一条消息。

8. 选择横幅中的链接或在 CloudWatch 中查看，即可在 CloudWatch 控制台中查看指标控制面板。

在 CloudWatch 中查看 Performance Insights 指标小组件

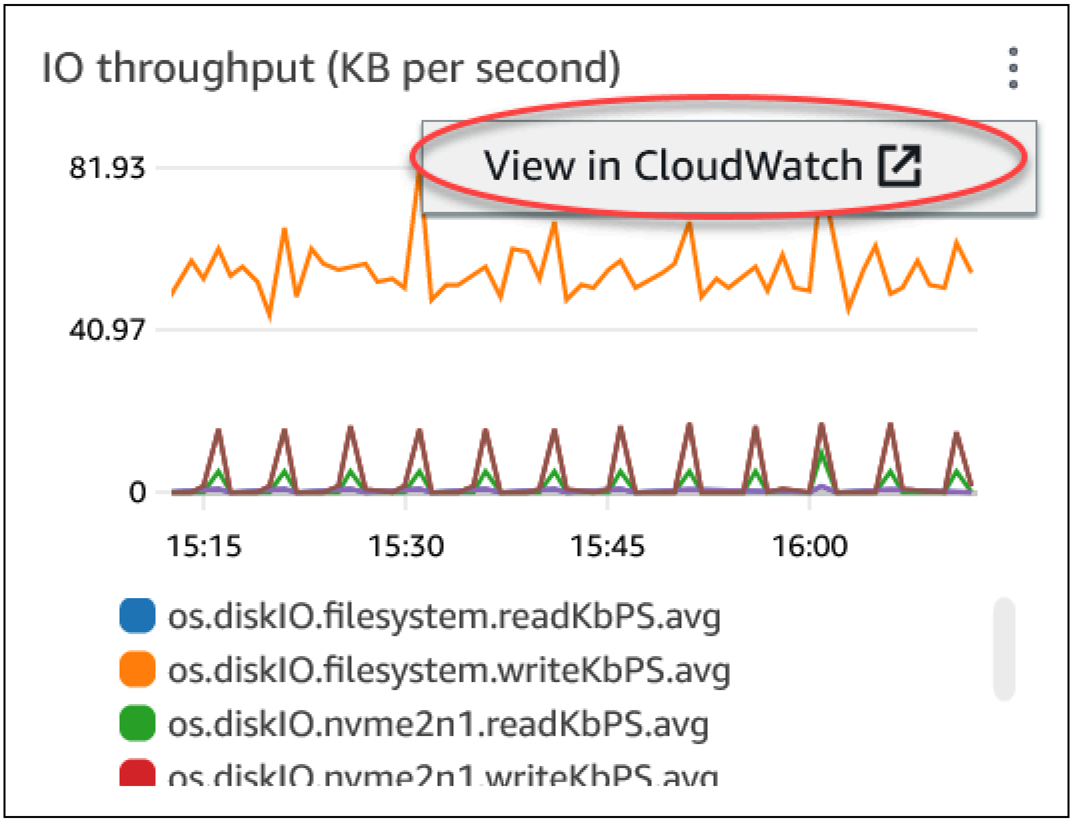
在 Amazon RDS Performance Insights 控制面板中选择一个 Performance Insights 指标小组件，然后在 CloudWatch 控制台中查看指标数据。

导出指标小组件并在 CloudWatch 控制台中查看指标数据

1. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在左侧导航窗格中，选择 Performance Insights。
3. 选择一个数据库实例。

   将显示该数据库实例的 Performance Insights 控制面板。
4. 向下滚动到指标。

   默认情况下，会显示包含 Performance Insights 指标的预配置控制面板。
5. 选择一个指标小组件，然后在菜单中选择在 CloudWatch 中查看。



指标数据显示在 CloudWatch 控制台中。

创建 CloudWatch 警报以监控 Amazon RDS

您可以创建在警报改变状态时发送 Amazon SNS 消息的 CloudWatch 警报。警报会监控您指定的时间段内的某个指标。此外，警报会根据指标值在多个时间段内对比给定阈值的情况执行一项或多项操作。操作是向 Amazon SNS 主题或 Amazon EC2 Auto Scaling 策略发送的通知。

警报只会调用操作进行持续的状态变更。CloudWatch 警报不会仅仅因为处于特定状态而调用操作。该状态必须改变并在指定数量的时间段内一直保持。

您可以使用 CloudWatch 控制台中的 DB_PERF_INSIGHTS 指标数学函数来查询 Amazon RDS 以获取性能详情计数器指标。DB_PERF_INSIGHTS 函数还包括以亚分钟为间隔的 DBLoad 指标。您可以根据这些指标设置 CloudWatch 警报。

有关如何创建警报的更多详细信息，请参阅针对 Amazon 数据库中的性能详情计数器指标创建警报。

使用 Amazon CLI 设置警报

- 调用 `put-metric-alarm`。有关更多信息，请参阅 Amazon CLI Command Reference。

使用 CloudWatch API 设置警报

- 调用 `PutMetricAlarm`。有关更多信息，请参阅 Amazon CloudWatch API 参考。

有关设置 Amazon SNS 主题和创建告警的更多信息，请参阅使用 Amazon CloudWatch 告警。

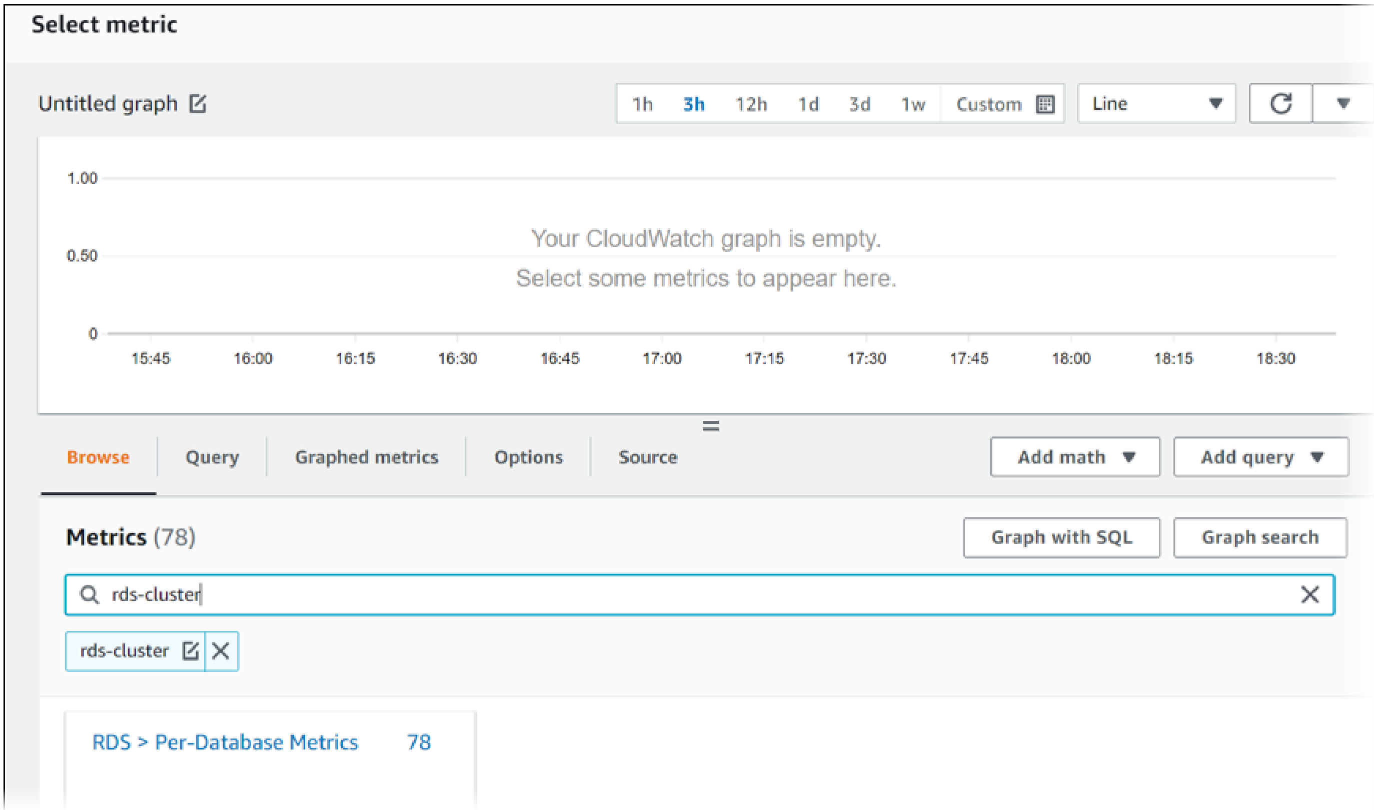
教程：为 Amazon RDS 的多可用区数据库集群副本滞后创建 Amazon CloudWatch 警报

您可以创建 Amazon CloudWatch 告警，使其在多可用区数据库集群的副本滞后超过阈值时发送 Amazon SNS 消息。告警会在您指定的时间范围内监控 `ReplicaLag` 指标。操作是向 Amazon SNS 主题或 Amazon EC2 Auto Scaling 策略发送的通知。

为多可用区数据库集群副本滞后设置 CloudWatch 告警

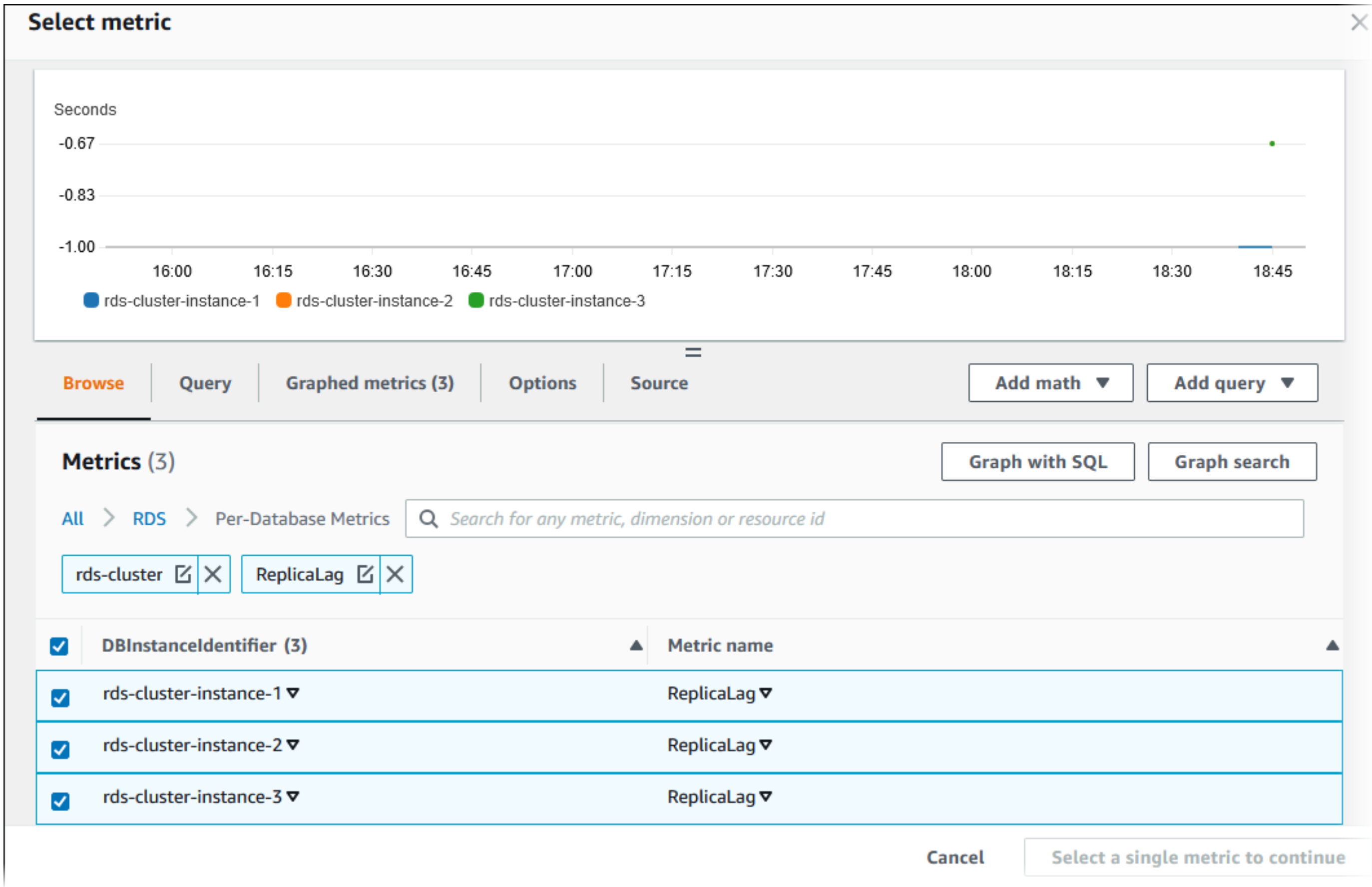
1. 登录 Amazon Web Services Management Console 并打开 CloudWatch 控制台，网址为 https://console.aws.amazon.com/cloudwatch/。
2. 在导航窗格中，依次选择 Alarms（警报）和 All alarms（所有警报）。
3. 选择Create alarm（创建警报）。
4. 在 Specify metric and conditions (指定指标和条件) 页面上，选择 Select metric (选择指标)。
5. 在搜索框中，输入多可用区数据库集群的名称，然后按 Enter。

   下图显示 Select metric（选择指标）页面，其中输入了名为 `rds-cluster` 的多可用区数据库集群。



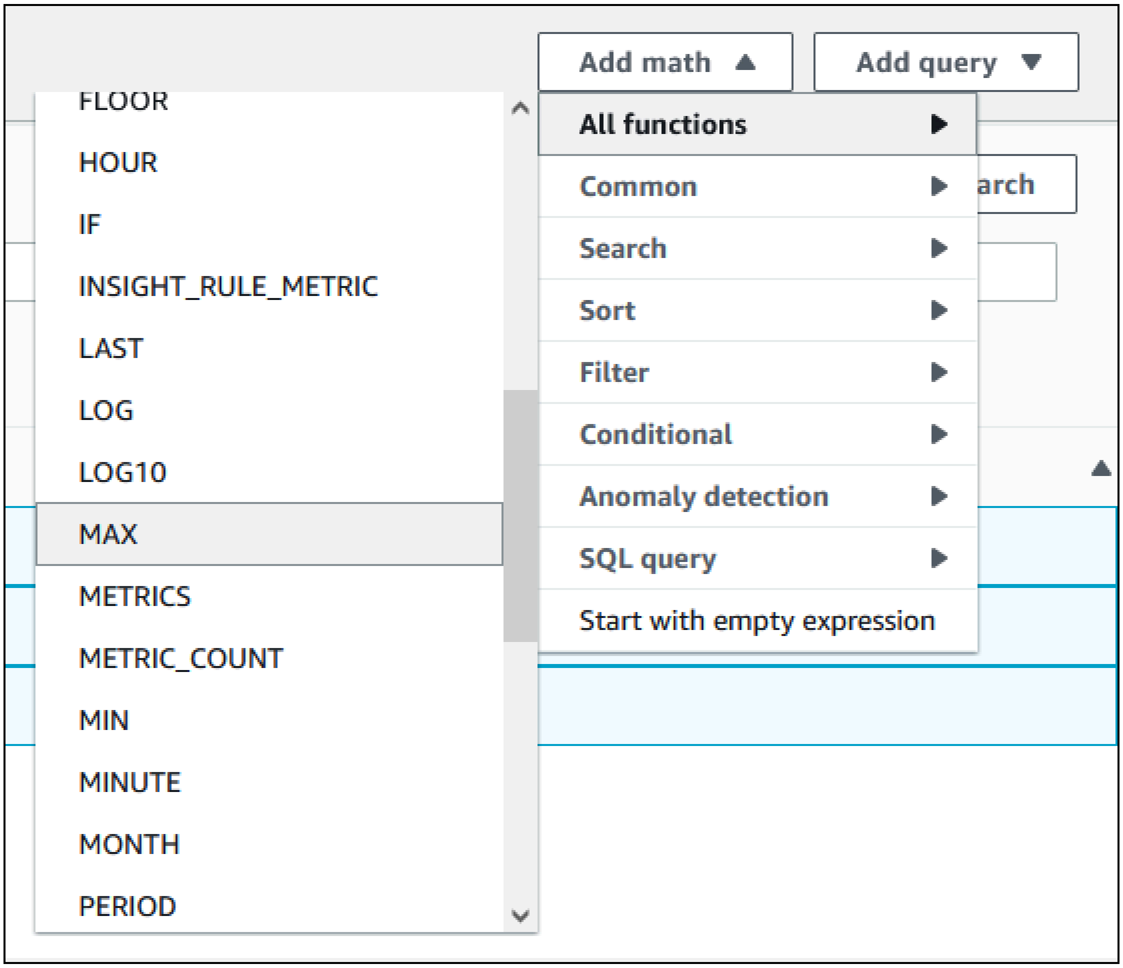
6. 依次选择 RDS、Per-Database Metrics（每个数据库的指标）。
7. 在搜索框中，输入 **ReplicaLag** 并按 Enter，然后选择数据库集群中的每个数据库实例。

   下图显示 Select metric（选择指标）页面，及为 ReplicaLag 指标选择的数据库实例。



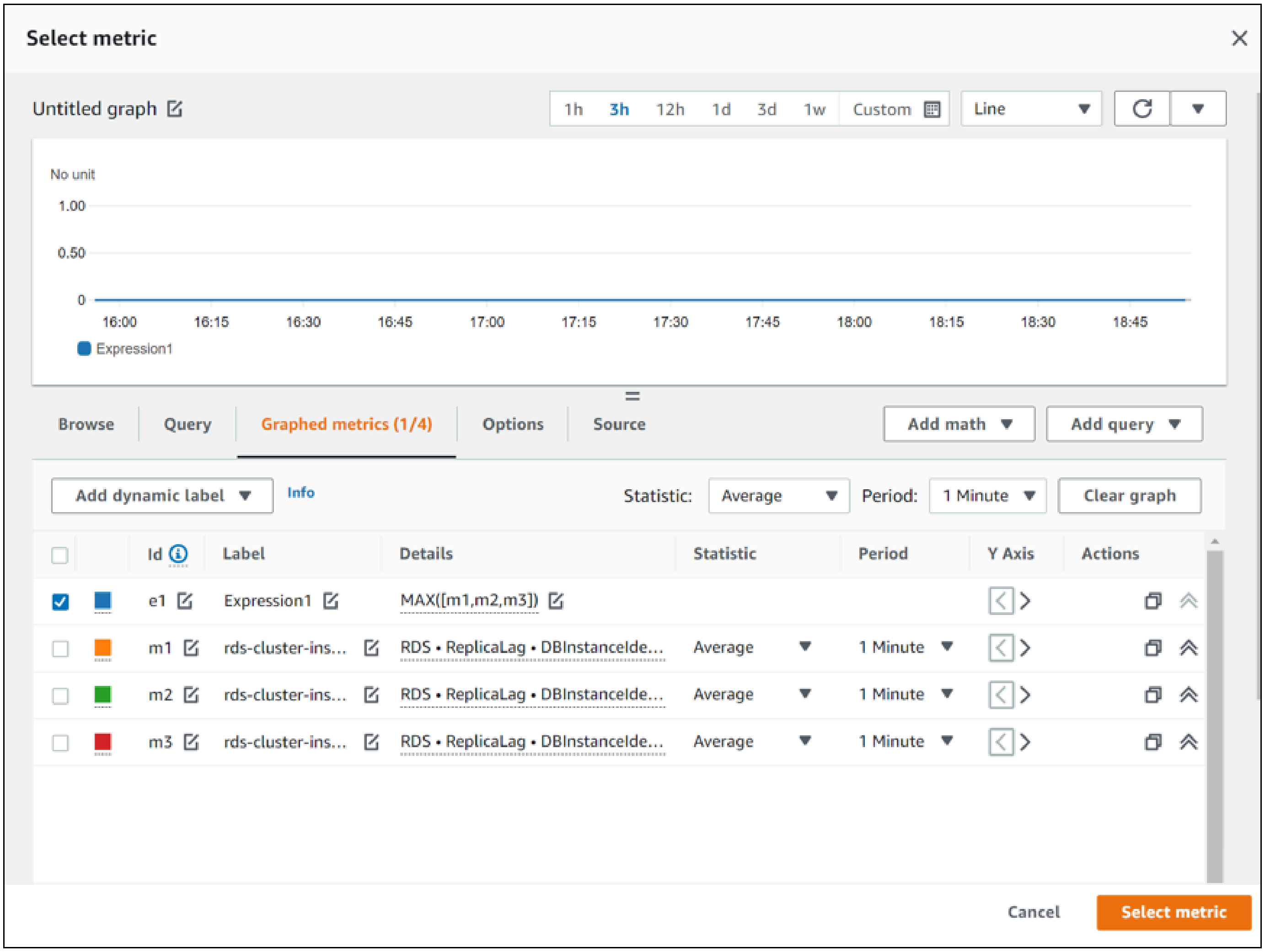
此告警考虑多可用区数据库集群中所有三个数据库实例的副本滞后。任何数据库实例超过阈值时，该告警都会响应。它使用一个数学表达式，返回三个指标的最大值。首先按指标名称排序，然后选择所有三个 RepliaLag 指标。

8. 从 Add math（添加数学表达式）中，依次选择 All functions（所有函数）、MAX。



9. 选择 Graphed metrics（绘制的指标）选项卡，然后将 Expression1（表达式 1）的详细信息编辑为 **`MAX([m1,m2,m3])`**。
10. 对于所有三个 ReplicaLag 指标，将 Period（周期）更改为 1 minute（1 分钟）。
11. 清除所有指标的选择， Expression1（表达式 1）除外。

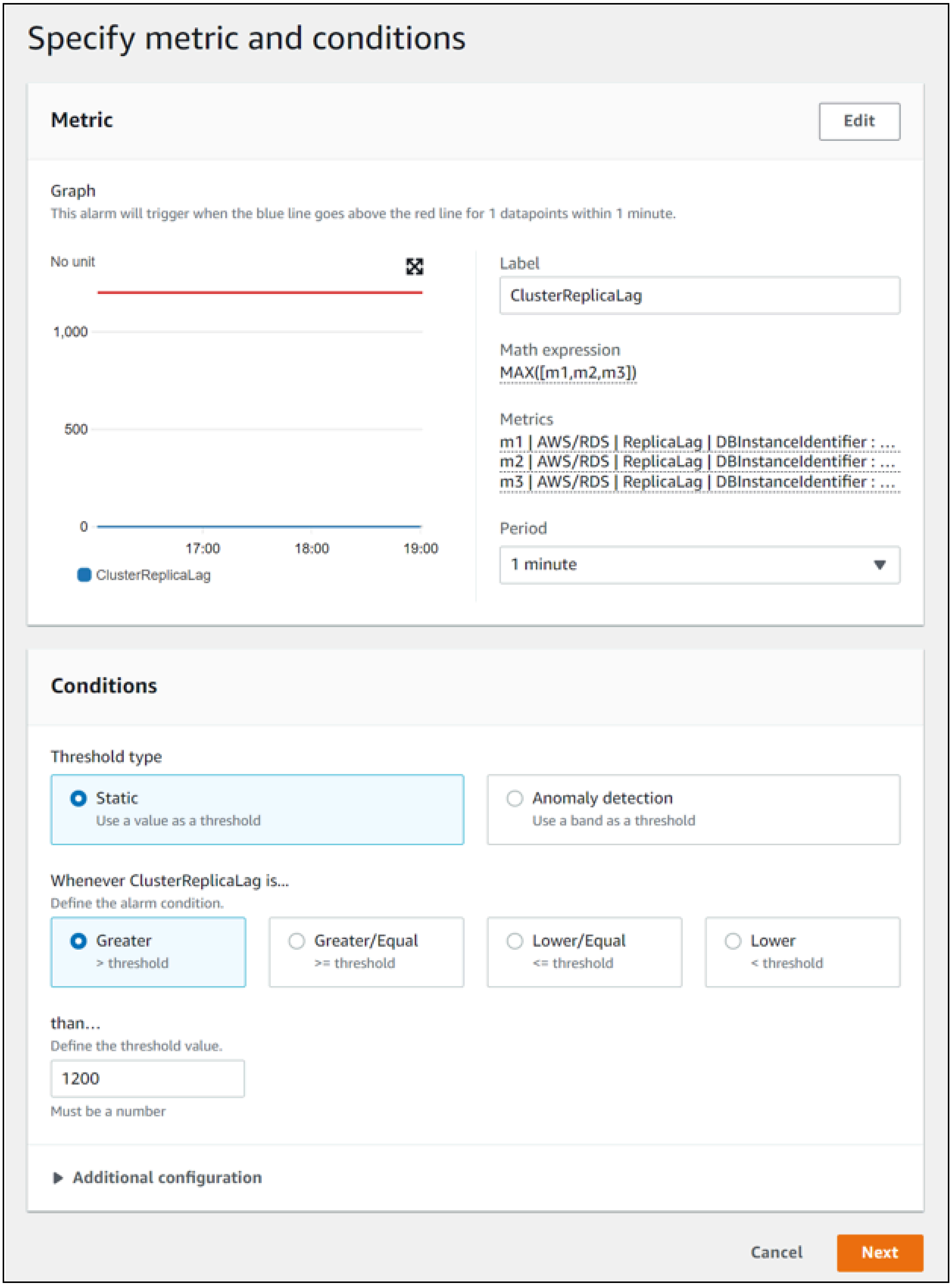
    Select metric（选择指标）页面应类似于以下图像。



12. 选择选择指标。

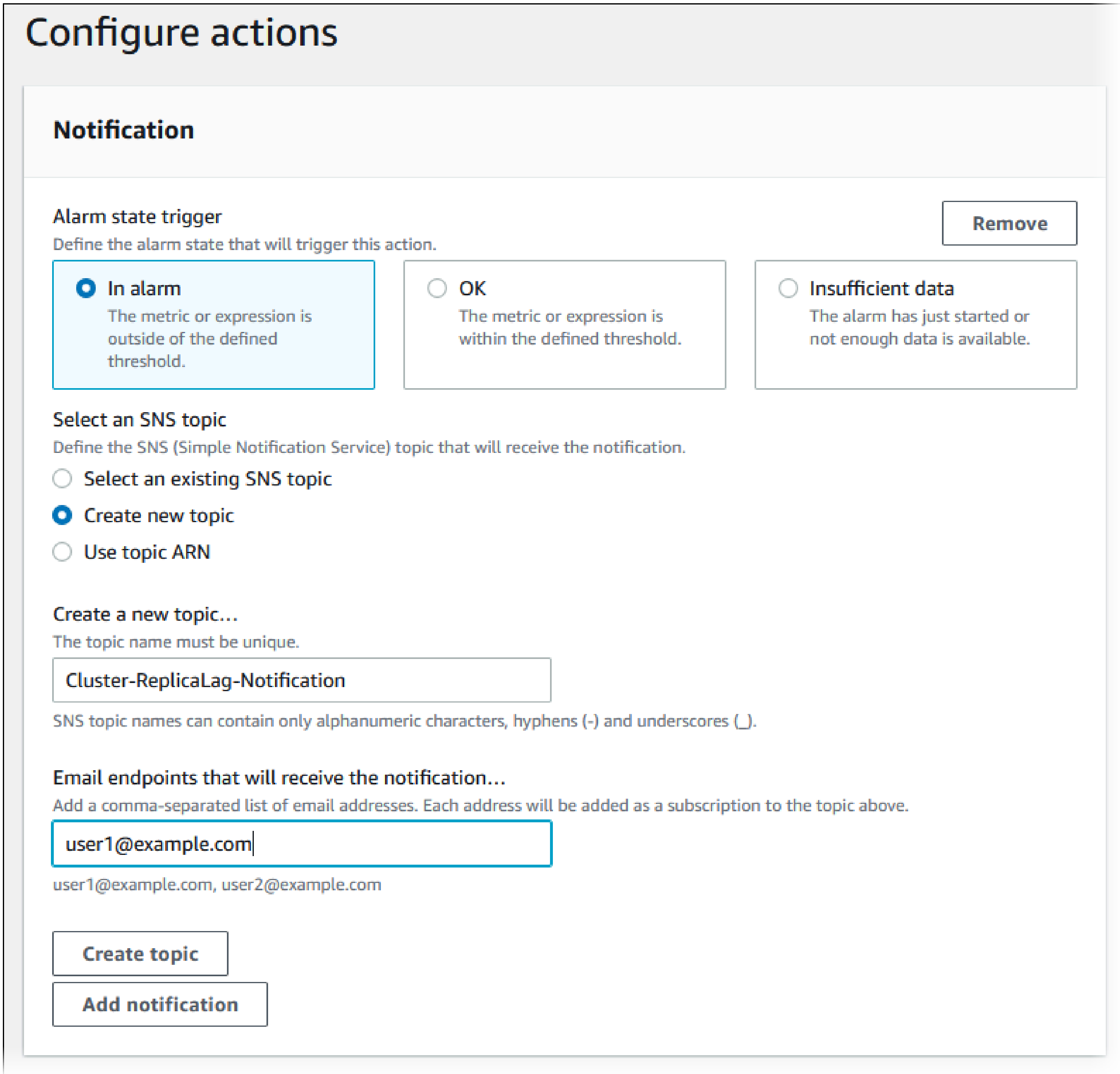
13. 在 Specify metric and conditions（指定指标和条件）页面中，将标签更改为有意义的名称（例如 **ClusterReplicaLag**），然后在 Define the threshold value（定义阈值）中输入秒数。对于本教程，请输入 **1200** 秒（20 分钟）。您可以根据工作负载要求调整该值。

    Specify metric and conditions（指定指标和条件）页面应类似于以下图像。



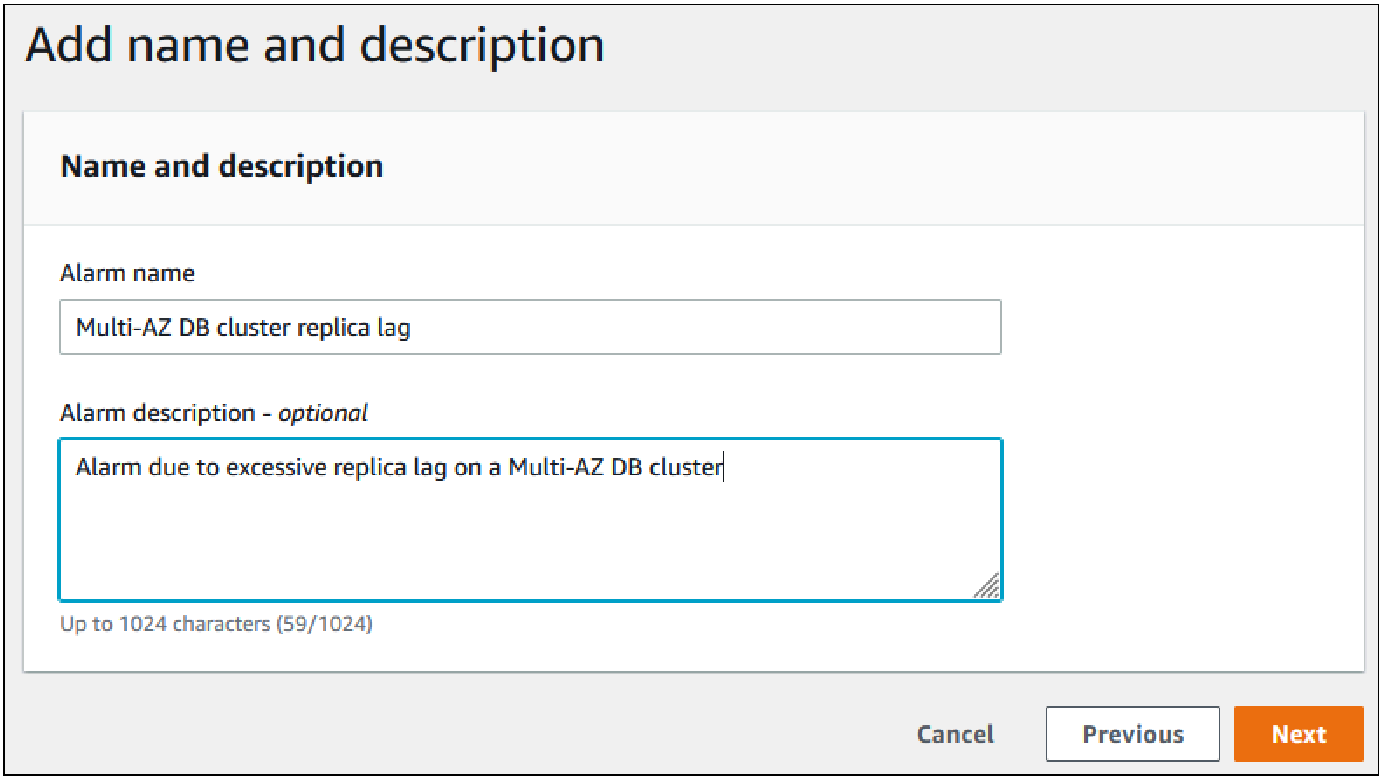
14. 选择 Next（下一步），随即显示 Configure actions（配置操作）页面。

15. 保持选中 In alarm（处于告警中），选择 Create new topic（创建新主题），然后输入主题名称和有效的电子邮件地址。



16. 选择 Create topic（创建主题），然后选择 Next（下一步）。

17. 在 Add name and description（添加名称和说明）页面中，输入 Alarm name（告警名称）和 Alarm description（告警说明），然后选择 Next（下一步）。



18. 预览您将在 Preview and create（预览和创建）页面上创建的告警，然后选择 Create alarm（创建告警）。

使用 CloudWatch 数据库洞察监控 Amazon RDS 数据库

使用数据库洞察监控 Amazon RDS 数据库实例的实例集的数据库负载（DB Load）。数据库负载可衡量数据库中会话活动的级别。您可以使用数据库洞察来大规模分析 Amazon RDS 数据库的性能并排查其故障。

借助数据库洞察，您可以直观呈现数据库实例集中的数据库负载，并按等待状态、SQL 语句、主机或用户来筛选负载。

默认情况下，RDS 为您的 Amazon RDS 数据库启用数据库洞察的标准模式。

有关在 Amazon CloudWatch 控制台中使用数据库洞察的信息，请参阅《Amazon CloudWatch 用户指南》中的 CloudWatch 数据库洞察。

定价

有关定价的信息，请参阅 Amazon CloudWatch 定价。

主题

支持数据库洞察的 Amazon RDS 数据库引擎、区域和实例类

下表提供支持数据库洞察的 Amazon RDS 数据库引擎。

Amazon RDS 数据库引擎	支持的引擎版本和区域	实例类限制
Amazon RDS for MariaDB	有关适用于 RDS for MariaDB 的数据库洞察的版本和区域可用性的更多信息，请参阅支持 Amazon RDS 中 Performance Insights 的区域和数据库引擎。	以下实例类不支持数据库洞察： • db.t2.micro • db.t2.small

Amazon RDS 数据库引擎	支持的引擎版本和区域	实例类限制
		- db.t3.micro - db.t3.small - db.t4g.micro - db.t4g.small
RDS for MySQL	有关适用于 RDS for MySQL 的数据库洞察的版本和区域可用性的更多信息，请参阅支持 Amazon RDS 中 Performance Insights 的区域和数据库引擎。	以下实例类不支持数据库洞察： - db.t2.micro - db.t2.small - db.t3.micro - db.t3.small - db.t4g.micro - db.t4g.small
Amazon RDS for Microsoft SQL Server	有关适用于 RDS for SQL Server 的数据库洞察的版本和区域可用性的更多信息，请参阅支持 Amazon RDS 中 Performance Insights 的区域和数据库引擎。	不适用
Amazon RDS for PostgreSQL	有关适用于 RDS for PostgreSQL 的数据库洞察的版本和区域可用性的更多信息，请参阅支持 Amazon RDS 中 Performance Insights 的区域和数据库引擎。	不适用

Amazon RDS 数据库引擎	支持的引擎版本和区域	实例类限制
Amazon RDS for Oracle	有关适用于 RDS for Oracle 的数据库洞察的版本和区域可用性的更多信息，请参阅支持 Amazon RDS 中 Performance Insights 的区域和数据库引擎。	不适用

Amazon RDS 在以下 Amazon Web Services 区域中支持数据库洞察。

- 美国东部（弗吉尼亚州北部）
- 美国东部（俄亥俄州）
- 美国西部（加利福尼亚北部）
- 美国西部（俄勒冈州）
- 非洲（开普敦）
- 亚太地区（香港）
- 亚太地区（海得拉巴）
- 亚太地区（雅加达）
- 亚太地区（马来西亚）
- 亚太地区（墨尔本）
- 亚太地区（孟买）
- 亚太地区（大阪）
- 亚太地区（首尔）
- 亚太地区（新加坡）
- 亚太地区（悉尼）
- 亚太地区（东京）
- 加拿大（中部）
- 加拿大西部（卡尔加里）
- 欧洲地区（法兰克福）
- 欧洲地区（爱尔兰）

- 欧洲地区（伦敦）
- 欧洲地区（米兰）
- 欧洲地区（巴黎）
- 欧洲地区（西班牙）
- 欧洲地区（斯德哥尔摩）
- 欧洲（苏黎世）
- 以色列（特拉维夫）
- 中东（巴林）
- 中东（阿联酋）
- 南美洲（圣保罗）
- Amazon GovCloud（美国东部）
- Amazon GovCloud（美国西部）

开启适用于 Amazon RDS 的数据库洞察的高级模式

要开启适用于 Amazon RDS 的数据库洞察的高级模式，请使用以下过程。

在创建数据库实例或多可用区数据库集群时开启数据库洞察的高级模式

在为 Amazon RDS 创建数据库时，开启数据库洞察的高级模式。

Console

在控制台中，您可以在创建数据库实例或多可用区数据库集群时开启数据库洞察的高级模式。

在使用控制台创建数据库实例或多可用区数据库集群时，开启数据库洞察的高级模式

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 选择数据库。
3. 选择创建数据库。
4. 在数据库洞察部分，选择高级模式。然后，选择以下选项：
    - 保留 – 保留性能详情数据的时间。对于数据库洞察的高级模式，保留期必须为 15-24 个月。

  - Amazon KMS key - 指定您的 KMS 密钥。性能详情使用您的 KMS 密钥来加密所有潜在的敏感数据。正在传输的数据和静态数据都会被加密。有关更多信息，请参阅 加密 Amazon RDS 资源。

5. 选择创建数据库。

Amazon CLI

要在创建数据库实例或多可用区数据库集群时开启数据库洞察的高级模式，请调用 create-db-instance 或 create-db-cluster Amazon CLI 命令并提供以下值：

- `--database-insights-mode advanced`，开启数据库洞察的高级模式。
- `--engine`：数据库实例的数据库引擎。
- `--db-instance-identifier`：数据库实例的标识符或 `--db-cluster-identifier`：多可用区数据库集群的标识符。
- `--enable-performance-insights`，开启数据库洞察的性能详情。
- `--performance-insights-retention-period`：数据库实例或多可用区数据库集群的数据的保留期。要开启数据库洞察，保留期必须至少为 465 天。

以下示例在创建数据库实例时启用数据库洞察的高级模式。

对于 Linux、macOS 或 Unix：

```
aws rds create-db-instance \
    --database-insights-mode advanced \
    --engine postgresql \
    --db-instance-identifier sample-db-identifier \
    --enable-performance-insights \
    --performance-insights-retention-period 465
```

对于 Windows：

```
aws rds create-db-instance ^
    --database-insights-mode advanced ^
    --engine postgresql ^
    --db-instance-identifier sample-db-identifier ^
    --enable-performance-insights ^
    --performance-insights-retention-period 465
```

RDS API

要在创建数据库实例或多可用区数据库集群时开启数据库洞察的高级模式，请为 CreateDBInstance 或 CreateDBCluster Amazon RDS API 操作指定以下参数。

- `DatabaseInsightsMode` 到 `advanced`
- `EnablePerformanceInsights` 到 `True`
- `PerformanceInsightsRetentionPeriod` 至少为 465 天

在修改数据库实例或多可用区数据库集群时开启数据库洞察的高级模式

在为 Amazon RDS 修改数据库时，开启数据库洞察。

> **Note**
>
> 要启用数据库洞察，多可用区数据库集群中的每个数据库实例都必须具有相同的性能详情和增强型监控设置。

Console

在控制台中，您可以在修改数据库实例或多可用区数据库集群时开启数据库洞察的高级模式。

在使用控制台修改数据库实例或多可用区数据库集群时，开启数据库洞察的高级模式

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 选择数据库。
3. 选择数据库实例或多可用区数据库集群，然后选择修改。
4. 在数据库洞察部分，选择高级模式。然后，选择以下选项：
   - 保留 – 保留性能详情数据的时间。对于数据库洞察的高级模式，保留期必须为 15-24 个月。
   - Amazon KMS key - 指定您的 KMS 密钥。性能详情使用您的 KMS 密钥来加密所有潜在的敏感数据。正在传输的数据和静态数据都会被加密。有关更多信息，请参阅“加密 Amazon RDS 资源”。
5. 选择继续。
6. 对于修改计划，选择立即应用。如果您选择在下一个计划维护时段内应用，则您的数据库将忽略此设置，并立即开启数据库洞察的高级模式。

7. 选择修改实例。

Amazon CLI

要在修改数据库实例或多可用区数据库集群时开启数据库洞察的高级模式，请调用 modify-db-instance 或 modify-db-cluster Amazon CLI 命令并提供以下值：

- `--database-insights-mode advanced`，开启数据库洞察的高级模式。
- `--db-instance-identifier`：数据库实例的标识符或 `--db-cluster-identifier`：多可用区数据库集群的标识符。
- `--enable-performance-insights`，开启数据库洞察的性能详情。
- `--performance-insights-retention-period`：数据库实例的数据的保留期。要开启数据库洞察的高级模式，保留期必须至少为 465 天。

以下示例在修改数据库实例时启用数据库洞察的高级模式。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --database-insights-mode advanced \
    --db-instance-identifier sample-db-identifier \
    --enable-performance-insights \
    --performance-insights-retention-period 465
```

对于 Windows：

```
aws rds modify-db-instance ^
    --database-insights-mode advanced ^
    --db-instance-identifier sample-db-identifier ^
    --enable-performance-insights ^
    --performance-insights-retention-period 465
```

RDS API

要在修改数据库实例或多可用区数据库集群时开启数据库洞察的高级模式，请为您的 ModifyDBInstance 或 ModifyDBCluster Amazon RDS API 操作指定以下参数。

- `DatabaseInsightsMode` 到 `advanced`

- `EnablePerformanceInsights` 到 `True`
- `PerformanceInsightsRetentionPeriod` 至少为 465 天

开启适用于 Amazon RDS 的数据库洞察的标准模式

要开启适用于 Amazon RDS 的数据库洞察的标准模式，请使用以下过程。

在创建数据库实例或多可用区数据库集群时开启数据库洞察的标准模式

在为 Amazon RDS 创建数据库时，开启数据库洞察的标准模式。

Console

在控制台中，您可以在创建数据库实例或多可用区数据库集群时开启数据库洞察的标准模式。

在使用控制台创建数据库实例或多可用区数据库集群时，开启数据库洞察的标准模式

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 选择数据库。
3. 选择创建数据库。
4. 在数据库洞察部分，选择标准模式。然后，从以下选项中进行选择，打开或关闭性能详情：
   - 要关闭性能详情，请取消选择启用性能详情。
   - 要开启性能详情，请选择启用性能详情。要配置性能详情，请指定以下选项：
     - 保留 – 保留性能详情数据的时间。保留期必须至少为 7 天。
     - Amazon KMS key - 指定您的 KMS 密钥。性能详情使用您的 KMS 密钥来加密所有潜在的敏感数据。正在传输的数据和静态数据都会被加密。有关更多信息，请参阅 加密 Amazon RDS 资源。
5. 选择创建数据库。

Amazon CLI

要在创建数据库实例或多可用区数据库集群时开启数据库洞察的标准模式，请调用 create-db-instance 或 create-db-cluster Amazon CLI 命令并提供以下值：

- `--database-insights-mode standard`，开启数据库洞察的标准模式。

- --engine：数据库实例的数据库引擎。
- --db-instance-identifier：数据库实例的标识符或 --db-cluster-identifier：多可用区数据库集群的标识符。
- --enable-performance-insights 或 --no-enable-performance-insights，打开和关闭性能详情。如果指定 --enable-performance-insights，则还必须指定 --performance-insights-retention-period，即数据库实例的数据的保留期。保留期必须至少为 7 天。

以下示例在创建数据库实例时启用数据库洞察的标准模式和性能详情。

对于 Linux、macOS 或 Unix：

```
aws rds create-db-instance \
    --database-insights-mode standard \
    --engine postgresql \
    --db-instance-identifier sample-db-identifier \
    --enable-performance-insights \
    --performance-insights-retention-period 7
```

对于 Windows：

```
aws rds create-db-instance ^
    --database-insights-mode standard ^
    --engine postgresql ^
    --db-instance-identifier sample-db-identifier ^
    --enable-performance-insights ^
    --performance-insights-retention-period 7
```

以下示例在创建数据库实例时启用数据库洞察的标准模式，但禁用性能详情。

对于 Linux、macOS 或 Unix：

```
aws rds create-db-instance \
    --database-insights-mode standard \
    --engine postgresql \
    --db-instance-identifier sample-db-identifier \
    --no-enable-performance-insights
```

对于 Windows：

```
aws rds create-db-instance ^
    --database-insights-mode standard ^
    --engine postgresql ^
    --db-instance-identifier sample-db-identifier ^
    --no-enable-performance-insights
```

RDS API

要在创建数据库实例或多可用区数据库集群时开启数据库洞察的标准模式，请为您的 CreateDBInstance 或 CreateDBCluster Amazon RDS API 操作指定以下参数。

- `DatabaseInsightsMode` 到 `standard`
- 将 `EnablePerformanceInsights` 设置为 `True` 或 `False`。如果将 `EnablePerformanceInsights` 设置为 `True`，则必须将 `PerformanceInsightsRetentionPeriod` 设置为至少 7 天。

在修改数据库实例或多可用区数据库集群时，开启数据库洞察的标准模式

在为 Amazon RDS 修改数据库时，开启数据库洞察的标准模式。

> **Note**
>
> 要启用数据库洞察，多可用区数据库集群中的每个数据库实例都必须具有相同的性能详情和增强型监控设置。

Console

在控制台中，您可以在修改数据库实例或多可用区数据库集群时开启数据库洞察的标准模式。

在使用控制台修改数据库实例或多可用区数据库集群时，开启数据库洞察的标准模式

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 选择数据库。
3. 选择数据库实例或多可用区数据库集群，然后选择修改。
4. 在数据库洞察部分，选择标准模式。然后从以下选项中进行选择：

   - 要关闭性能详情，请取消选择启用性能详情。

- 要开启性能详情，请选择启用性能详情。要配置性能详情，请指定以下选项：
  - 保留 – 保留性能详情数据的时间。保留期必须至少为 7 天。
  - Amazon KMS key - 指定您的 KMS 密钥。性能详情使用您的 KMS 密钥来加密所有潜在的敏感数据。正在传输的数据和静态数据都会被加密。有关更多信息，请参阅“加密 Amazon RDS 资源”。

5. 选择继续。
6. 对于修改计划，选择立即应用。如果您选择在下一个计划维护时段内应用，则您的数据库将忽略此设置，并立即开启数据库洞察的标准模式。
7. 选择修改实例。

Amazon CLI

要在修改数据库实例或多可用区数据库集群时开启数据库洞察的标准模式，请调用 modify-db-instance 或 modify-db-cluster Amazon CLI 命令并提供以下值：

- `--database-insights-mode standard`，开启数据库洞察的标准模式。
- `--db-instance-identifier`：数据库实例的标识符或 `--db-cluster-identifier`：多可用区数据库集群的标识符。
- `--enable-performance-insights` 或 `--no-enable-performance-insights`，打开和关闭性能详情。如果指定 `--enable-performance-insights`，则还必须指定 `--performance-insights-retention-period`，即数据库实例或多可用区数据库集群的数据的保留期。保留期必须至少为 7 天。

以下示例在修改数据库实例时启用数据库洞察的标准模式并启用性能详情。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --database-insights-mode standard \
    --db-instance-identifier sample-db-identifier \
    --enable-performance-insights \
    --performance-insights-retention-period 7
```

对于 Windows：

```
aws rds modify-db-instance ^
```

```
    --database-insights-mode standard ^
    --db-instance-identifier sample-db-identifier ^
    --enable-performance-insights ^
    --performance-insights-retention-period 7
```

以下示例在修改数据库实例时启用数据库洞察的标准模式并禁用性能详情。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --database-insights-mode standard \
    --db-instance-identifier sample-db-identifier \
    --no-enable-performance-insights
```

对于 Windows：

```
aws rds modify-db-instance ^
    --database-insights-mode standard ^
    --db-instance-identifier sample-db-identifier ^
    --no-enable-performance-insights
```

RDS API

要在修改数据库实例或多可用区数据库集群时开启数据库洞察的标准模式，请为您的 ModifyDBInstance 或 ModifyDBCluster Amazon RDS API 操作指定以下参数。

- `DatabaseInsightsMode` 到 `standard`
- 将 `EnablePerformanceInsights` 设置为 `True` 或 `False`。如果将 `EnablePerformanceInsights` 设置为 `True`，则必须将 `PerformanceInsightsRetentionPeriod` 设置为至少 7 天。

使用适用于 Amazon RDS 的数据库洞察配置您的数据库以监控慢速 SQL 查询

要监控数据库的慢速 SQL 查询，可以使用“数据库洞察”控制面板中的慢速 SQL 查询部分。在配置数据库以监控慢速 SQL 查询之前，慢速 SQL 查询部分为空。

有关在“数据库洞察”控制面板中监控慢速 SQL 查询的更多信息，请参阅《Amazon CloudWatch 用户指南》中的查看数据库实例控制面板以获取 CloudWatch 数据库洞察。

要使用数据库洞察配置您的数据库以监控慢速 SQL 查询，请完成以下步骤：

1. 启用到 CloudWatch Logs 的日志导出。
2. 创建或修改数据库实例的数据库参数组。

有关配置日志导出的信息，请参阅《Amazon RDS 用户指南》中的将数据库日志发布到 Amazon CloudWatch Logs。

要创建或修改数据库参数组，请参阅以下主题。

- 在 Amazon RDS 中创建数据库参数组
- 在 Amazon RDS 中修改数据库参数组中的参数

RDS for MariaDB

要将 RDS for MariaDB 数据库实例配置为监控慢速 SQL 查询，请设置以下参数。

- `log_slow_query` – 设置为 `1`
- `log_slow_query_time` – 设置为 `1.0`
- `log_output` – 设置为 `FILE`

RDS for MySQL

要将 RDS for MySQL 数据库实例配置为监控慢速 SQL 查询，请设置以下参数。

- `slow_query_log` – 设置为 `1`
- `long_query_time` – 设置为 `1.0`
- `log_output` – 设置为 `FILE`

RDS for PostgreSQL

要将 RDS for PostgreSQL 数据库实例配置为监控慢速 SQL 查询，请设置以下参数。请注意，设置这些参数可能会降低数据库实例的性能。

- `log_min_duration_statement` – 设置为 `1000`
- `log_statement` – 设置为 `none`
- `log_destination` – 设置为 `stderr`

> **Note**
>
> 对于 RDS for MySQL，可以将参数 `long_query_time` 配置为 1 微秒的粒度。例如，您可以将此参数设置为 `0.000001`。根据数据库实例上的查询量，参数 `long_query_time` 的值可能会降低性能。从值 `1.0` 开始，然后根据工作负载进行调整。当您将此参数设置为 `0` 时，数据库洞察会记录所有查询。

有关 RDS for MariaDB、RDS for MySQL 和 RDS for PostgreSQL 日志的信息，请参阅以下内容。

- MariaDB 数据库日志文件
- MySQL 数据库日志文件
- RDS for PostgreSQL 数据库日志文件

适用于 Amazon RDS 的数据库洞察的注意事项

以下是适用于 Amazon RDS 的数据库洞察的注意事项。

- 您无法管理多可用区数据库集群中数据库实例的数据库洞察。
- 要启用数据库洞察的高级模式，您必须启用性能详情并将性能详情的保留期设置为至少 465 天（15 个月）。除了数据库洞察的成本外，将性能详情的保留期设置为 15 个月没有额外成本。有关数据库洞察的定价信息，请参阅 Amazon CloudWatch 定价。
- 要启用数据库洞察，多可用区数据库集群中的每个数据库实例都必须具有相同的性能详情和增强型监控设置。

在 Amazon RDS 上使用性能详情监控数据库负载

> **Important**
>
> Amazon 已宣布性能详情的生命周期终止日期为：2025 年 11 月 30 日。在此日期之后，Amazon RDS 将不再支持性能详情控制台体验、灵活的保留期（1-24 个月）及其关联定价。性能详情 API 将继续存在，定价不变。性能详情 API 的费用将与 CloudWatch 数据库洞察的费用一起显示在您的 Amazon 账单中。
>
> 我们建议您在 2025 年 11 月 30 日之前，将使用性能详情付费套餐的所有数据库实例升级到数据库洞察的高级模式。有关升级到数据库洞察的高级模式的信息，请参阅开启适用于 Amazon RDS 的数据库洞察的高级模式。
>
> 如果您不采取任何行动，则使用性能详情的数据库实例将默认为使用数据库洞察的标准模式。在数据库洞察的标准模式下，您可能会在 7 天后无法访问性能数据历史记录，并且可能无法在 Amazon RDS 控制台中使用执行计划和按需分析功能。在 2025 年 11 月 30 日之后，只有数据库洞察的高级模式将支持执行计划和按需分析。
>
> 借助 CloudWatch 数据库洞察，您可以监控数据库实例集的数据库负载，并对性能进行大规模分析和故障排除。有关数据库洞察的更多信息，请参阅使用 CloudWatch 数据库洞察监控 Amazon RDS 数据库。有关定价信息，请参阅 Amazon CloudWatch 定价。

Performance Insights 在现有 Amazon RDS 监控功能的基础上进行了扩展，以便展示并帮助您分析数据库性能。利用 Performance Insights 控制面板，您可以可视化 Amazon RDS 数据库实例负载上的数据库负载，并按等待状态、SQL 语句、主机或用户来筛选负载。有关将 Performance Insights 与 Amazon DocumentDB 结合使用的信息，请参阅 Amazon DocumentDB 开发人员指南。

主题

Amazon RDS 上的 Performance Insights 概述

> **Important**
>
> Amazon 已宣布性能详情的生命周期终止日期为：2025 年 11 月 30 日。在此日期之后，Amazon RDS 将不再支持性能详情控制台体验、灵活的保留期（1-24 个月）及其关联定价。性能详情 API 将继续存在，定价不变。性能详情 API 的费用将与 CloudWatch 数据库洞察的费用一起显示在您的 Amazon 账单中。
>
> 我们建议您在 2025 年 11 月 30 日之前，将使用性能详情付费套餐的所有数据库实例升级到数据库洞察的高级模式。有关升级到数据库洞察的高级模式的信息，请参阅开启适用于 Amazon RDS 的数据库洞察的高级模式。
>
> 如果您不采取任何行动，则使用性能详情的数据库实例将默认为使用数据库洞察的标准模式。在数据库洞察的标准模式下，您可能会在 7 天后无法访问性能数据历史记录，并且可能无法在 Amazon RDS 控制台中使用执行计划和按需分析功能。在 2025 年 11 月 30 日之后，只有数据库洞察的高级模式将支持执行计划和按需分析。
>
> 借助 CloudWatch 数据库洞察，您可以监控数据库实例集的数据库负载，并对性能进行大规模分析和故障排除。有关数据库洞察的更多信息，请参阅使用 CloudWatch 数据库洞察监控 Amazon RDS 数据库。有关定价信息，请参阅 Amazon CloudWatch 定价。

默认情况下，RDS 在控制台创建向导中为所有 Amazon RDS 引擎启用 Performance Insights。如果一个数据库实例中有多个数据库，则 Performance Insights 会聚合性能数据。

可在以下视频中大致了解 Amazon RDS 的 Performance Insights。

使用 Performance Insights 分析 Amazon Aurora PostgreSQL 的性能

> **Important**
>
> 以下主题说明如何将 Amazon RDS Performance Insights 与非 Aurora 数据库引擎结合使用。有关将 Amazon RDS Performance Insights 与 Amazon Aurora 结合使用的信息，请参阅《Amazon Aurora 用户指南》中的使用 Amazon RDS Performance Insights。

主题

- 数据库负载
- 最大 CPU

- 支持性能详情的 Amazon RDS 数据库引擎、区域和实例类
- 性能详情的定价和数据留存

数据库负载

数据库负载（DB 负载）衡量数据库中的会话活动级别。`DBLoad` 是 Performance Insights 中的关键指标，Performance Insights 每秒都会收集数据库负载。

主题

- 活动的会话
- 平均活动会话数
- 平均活动执行数
- 尺寸

活动的会话

数据库会话表示的是应用程序与关系数据库的对话。活动的会话 是已将作业提交到数据库引擎并且正在等待响应的连接。

当会话在 CPU 上运行或等待资源变为可用以便继续执行时，该会话即处于活动状态。例如，活动的会话可能会等待页面（或块）读入内存中，然后占用 CPU 以从页面读取数据。

平均活动会话数

平均活动会话数（AAS）是 Performance Insights 中 `DBLoad` 指标的单位。它衡量数据库上有多少个会话同时处于活动状态。

Performance Insights 每秒都会对并行运行查询的会话数进行采样。对于每个活动的会话，Performance Insights 都会收集以下数据：

- SQL 语句
- 会话状态（正在 CPU 上运行或正在等待）
- Host
- 运行 SQL 的用户

Performance Insights 通过以下方式计算 AAS：对于特定时间段内，将会话总数除以样本数。例如，下表显示了正在运行的查询的 5 个连续样本，其间隔为 1 秒。

示例	运行查询的会话数	AAS	计算
1	2	2	总计 2 个会话/1 个样本
2	0	1	总计 2 个会话/2 个样本
3	4	2	总计 6 个会话/3 个样本
4	0	1.5	总计 6 个会话/4 个样本
5	4	2	总计 10 个会话/5 个样本

在前面的示例中，该时间间隔的数据库负载为 2 AAS。这一测量结果表明，在采集 5 个样本的时间间隔内，在任何给定时间，平均有 2 个会话处于活动状态。

平均活动执行数

每秒的平均活动执行数 (AAE) 与平均活动会话数相关。为了计算平均活动执行数，Performance Insights 使用查询的总执行时间除以时间间隔。下表显示了上表中同一查询的平均活动执行数计算。

运行时间（秒）	总执行时间（秒）	AAE	计算
60	120	2	120 秒执行时间/60 秒运行时间
120	120	1	120 秒执行时间/120 秒运行时间
180	380	2.1.1	380 秒执行时间/180 秒运行时间
240	380	1.58	380 秒执行时间/240 秒运行时间
300	600	2	600 秒执行时间/300 秒运行时间

在大多数情况下，查询的平均活动会话数和平均活动执行数大致相同。但是，由于计算所用的数据是不同的数据源，因此计算通常略有不同。

尺寸

`db.load` 指标不同于其他时间序列指标，因为您可以将它分为称为维度的子组件。您可以将维度视为 `DBLoad` 指标的不同特征的“切片依据”类别。

诊断性能问题时，以下维度通常最有用：

主题

- 等待事件
- 主要 SQL
- 计划

有关 Amazon RDS 引擎维度的完整列表，请参阅 按维度切片的数据库负载。

等待事件

等待事件 会导致 SQL 语句等待特定事件发生，然后才能继续运行。等待事件是数据库负载的一个重要维度（或称类别），因为它们指示了工作受阻的位置。

每个活动的会话要么会在 CPU 上运行，要么仍在等待。例如，在搜索内存寻找缓冲区、执行计算或运行过程代码时，会话都会占用 CPU。当会话不占用 CPU 时，它们可能正在等待内存缓冲区变为空闲、等待要读取的数据文件或等待将要写入的日志。会话等待资源的时间越长，它在 CPU 上运行的时间就越少。

当您优化数据库时，经常尝试找出会话正在等待的资源。例如，两三个等待事件可能会占据数据库负载的 90%。这一测量结果表明，平均而言，活动会话花费了大部分时间用于等待少量资源。如果您能找出导致这些等待的原因，就可以尝试提出解决方案了。

等待事件因数据库引擎而异：

- 有关所有 MariaDB 和 MySQL 等待事件的信息，请参阅 MySQL 文档中的等待事件摘要表。
- 有关所有 PostgreSQL 等待事件的信息，请参阅 PostgreSQL 文档中的统计数据收集器 > 等待事件表。
- 有关所有 Oracle 等待事件的信息，请参阅 Oracle 文档中的等待事件说明。
- 有关所有 SQL Server 等待事件的信息，请参阅 SQL Server 文档中的等待类型。

> **Note**
>
> 对于 Oracle，后台进程有时在没有关联的 SQL 语句的情况下工作。在这些情况下，Performance Insights 报告以冒号连接的后台进程类型以及与该后台进程关联的等待类。后台进程类型包括 `LGWR`、`ARC0`、`PMON` 等。
>
> 例如，在存档程序执行 I/O 时，它的 Performance Insights 报告类似于 `ARC1:System I/O`。有时，还会缺少后台进程类型，而 Performance Insights 仅报告等待类，例如 `:System I/O`。

主要 SQL

等待事件显示运行中的瓶颈，而主要 SQL 则显示哪些查询对数据库负载的影响最大。例如，当前可能正在数据库上运行着许多查询，但某一个查询可能会占用 99% 的数据库负载。在这种情况下，高负载可能表示查询存在问题。

默认情况下，Performance Insights 控制台会显示导致数据库负载的主要 SQL 查询。控制台还显示每个语句的相关统计信息。要诊断特定语句的性能问题，可以检查其执行计划。

计划

执行计划，也简称为计划，是访问数据的一系列步骤。例如，连接表 `t1` 和 `t2` 的计划可能会循环访问 `t1` 中的所有行，并将每一行与 `t2` 中的行进行比较。在关系数据库中，优化程序是确定 SQL 查询最有效计划的内置代码。

对于数据库实例，Performance Insights 会自动收集执行计划。要诊断 SQL 性能问题，请检查为高资源 SQL 查询捕获的计划。这些计划会显示数据库如何解析和运行查询。

要了解如何使用计划分析数据库负载，请参阅：

- Oracle：使用 Amazon RDS 的性能详情控制面板分析 Oracle 执行计划
- SQL Server：使用 Amazon RDS 的性能详情控制面板分析 SQL Server 执行计划

计划捕获

Performance Insights 每五分钟就会识别资源密集度最高的查询并捕获其计划。因此，无需手动收集和管理大量计划。相反，您可以使用 Top SQL（主要 SQL）选项卡重点关注问题最严重的查询计划。

> **Note**
>
> Performance Insights 不会捕获文本超过最大可收集查询文本限制的查询计划。有关更多信息，请参阅 在 Performance Insights 控制面板中访问更多 SQL 文本。

执行计划的保留期与 Performance Insights 数据的保留期相同。免费套餐中的保留设置为 Default (7 days) [原定设置（7 天）]。要将性能数据保留更长时间，请指定 1–24 个月。有关保留期的更多信息，请参阅性能详情的定价和数据留存。

摘要查询

Top SQL（主要 SQL）选项卡默认显示摘要查询。摘要查询本身没有计划，但所有使用文本值的查询都有计划。例如，摘要查询可能包括文本 `WHERE `email`=?`。摘要可能包含两个查询，一个带有文本 `WHERE email=user1@example.com`，另一个带有 `WHERE email=user2@example.com`。每个文本查询都可能包括多个计划。

如果选择摘要查询，控制台将显示所选摘要的子语句的所有计划。因此，无需浏览所有子语句即可查看计划。计划可能不会显示在前 10 个子语句的列表中。控制台会显示已收集计划的所有子查询计划，无论查询是否在前 10 列表中。

最大 CPU

在控制面板中，数据库负载图表会收集、聚合和显示会话信息。要查看活动会话是否超过最大 CPU，请查看它们与最大 vCPU 线的关系。Performance Insights 根据数据库实例的 vCPU（虚拟 CPU）内核数，来决定最大 vCPU 值。

一次只能在 vCPU 上运行一个进程。如果进程数超过 vCPU 的数量，则进程开始排队。当排队增加时，数据库性能就会降低。如果数据库负载经常高于最大 vCPU 线并且主要等待状态为 CPU，则表示 CPU 过载。在这种情况下，您可能需要限制与实例的连接数，优化具有高 CPU 负载的任何 SQL 查询，或考虑使用更大的实例类。如果始终有大量实例处于任何等待状态，则表示可能存在要解决的瓶颈或资源争用问题。即使数据库负载未越过最大 vCPU 线，也可能会出现此问题。

支持性能详情的 Amazon RDS 数据库引擎、区域和实例类

> **Important**
>
> Amazon 已宣布性能详情的生命周期终止日期为：2025 年 11 月 30 日。在此日期之后，Amazon RDS 将不再支持性能详情控制台体验、灵活的保留期（1-24 个月）及其关联定

价。性能详情 API 将继续存在，定价不变。性能详情 API 的费用将与 CloudWatch 数据库洞察的费用一起显示在您的 Amazon 账单中。

我们建议您在 2025 年 11 月 30 日之前，将使用性能详情付费套餐的所有数据库实例升级到数据库洞察的高级模式。有关升级到数据库洞察的高级模式的信息，请参阅开启适用于 Amazon RDS 的数据库洞察的高级模式。

如果您不采取任何行动，则使用性能详情的数据库实例将默认为使用数据库洞察的标准模式。在数据库洞察的标准模式下，您可能会在 7 天后无法访问性能数据历史记录，并且可能无法在 Amazon RDS 控制台中使用执行计划和按需分析功能。在 2025 年 11 月 30 日之后，只有数据库洞察的高级模式将支持执行计划和按需分析。

借助 CloudWatch 数据库洞察，您可以监控数据库实例集的数据库负载，并对性能进行大规模分析和故障排除。有关数据库洞察的更多信息，请参阅使用 CloudWatch 数据库洞察监控 Amazon RDS 数据库。有关定价信息，请参阅 Amazon CloudWatch 定价。

下表提供支持性能详情的 Amazon RDS 数据库引擎。

> **Note**
>
> 有关 Amazon Aurora，请参阅《Amazon Aurora 用户指南》中的支持性能详情的 Amazon Aurora 数据库引擎

Amazon RDS 数据库引擎	支持的引擎版本和区域	实例类限制
Amazon RDS for MariaDB	有关适用于 RDS for MariaDB 的性能详情的版本和区域可用性的更多信息，请参阅支持 Amazon RDS 中 Performance Insights 的区域和数据库引擎。	以下实例类别不支持性能详情： • db.t2.micro • db.t2.small • db.t3.micro • db.t3.small • db.t4g.micro •

Amazon RDS 数据库引擎	支持的引擎版本和区域	实例类限制
		db.t4g.small
RDS for MySQL	有关适用于 RDS for MySQL 的性能详情的版本和区域可用性的更多信息，请参阅支持 Amazon RDS 中 Performance Insights 的区域和数据库引擎。	以下实例类别不支持性能详情： • db.t2.micro • db.t2.small • db.t3.micro • db.t3.small • db.t4g.micro • db.t4g.small
Amazon RDS for Microsoft SQL Server	有关适用于 RDS for SQL Server 的性能详情的版本和区域可用性的更多信息，请参阅支持 Amazon RDS 中 Performance Insights 的区域和数据库引擎。	不适用
Amazon RDS for PostgreSQL	有关适用于 RDS for PostgreSQL 的性能详情的版本和区域可用性的更多信息，请参阅支持 Amazon RDS 中 Performance Insights 的区域和数据库引擎。	不适用
Amazon RDS for Oracle	有关适用于 RDS for Oracle 的性能详情的版本和区域可用性的更多信息，请参阅支持 Amazon RDS 中 Performance Insights 的区域和数据库引擎。	不适用

支持性能详情功能的 Amazon RDS 数据库引擎、区域和实例类

下表提供支持性能详情功能的 Amazon RDS 数据库引擎。

功能	定价套餐	支持的区域	支持的数据库引擎	支持的实例类
Performance Insights 的 SQL 统计数据	全部	全部	全部	全部
使用 Amazon RDS 的性能详情控制面板分析 Oracle 执行计划	全部	全部	RDS for Oracle	全部
分析一段时间内的数据库性能	仅限付费套餐	全部	• RDS for MariaDB • RDS for MySQL • RDS for PostgreSQL	全部
查看性能详情主动建议	仅限付费套餐	• 美国东部（俄亥俄州） • 美国东部（弗吉尼亚州北部） • 美国西部（加利福尼亚北部） • 美国西部（俄勒冈州） • 亚太地区（孟买） • 亚太地区（首尔）	全部	全部

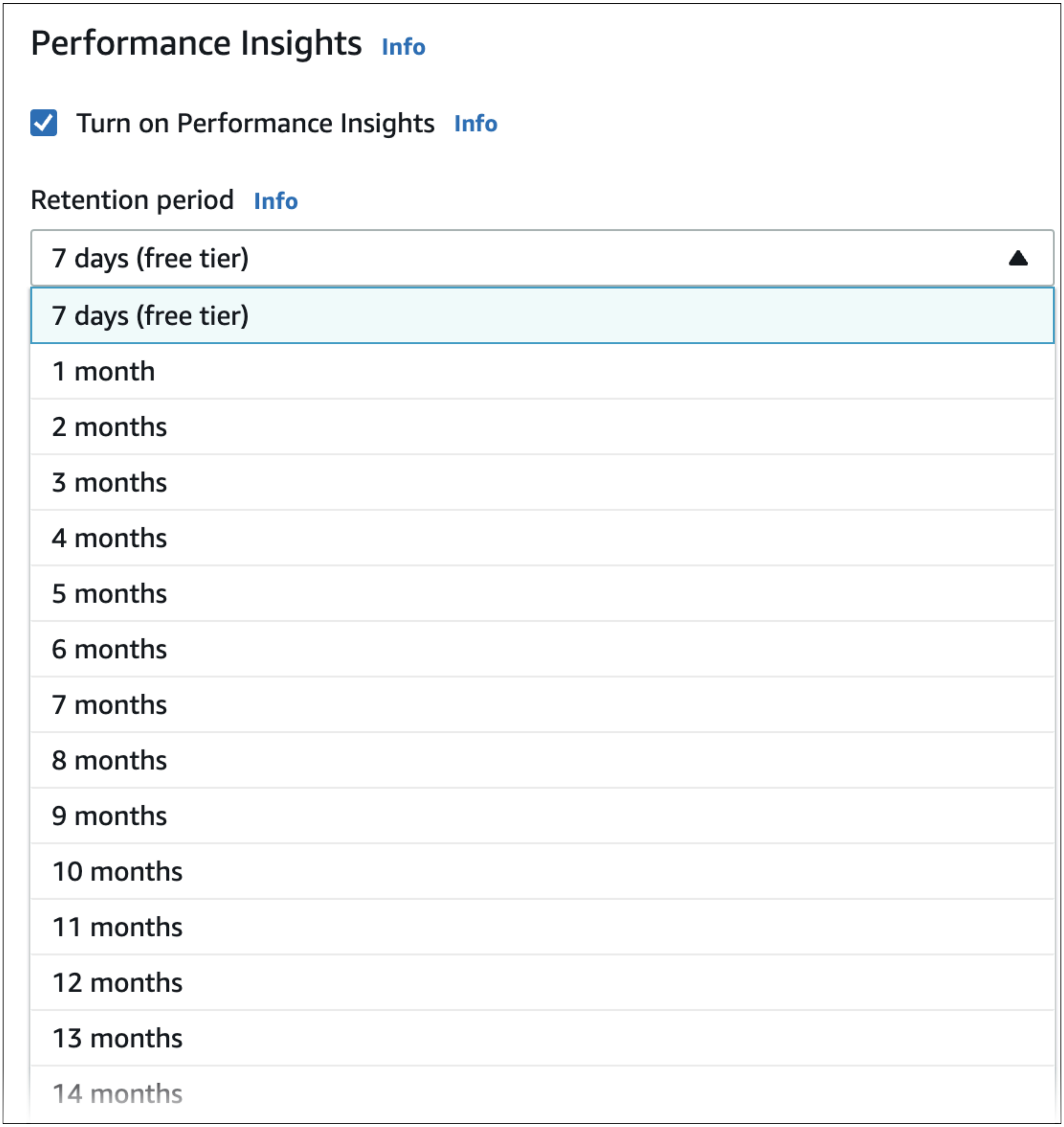
功能	定价套餐	支持的区域	支持的数据库引擎	支持的实例类
		• 亚太地区（新加坡） • 亚太地区（悉尼） • 亚太地区（东京） • 加拿大（中部） • 欧洲地区（法兰克福） • 欧洲地区（爱尔兰） • 欧洲地区（伦敦） • 欧洲地区（巴黎） • 欧洲地区（斯德哥尔摩） • 南美洲（圣保罗）		

性能详情的定价和数据留存

原定设置情况下，性能详情提供免费套餐，其中包括 7 天的性能数据历史记录和每月 100 万个 API 请求。您也可以购买更长的保留期。有关完整的定价信息，请参阅性能详情定价。

在 RDS 控制台中，您可以为性能详情数据选择以下任何保留期：

- 原定设置（7 天）
- ***n*** 个月，其中 ***n*** 是 1–24 之间的数字



要了解如何使用 Amazon CLI 设置保留期，请参阅[为 Amazon RDS 打开和关闭性能详情](#)。

> **Note**
>
> 在启用性能详情的情况下停止数据库实例或多可用区数据库集群不会影响数据留存。当数据库实例或多可用区数据库集群停止时，性能详情不会收集任何数据。

为 Amazon RDS 打开和关闭性能详情

> **Important**
>
> Amazon 已宣布性能详情的生命周期终止日期为：2025 年 11 月 30 日。在此日期之后，Amazon RDS 将不再支持性能详情控制台体验、灵活的保留期（1-24 个月）及其关联定价。性能详情 API 将继续存在，定价不变。性能详情 API 的费用将与 CloudWatch 数据库洞察的费用一起显示在您的 Amazon 账单中。
>
> 我们建议您在 2025 年 11 月 30 日之前，将使用性能详情付费套餐的所有数据库实例升级到数据库洞察的高级模式。有关升级到数据库洞察的高级模式的信息，请参阅开启适用于 Amazon RDS 的数据库洞察的高级模式。
>
> 如果您不采取任何行动，则使用性能详情的数据库实例将默认为使用数据库洞察的标准模式。在数据库洞察的标准模式下，您可能会在 7 天后无法访问性能数据历史记录，并且可能无法在 Amazon RDS 控制台中使用执行计划和按需分析功能。在 2025 年 11 月 30 日之后，只有数据库洞察的高级模式将支持执行计划和按需分析。
>
> 借助 CloudWatch 数据库洞察，您可以监控数据库实例集的数据库负载，并对性能进行大规模分析和故障排除。有关数据库洞察的更多信息，请参阅使用 CloudWatch 数据库洞察监控 Amazon RDS 数据库。有关定价信息，请参阅 Amazon CloudWatch 定价。

您可以在创建数据库实例或多可用区数据库集群时，为其打开性能详情。如果需要，可以稍后通过控制台修改数据库实例来关闭性能详情。打开和关闭性能详情不会导致停机、重启或失效转移。

> **Note**
>
> Performance Schema 是 Amazon RDS for MariaDB or MySQL 使用的一个可选性能工具。如果打开或关闭 Performance Schema，则需要重新启动。但是，如果您打开或关闭性能详情，则无需重新启动。有关更多信息，请参阅 Amazon RDS for MariaDB 或 Amazon RDS for MySQL 上性能详情的性能架构概述。

性能详情代理占用数据库主机上有限的 CPU 和内存。当数据库负载较高时，代理将通过降低收集数据的频率来限制性能影响。

Console

在控制台中，您可以在创建或修改数据库实例或多可用区数据库集群时打开或关闭性能详情。

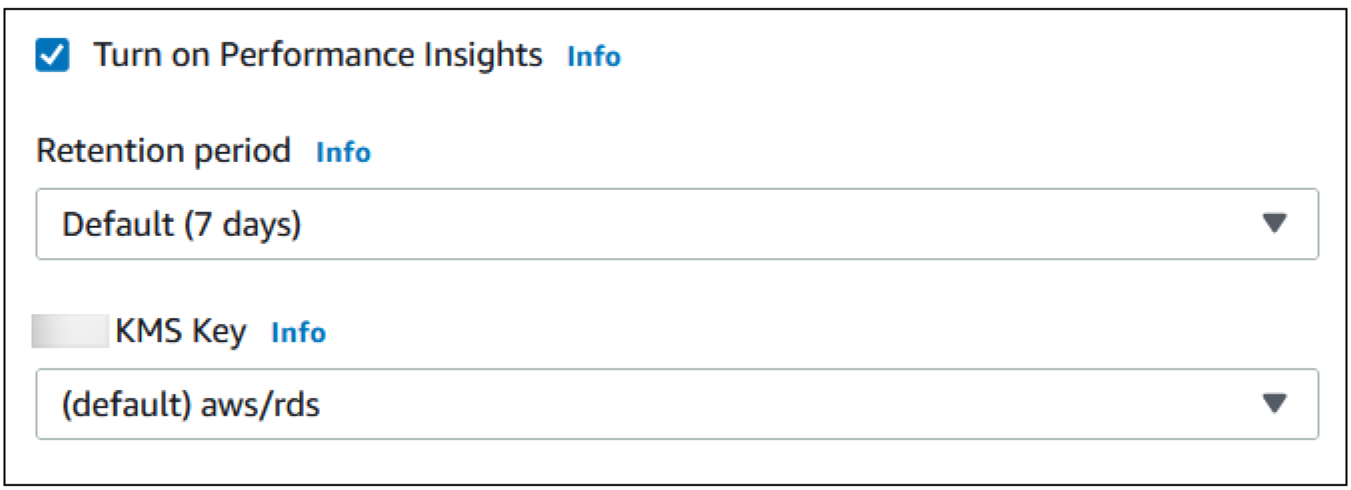
创建数据库实例或多可用区数据库集群时打开或关闭性能详情

创建新的数据库实例或多可用区数据库集群后，Amazon RDS 默认情况下会启用性能详情。要关闭性能详情，请选择数据库洞察 - 标准选项，然后取消选择启用性能详情选项。

有关更多信息，请参阅以下主题。

- 要创建数据库实例，请按照创建 Amazon RDS 数据库实例中用于您的数据库引擎的说明操作。
- 要创建多可用区数据库集群，请按 创建 Amazon RDS 的多可用区数据库集群 中数据库引擎的说明操作。

以下屏幕截图显示了性能详情部分。



如果您选择启用性能详情，您有以下选项：

- 保留期（仅适用于数据库洞察的标准模式）：保留性能详情数据的时间量。免费套餐中的保留设置为 Default (7 days) [默认（7 天）]。要将性能数据保留更长时间，请指定 1–24 个月。有关保留期的更多信息，请参阅性能详情的定价和数据留存。
- Amazon KMS key - 指定您的 Amazon KMS key。性能详情使用您的 KMS 密钥来加密所有潜在的敏感数据。正在传输的数据和静态数据都会被加密。有关更多信息，请参阅 更改性能详情的 Amazon KMS 策略。

在修改数据库实例或多可用区数据库集群时打开或关闭性能详情

在控制台中，您可以修改数据库实例或多可用区数据库集群来管理性能详情。

使用控制台为数据库实例或多可用区数据库集群管理性能详情

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 选择数据库。
3. 选择数据库实例或多可用区数据库集群，然后选择修改。
4. 要开启性能详情，请选择启用性能详情。要关闭性能详情，请选择数据库洞察 - 标准选项，然后取消选择启用性能详情选项。

   如果您选择启用性能详情，您有以下选项：

   - 保留期（仅适用于数据库洞察的标准模式）：保留性能详情数据的时间量。免费套餐中的保留设置为 Default (7 days) [默认（7 天）]。要将性能数据保留更长时间，请指定 1–24 个月。有关保留期的更多信息，请参阅性能详情的定价和数据留存。
   - Amazon KMS key - 指定您的 KMS 密钥。性能详情使用您的 KMS 密钥来加密所有潜在的敏感数据。正在传输的数据和静态数据都会被加密。有关更多信息，请参阅 加密 Amazon RDS 资源。
5. 选择继续。
6. 对于修改计划，请选择“立即应用”。如果您选择“Apply during the next scheduled maintenance window”（在下一个计划的维护时段内应用），则您的实例将忽略此设置并立即打开性能详情。
7. 选择修改实例。

Amazon CLI

使用 create-db-instance Amazon CLI 命令时，通过指定 `--enable-performance-insights` 可开启性能详情，并将 `--database-insights-mode` 设置为 `advanced` 或 `standard`。要关闭性能详情，请指定 `--no-enable-performance-insights` 并将 `database-insights-mode` 设置为 `standard`。

您还可以使用以下 Amazon CLI 命令指定这些值：

- create-db-cluster
- modify-db-cluster

- create-db-instance-read-replica
- modify-db-instance
- restore-db-instance-from-s3

在 CLI 中打开性能详情时，可以选择通过 `--performance-insights-retention-period` 选项指定保留性能详情数据的天数。您可以指定 7、*month* * 31（其中 *month* 为 1–23 之间的数字）或 731。例如，如果您想将性能数据保留 3 个月，请指定 93，也就是 3 * 31。默认值为 7 天。有关保留期的更多信息，请参阅性能详情的定价和数据留存。

以下示例为 `sample-db-cluster` 开启性能详情并指定将性能详情数据保留 93 天（3 个月）。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-cluster \
 --database-insights-mode standard \
    --db-cluster-identifier sample-db-instance \
    --enable-performance-insights \
    --performance-insights-retention-period 93
```

对于 Windows：

```
aws rds modify-db-cluster ^
 --database-insights-mode standard ^
    --db-cluster-identifier sample-db-instance ^
    --enable-performance-insights ^
    --performance-insights-retention-period 93
```

如果您指定保留期（如 94 天），这不是一个有效值，则 RDS 会发出错误。

```
An error occurred (InvalidParameterValue) when calling the CreateDBInstance
 operation:
Invalid Performance Insights retention period. Valid values are: [7, 31, 62, 93,
 124, 155, 186, 217,
248, 279, 310, 341, 372, 403, 434, 465, 496, 527, 558, 589, 620, 651, 682, 713, 731]
```

> **Note**
>
> 对于未在集群级别管理性能详情的数据库集群中的实例，只能切换性能详情。

RDS API

在使用 CreateDBInstance 操作 Amazon RDS API 操作创建新数据库实例时，通过将 `EnablePerformanceInsights` 设置为 `True` 来打开性能详情。要关闭性能详情，请将 `EnablePerformanceInsights` 设置为 `False` 并将 `DatabaseInsightsMode` 设置为 `standard`。

您还可以使用以下 API 操作指定 `EnablePerformanceInsights` 值：

- CreateDBCluster（多可用区数据库集群）
- ModifyDBCluster（多可用区数据库集群）
- ModifyDBInstance
- CreateDBInstanceReadReplica
- RestoreDBInstanceFromS3

在打开性能详情时，可以通过 `PerformanceInsightsRetentionPeriod` 参数指定性能详情数据的保留时间，以天为单位。您可以指定 7、*month* * 31（其中 *month* 为 1–23 之间的数字）或 731。例如，如果您想将性能数据保留 3 个月，请指定 93，也就是 3 * 31。默认值为 7 天。有关保留期的更多信息，请参阅性能详情的定价和数据留存。

Amazon RDS for MariaDB 或 Amazon RDS for MySQL 上性能详情的性能架构概述

Performance Schema 是一项可选功能，用来监控 Amazon RDS for MariaDB 或 Amazon RDS for MySQL 低细节层次的运行时性能。性能架构旨在将对数据库性能的影响降至最低。Performance Insights 是一个单独的功能，无论有没有 Performance Schema，您都可以使用该功能。

主题

性能架构概览

Performance Schema 监控 MariaDB 和 MySQL 数据库中的事件。事件是一项消耗时间的数据库服务器操作，已进行了分析，以便收集计时信息。以下为事件示例：

- 函数调用
- 等待操作系统
- SQL 执行阶段
- SQL 语句组

`PERFORMANCE_SCHEMA` 存储引擎是实施 Performance Schema 功能的一种机制。该引擎使用数据库源代码中的分析来收集事件数据。该引擎将事件存储在 `performance_schema` 数据库的 memory-only 表中。您可以查询 `performance_schema`，就像您可以查询任何其他表一样。有关更多信息，请参阅 MySQL 参考手册中的 MySQL Performance Schema。

性能详情和性能架构

Performance Insights 和 Performance Schema 是独立的功能，但它们相互关联。Amazon RDS for MariaDB 或 MySQL 的性能详情的行为取决于性能架构是否已开启，如果已开启，则取决于性能详情是否自动管理性能架构。下表描述了该行为。

性能架构已开启	性能详情管理模式	性能详情行为
是	自动	• 收集详细的低级别监控信息 • 每秒收集活动会话指标 • 显示按详细等待事件分类的数据库负载，您可以用它来识别瓶颈
是	手动	• 收集等待事件和每 SQL 指标 • 每秒收集活动会话指标 • 报告用户状态（如插入和发送），这不能帮助您识别瓶颈

性能架构已开启	性能详情管理模式	性能详情行为
否	不适用	• 不收集等待事件、按 SQL 指标或其他详细的低级别监控信息 • 每五秒收集活动会话指标，而不是每秒收集一次 • 报告用户状态（如插入和发送），这不能帮助您识别瓶颈

让 Performance Insights 自动管理 Performance Schema

在打开 Performance Insights 的情况下创建 Amazon RDS for MariaDB 或 Amazon RDS for MySQL 数据库实例时，也会打开 Performance Schema。在这种情况下，性能详情会自动管理您的性能架构参数。这是建议的配置。

当 Performance Insights 自动管理 Performance Schema 时，`performance_schema` 的来源是 `System default`。

> **Note**
>
> t4g.medium 实例类不支持自动管理性能架构。

如果手动更改 `performance_schema` 参数值，然后稍后想恢复为自动管理，请参阅[为 Amazon RDS for MariaDB 或 Amazon RDS for MySQL 开启 Performance Schema]()。

> **Important**
>
> 当 Performance Insights 打开 Performance Schema 时，它不会更改参数组值。但是，正在运行的数据库实例上的值会更改。查看更改后的值的唯一方法是运行 `SHOW GLOBAL VARIABLES` 命令。

重启对 Performance Schema 的影响

Performance Insights 和 Performance Schema 对数据库实例重启的要求不同：

Performance Schema

要打开或关闭此功能，您必须重启数据库实例。

性能详情

要打开或关闭此功能，您不需要重启数据库实例。

如果当前未打开 Performance Schema，并且您在不重启数据库实例的情况下打开 Performance Insights，则不会打开 Performance Schema。

确定性能详情是否正在管理 Performance Schema

要了解性能详情当前是否正在管理所有受支持引擎版本的 Performance Schema，请查看下表。

performance_schema 参数设置	“源”列设置	性能详情正在管理 Performance Schema？
`0`	`System default`	是
`0` 或者 `1`	`Modified`	否

在以下过程中，您可以确定性能详情是否正在自动管理 Performance Schema。

自动确定性能详情是否正在管理 Performance Schema

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 选择参数组。
3. 选择数据库实例的参数组名称。
4. 在搜索栏中输入 **performance_schema**。
5. 检查源是否为系统默认设置，以及值是否为 0。如果是这样，性能详情将自动管理 Performance Schema。

   在此处所示的示例中，性能详情未自动管理 Performance Schema。

为 Amazon RDS for MariaDB 或 Amazon RDS for MySQL 开启 Performance Schema

假设已为数据库实例或多可用区数据库集群打开性能详情，但当前未管理 Performance Schema。如果要让性能详情自动管理 Performance Schema，请完成以下步骤。

配置 Performance Schema 以进行自动管理

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 选择参数组。
3. 选择数据库实例或多可用区数据库集群的参数组名称。
4. 选择编辑。
5. 在搜索栏中输入 **`performance_schema`**。
6. 选择 `performance_schema` 参数。
7. 选择设置为默认值。
8. 选择将值设置为默认值进行确认。
9. 选择保存更改。
10. 重启数据库实例或多可用区数据库集群。

> **Important**
>
> 无论何时打开或关闭 Performance Schema，都要确保重启数据库实例或多可用区数据库集群。

有关修改实例参数的更多信息，请参阅在 Amazon RDS 中修改数据库参数组中的参数。有关控制面板的更多信息，请参阅 使用 Performance Insights 控制面板分析指标。有关 MySQL Performance Schema 的更多信息，请参阅 MySQL 文档中的 MySQL Performance Schema（适用于 8.0）和 MySQL Performance Schema（适用于 8.4）。

为 Performance Insights 配置访问策略

要访问 Performance Insights，主体必须拥有 Amazon Identity and Access Management (IAM) 的相应权限。您可以通过以下方式授予访问权限：

- 将 `AmazonRDSPerformanceInsightsReadOnly` 托管式策略附加到权限集或角色，以访问 Performance Insights API 的所有只读操作。附加以下 CloudWatch 权限：`GetMetricStatistics`、`ListMetrics` 和 `GetMetricData`。有关 CloudWatch 权限的更多信息，请参阅 [Amazon CloudWatch 权限参考]。
- 将 `AmazonRDSPerformanceInsightsFullAccess` 托管式策略附加到权限集或角色，以访问 Performance Insights API 的所有操作。附加以下 CloudWatch 权限：`GetMetricStatistics`、`ListMetrics` 和 `GetMetricData`。有关 CloudWatch 权限的更多信息，请参阅 [Amazon CloudWatch 权限参考]。
- 创建自定义 IAM 策略并将其附加到权限集或角色。

如果您在开启性能详情时指定了客户自主管理型密钥，请确保账户中的用户对 Amazon KMS key具有 `kms:Decrypt` 和 `kms:GenerateDataKey` 权限。

在以下各节中，将 Amazon 托管式策略附加到 IAM 主体，创建自定义 IAM 策略，更改 Amazon KMS 策略，并授予对性能详情的精细访问权限。

主题

将 AmazonRDSPerformanceInsightsReadOnly 策略附加到 IAM 主体

`AmazonRDSPerformanceInsightsReadOnly` 是 Amazon 托管策略，可以授予对 Amazon RDS Performance Insights API 的所有只读操作的访问权限。

如果将 `AmazonRDSPerformanceInsightsReadOnly` 附加到权限集或角色，接收人可以使用 Performance Insights 以及其他控制台功能。

有关更多信息，请参阅 Amazon 托管式策略：AmazonRDSPerformanceInsightsReadOnly。

将 AmazonRDSPerformanceInsightsFullAccess 策略附加到 IAM 主体

`AmazonRDSPerformanceInsightsFullAccess` 是 Amazon 托管策略，可以授予对 Amazon RDS Performance Insights API 的所有操作的访问权限。

如果将 `AmazonRDSPerformanceInsightsFullAccess` 附加到权限集或角色，接收人可以使用 Performance Insights 以及其他控制台功能。

有关更多信息，请参阅 Amazon 托管式策略：AmazonRDSPerformanceInsightsFullAccess。

为 Performance Insights 创建自定义 IAM 策略

对于没有 `AmazonRDSPerformanceInsightsReadOnly` 或 `AmazonRDSPerformanceInsightsFullAccess` 策略的用户，可以通过创建或修改用户托管式 IAM 策略，来授予对 Performance Insights 的访问权限。在将策略附加到 IAM 权限集或角色时，接收人可以使用 Performance Insights。

创建自定义策略

1. 打开 IAM 控制台：https://console.aws.amazon.com/iam/。
2. 在导航窗格中，选择策略。
3. 选择创建策略。
4. 在创建策略页面上，选择 JSON 选项。
5. 复制并粘贴《Amazon 托管策略参考指南》中的 JSON 策略文档部分提供的有关 AmazonRDSPerformanceInsightsReadOnly 或 AmazonRDSPerformanceInsightsFullAccess 策略的文本。
6. 选择查看策略。
7. 为策略提供名称并可以选择提供描述，然后选择创建策略。

现在，可以将策略附加到权限集或角色。以下过程假设您已经有一个可用于此目的的用户。

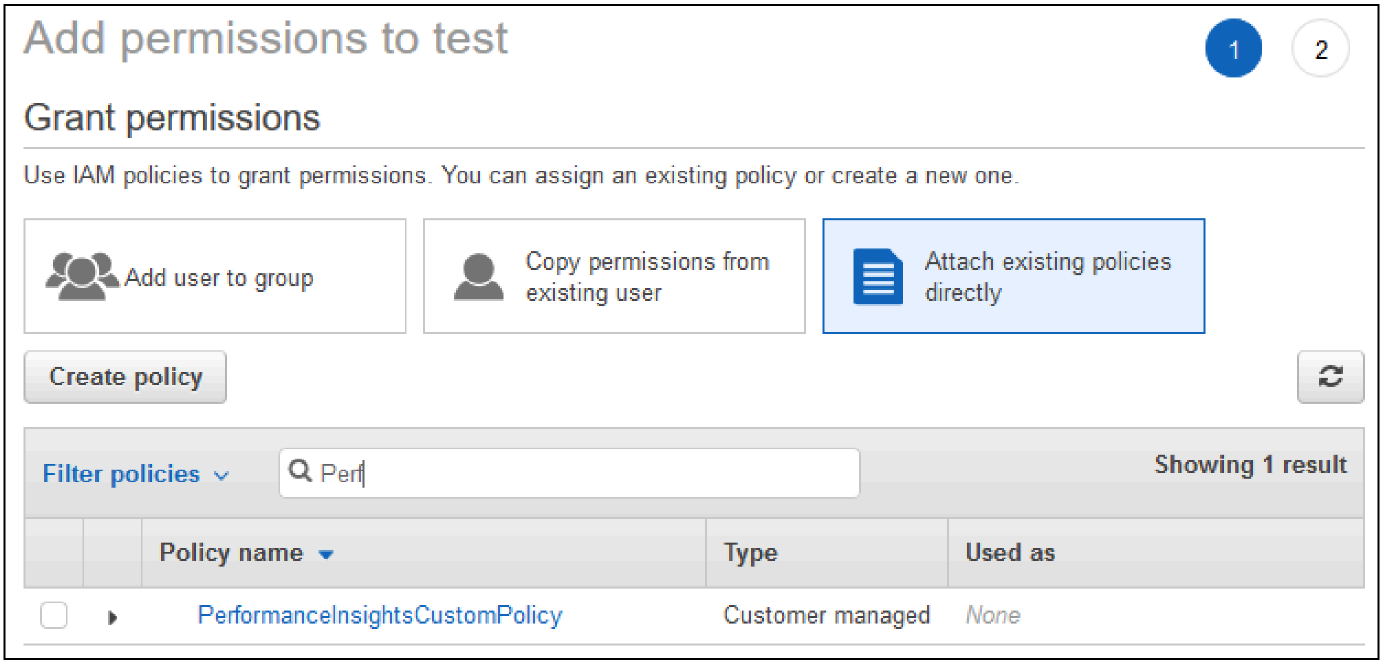
将策略附加到用户

1. 打开 IAM 控制台：https://console.aws.amazon.com/iam/。
2. 在导航窗格中，选择 Users。

3. 从列表中选择现有用户。

> **Important**
>
> 要使用 Performance Insights，请确保除了自定义策略之外，您还有权访问 Amazon RDS。例如，`AmazonRDSPerformanceInsightsReadOnly` 预定义策略提供了对 Amazon RDS 的只读访问权限。有关更多信息，请参阅“使用策略管理访问”。

4. 在 Summary (摘要) 页上，选择 Add permissions (添加权限)。
5. 选择直接附加现有策略。对于搜索，键入策略名称的前几个字符，如下图所示。



6. 选择策略，然后选择 Next: Review。
7. 选择 Add permissions (添加权限)。

更改性能详情的 Amazon KMS 策略

Performance Insights 使用 Amazon KMS key 加密敏感数据。通过 API 或控制台启用 Performance Insights 时，您可以执行以下任一操作：

- 选择默认 Amazon 托管式密钥。

  Amazon RDS 为新的数据库实例使用 Amazon 托管式密钥。Amazon RDS 将为您的 Amazon Web Services 账户创建 Amazon 托管式密钥。您的 Amazon Web Services 账户在每个 Amazon Web Services 区域都有用于 Amazon RDS 的不同 Amazon 托管式密钥。

- 选择客户托管密钥。

  如果您指定一个客户托管密钥，则您账户中调用 Performance Insights API 的用户需要在 KMS 密钥具有 `kms:Decrypt` 和 `kms:GenerateDataKey` 权限。您可以通过 IAM policy 配置这些权限。但是，我们建议您通过 KMS 密钥策略来管理这些权限。有关更多信息，请参阅《Amazon Key Management Service 开发人员指南》中的在 Amazon KMS 中使用密钥策略。

Example

以下示例显示了如何将语句添加到 KMS 密钥策略。这些语句可以访问 Performance Insights。您可能需要更改一些限制，这取决于您使用 KMS 密钥的方式。在将语句添加到您的策略之前，请删除所有注释。

```
{
"Version" : "2012-10-17",
 "Id" : "your-policy",
 "Statement" : [ {
    //This represents a statement that currently exists in your policy.
 }
 ....,
 //Starting here, add new statement to your policy for Performance Insights.
 //We recommend that you add one new statement for every RDS instance
{
    "Sid" : "Allow viewing RDS Performance Insights",
    "Effect": "Allow",
    "Principal": {
        "AWS": [
            //One or more principals allowed to access Performance Insights
            "arn:aws:iam::444455556666:role/Role1"
        ]
    },
    "Action": [
        "kms:Decrypt",
        "kms:GenerateDataKey"
    ],
    "Resource": "*",
    "Condition" : {
        "StringEquals" : {
            //Restrict access to only RDS APIs (including Performance Insights).
            //Replace region with your Amazon Region.
            //For example, specify us-west-2.
            "kms:ViaService" : "rds.region.amazonaws.com"
```

```
        },
        "ForAnyValue:StringEquals": {
            //Restrict access to only data encrypted by Performance Insights.
            "kms:EncryptionContext:aws:pi:service": "rds",
            "kms:EncryptionContext:service": "pi",

            //Restrict access to a specific RDS instance.
            //The value is a DbiResourceId.
           "kms:EncryptionContext:aws:rds:db-id": "db-AAAAABBBBBCCCCDDDDDEEEEE"
        }
    }
}
```

Performance Insights 如何使用 Amazon KMS 客户托管密钥

Performance Insights 使用客户托管密钥加密敏感数据。当您开启 Performance Insights 时，可以通过 API 提供 Amazon KMS 密钥。Performance Insights 对此密钥创建 KMS 权限。它使用密钥并执行必要的操作来处理敏感数据。敏感数据包括用户、数据库、应用程序和 SQL 查询文本等字段。Performance Insights 可确保数据在静态和动态时都保持加密状态。

Performance Insights IAM 如何使用 Amazon KMS

IAM 提供针对特定 API 的权限。Performance Insights 具有以下公有 API，您可以使用 IAM 策略对其进行限制：

- `DescribeDimensionKeys`
- `GetDimensionKeyDetails`
- `GetResourceMetadata`
- `GetResourceMetrics`
- `ListAvailableResourceDimensions`
- `ListAvailableResourceMetrics`

您可以使用以下 API 请求来获取敏感数据。

- `DescribeDimensionKeys`
- `GetDimensionKeyDetails`
- `GetResourceMetrics`

当您使用 API 获取敏感数据时，Performance Insights 会利用调用方的凭证。此检查可确保敏感数据的访问权限仅限于有权访问 KMS 密钥的用户。

调用这些 API 时，您需要通过 IAM 策略调用 API 的权限，以及通过 Amazon KMS 密钥策略调用 `kms:decrypt` 操作的权限。

`GetResourceMetrics` API 可以返回敏感和非敏感数据。请求参数决定响应是否应包含敏感数据。当请求的筛选条件或分组依据参数中包含敏感维度时，API 会返回敏感数据。

有关可以与 `GetResourceMetrics` API 一起使用的维度的更多信息，请参阅 DimensionGroup。

Example 示例

以下示例请求 `db.user` 组的敏感数据：

```
POST / HTTP/1.1
Host: <Hostname>
Accept-Encoding: identity
X-Amz-Target: PerformanceInsightsv20180227.GetResourceMetrics
Content-Type: application/x-amz-json-1.1
User-Agent: <UserAgentString>
X-Amz-Date: <Date>
Authorization: AWS4-HMAC-SHA256 Credential=<Credential>, SignedHeaders=<Headers>,
 Signature=<Signature>
Content-Length: <PayloadSizeBytes>
{
  "ServiceType": "RDS",
  "Identifier": "db-ABC1DEFGHIJKL2MNOPQRSTUV3W",
  "MetricQueries": [
    {
      "Metric": "db.load.avg",
      "GroupBy": {
        "Group": "db.user",
        "Limit": 2
      }
    }
  ],
  "StartTime": 1693872000,
  "EndTime": 1694044800,
  "PeriodInSeconds": 86400
}
```

Example

以下示例请求 `db.load.avg` 指标的非敏感数据：

```
POST / HTTP/1.1
Host: <Hostname>
Accept-Encoding: identity
X-Amz-Target: PerformanceInsightsv20180227.GetResourceMetrics
Content-Type: application/x-amz-json-1.1
User-Agent: <UserAgentString>
X-Amz-Date: <Date>
Authorization: AWS4-HMAC-SHA256 Credential=<Credential>, SignedHeaders=<Headers>,
 Signature=<Signature>
Content-Length: <PayloadSizeBytes>
{
    "ServiceType": "RDS",
    "Identifier": "db-ABC1DEFGHIJKL2MNOPQRSTUV3W",
    "MetricQueries": [
        {
            "Metric": "db.load.avg"
        }
    ],
    "StartTime": 1693872000,
    "EndTime": 1694044800,
    "PeriodInSeconds": 86400
}
```

授予 Performance Insights 精细访问权限

精细访问控制功能提供用于控制对 Performance Insights 的访问的其他方法。此访问控制功能可以允许或拒绝对 `GetResourceMetrics`、`DescribeDimensionKeys` 和 `GetDimensionKeyDetails` Performance Insights 操作的各个维度的访问。要使用精细访问功能，请使用条件键在 IAM 策略中指定维度。访问权限的评估遵循 IAM 策略评估逻辑。有关更多信息，请参阅《IAM 用户指南》中的策略评估逻辑。如果 IAM 策略声明未指定任何维度，该声明将控制对指定操作的所有维度的访问。有关可用维度的列表，请参阅 DimensionGroup。

要找出您的凭证有权访问的维度，请使用 `ListAvailableResourceDimensions` 中的 `AuthorizedActions` 参数并指定操作。允许的 `AuthorizedActions` 值如下所示：

- `GetResourceMetrics`

- DescribeDimensionKeys
- GetDimensionKeyDetails

例如，如果您为 AuthorizedActions 参数指定 GetResourceMetrics，ListAvailableResourceDimensions 将返回 GetResourceMetrics 操作有权访问的维度列表。如果您在 AuthorizedActions 参数中指定了多个操作，ListAvailableResourceDimensions 将返回这些操作有权访问的维度的交集。

Example

以下示例提供了对 GetResourceMetrics 和 DescribeDimensionKeys 操作的指定维度的访问权限。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowToDiscoverDimensions",
            "Effect": "Allow",
            "Action": [
                "pi:ListAvailableResourceDimensions"
            ],
            "Resource": [
                "arn:aws:pi:us-east-1:123456789012:metrics/rds/db-
ABC1DEFGHIJKL2MNOPQRSTUV3W"
            ]
        },
        {
            "Sid": "SingleAllow",
            "Effect": "Allow",
            "Action": [
                "pi:GetResourceMetrics",
                "pi:DescribeDimensionKeys"
            ],
            "Resource": [
                "arn:aws:pi:us-east-1:123456789012:metrics/rds/db-
ABC1DEFGHIJKL2MNOPQRSTUV3W"
            ],
            "Condition": {
                "ForAllValues:StringEquals": {
                    // only these dimensions are allowed. Dimensions not included in
                    // a policy with "Allow" effect will be denied
```

```
                    "pi:Dimensions": [
                        "db.sql_tokenized.id",
                        "db.sql_tokenized.statement"
                    ]
                }
            }
        }


    ]
}
```

以下是对所请求维度的响应：

```

 // ListAvailableResourceDimensions API
// Request
{
    "ServiceType": "RDS",
    "Identifier": "db-ABC1DEFGHIJKL2MNOPQRSTUV3W",
    "Metrics": [ "db.load" ],
    "AuthorizedActions": ["DescribeDimensionKeys"]
}

// Response
{
    "MetricDimensions": [ {
        "Metric": "db.load",
        "Groups": [
            {
                "Group": "db.sql_tokenized",
                "Dimensions": [
                    { "Identifier": "db.sql_tokenized.id" },
                  //  { "Identifier": "db.sql_tokenized.db_id" }, // not included
 because not allows in the IAM Policy
                    { "Identifier": "db.sql_tokenized.statement" }
                ]
            }

        ] }
    ]
}
```

以下示例为维度指定一个允许访问和两个拒绝访问权限。

```
{
    "Version": "2012-10-17",
    "Statement": [
          {
            "Sid": "AllowToDiscoverDimensions",
            "Effect": "Allow",
            "Action": [
                "pi:ListAvailableResourceDimensions"
            ],
            "Resource": [
                "arn:aws:pi:us-east-1:123456789012:metrics/rds/db-
ABC1DEFGHIJKL2MNOPQRSTUV3W"
            ]
          },

          {
            "Sid": "O01AllowAllWithoutSpecifyingDimensions",
            "Effect": "Allow",
            "Action": [
                "pi:GetResourceMetrics",
                "pi:DescribeDimensionKeys"
            ],
            "Resource": [
                "arn:aws:pi:us-east-1:123456789012:metrics/rds/db-
ABC1DEFGHIJKL2MNOPQRSTUV3W"
            ]
        },

        {
            "Sid": "O01DenyAppDimensionForAll",
            "Effect": "Deny",
            "Action": [
                "pi:GetResourceMetrics",
                "pi:DescribeDimensionKeys"
            ],
            "Resource": [
                "arn:aws:pi:us-east-1:123456789012:metrics/rds/db-
ABC1DEFGHIJKL2MNOPQRSTUV3W"
            ],
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "pi:Dimensions": [
```

```
                        "db.application.name"
                    ]
                }
            }
        },

        {
            "Sid": "O01DenySQLForGetResourceMetrics",
            "Effect": "Deny",
            "Action": [
                "pi:GetResourceMetrics"
            ],
            "Resource": [
                "arn:aws:pi:us-east-1:123456789012:metrics/rds/db-
ABC1DEFGHIJKL2MNOPQRSTUV3W"
            ],
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "pi:Dimensions": [
                        "db.sql_tokenized.statement"
                    ]
                }
            }
        }
    ]
}
```

以下是对请求维度的响应：

```
   // ListAvailableResourceDimensions API
// Request
{
    "ServiceType": "RDS",
    "Identifier": "db-ABC1DEFGHIJKL2MNOPQRSTUV3W",
    "Metrics": [ "db.load" ],
    "AuthorizedActions": ["GetResourceMetrics"]
}

// Response
{
    "MetricDimensions": [ {
```

```
        "Metric": "db.load",
        "Groups": [
            {
                "Group": "db.application",
                "Dimensions": [

                  // removed from response because denied by the IAM Policy
                  //  { "Identifier": "db.application.name" }
                ]
            },
            {
                "Group": "db.sql_tokenized",
                "Dimensions": [
                    { "Identifier": "db.sql_tokenized.id" },
                    { "Identifier": "db.sql_tokenized.db_id" },

                  // removed from response because denied by the IAM Policy
                  //  { "Identifier": "db.sql_tokenized.statement" }
                ]
            },
            ...
        ] }
    ]
}
```

```
// ListAvailableResourceDimensions API
// Request
{
    "ServiceType": "RDS",
    "Identifier": "db-ABC1DEFGHIJKL2MNOPQRSTUV3W",
    "Metrics": [ "db.load" ],
    "AuthorizedActions": ["DescribeDimensionKeys"]
}

// Response
{
    "MetricDimensions": [ {
        "Metric": "db.load",
        "Groups": [
            {
                "Group": "db.application",
                "Dimensions": [
```

```
                    // removed from response because denied by the IAM Policy
                    //  { "Identifier": "db.application.name" }
                  ]
              },
              {
                  "Group": "db.sql_tokenized",
                  "Dimensions": [
                      { "Identifier": "db.sql_tokenized.id" },
                      { "Identifier": "db.sql_tokenized.db_id" },

                    // allowed for DescribeDimensionKeys because our IAM Policy
                    // denies it only for GetResourceMetrics
                      { "Identifier": "db.sql_tokenized.statement" }
                  ]
              },
              ...
          ] }
      ]
}
```

使用 Performance Insights 控制面板分析指标

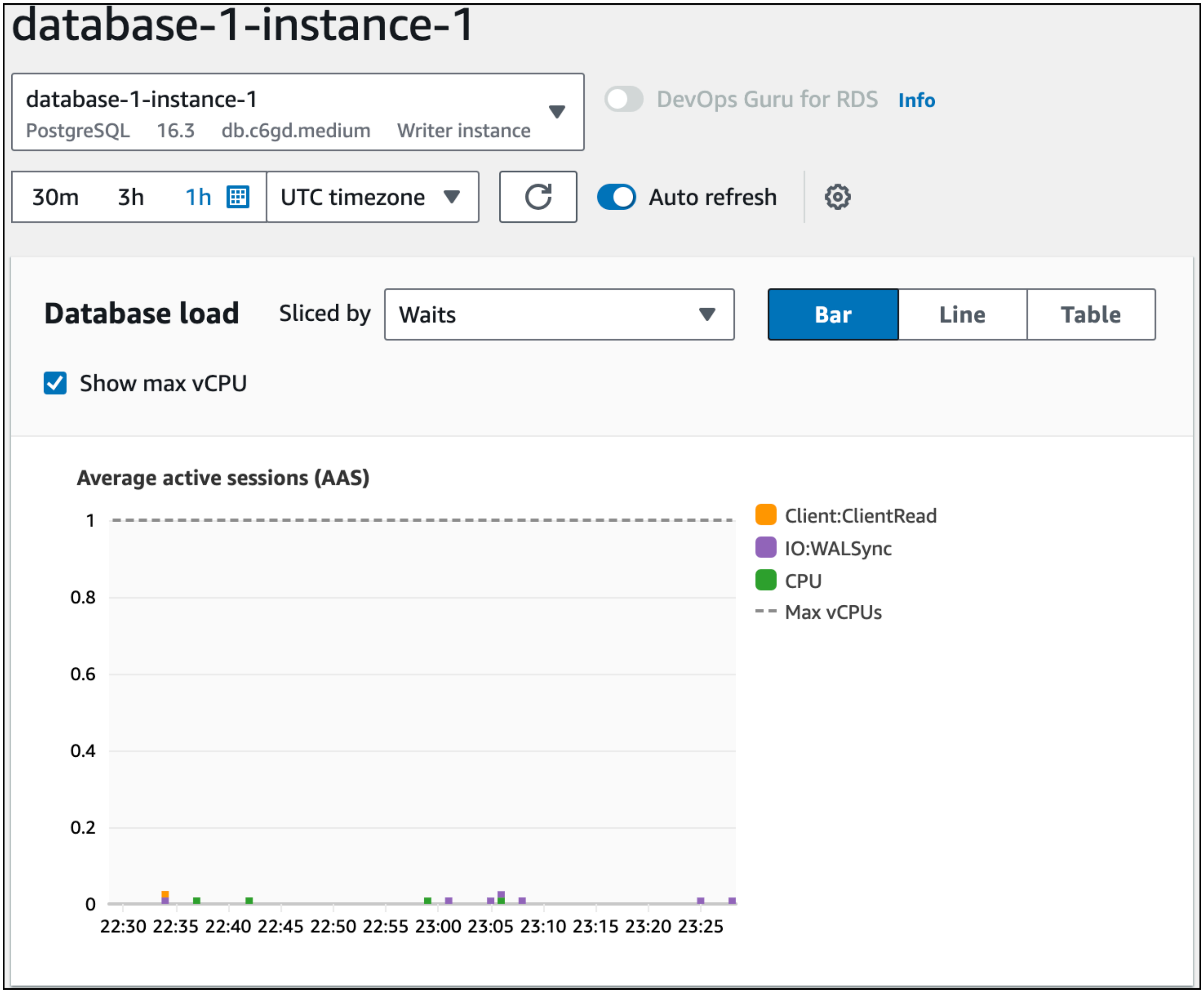
> **Important**
>
> Amazon 已宣布性能详情的生命周期终止日期为：2025 年 11 月 30 日。在此日期之后，Amazon RDS 将不再支持性能详情控制台体验、灵活的保留期（1-24 个月）及其关联定价。性能详情 API 将继续存在，定价不变。性能详情 API 的费用将与 CloudWatch 数据库洞察的费用一起显示在您的 Amazon 账单中。
>
> 我们建议您在 2025 年 11 月 30 日之前，将使用性能详情付费套餐的所有数据库实例升级到数据库洞察的高级模式。有关升级到数据库洞察的高级模式的信息，请参阅开启适用于 Amazon RDS 的数据库洞察的高级模式。
>
> 如果您不采取任何行动，则使用性能详情的数据库实例将默认为使用数据库洞察的标准模式。在数据库洞察的标准模式下，您可能会在 7 天后无法访问性能数据历史记录，并且可能无法在 Amazon RDS 控制台中使用执行计划和按需分析功能。在 2025 年 11 月 30 日之后，只有数据库洞察的高级模式将支持执行计划和按需分析。
>
> 借助 CloudWatch 数据库洞察，您可以监控数据库实例集的数据库负载，并对性能进行大规模分析和故障排除。有关数据库洞察的更多信息，请参阅使用 CloudWatch 数据库洞察监控 Amazon RDS 数据库。有关定价信息，请参阅 Amazon CloudWatch 定价。

Performance Insights 控制面板包含帮助您分析和排查性能问题的数据库性能信息。在主控制面板页面上，可以查看有关数据库负载的信息。您可以按维度（例如等待事件或 SQL）对数据库负载进行“切片”。

性能详情控制面板

性能详情控制面板概览

与性能详情进行交互的最简单方式即为控制面板。以下示例显示了 PostgreSQL 数据库实例的控制面板。

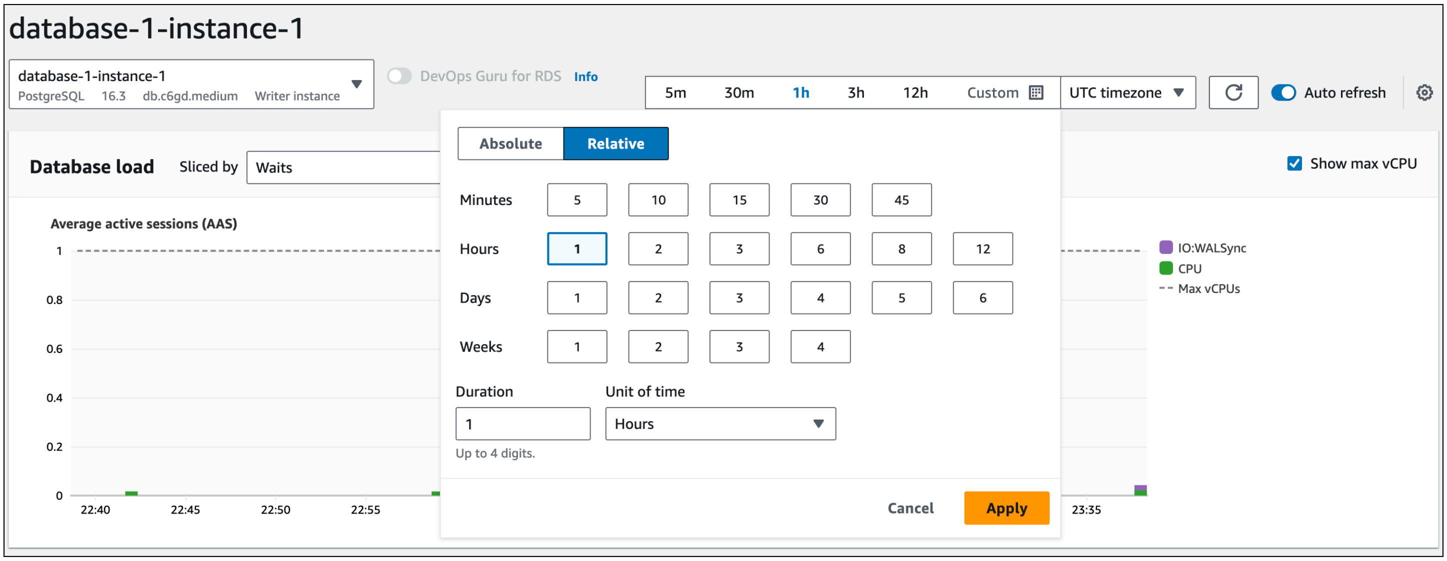


主题

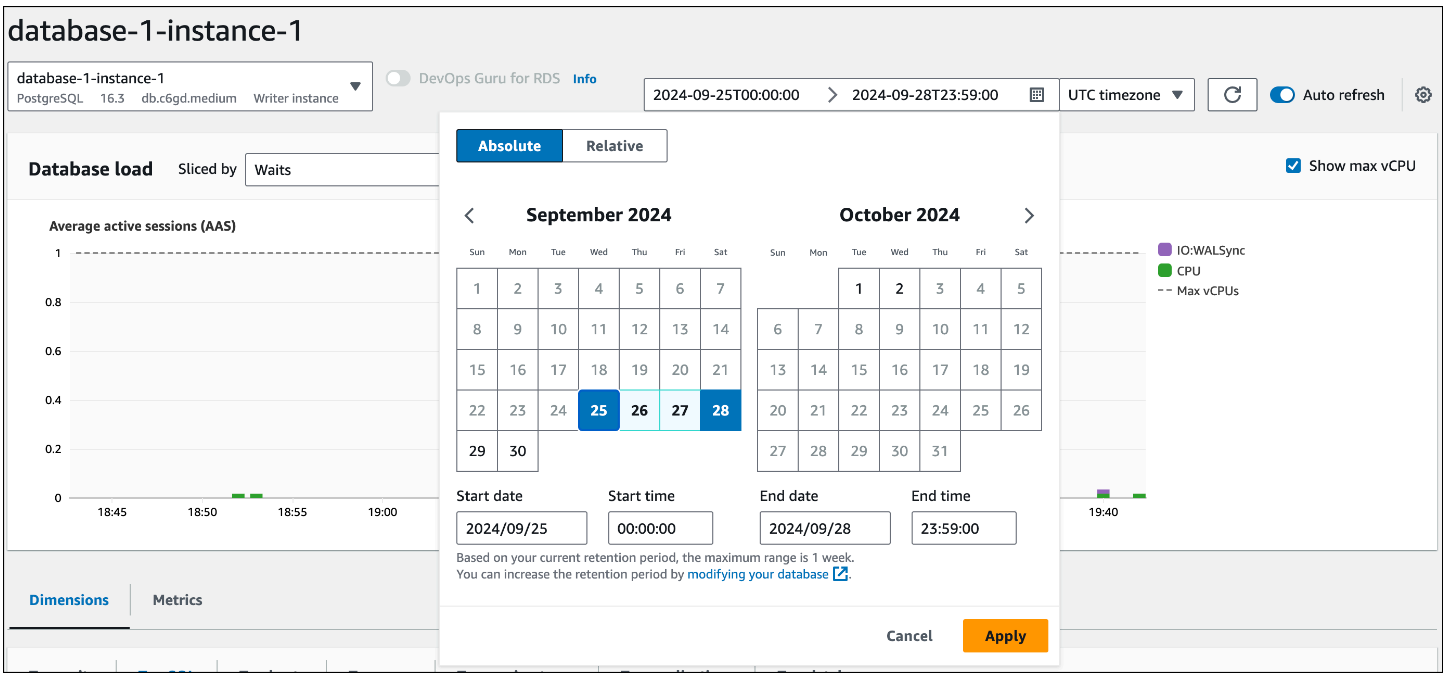
- [时间范围筛选器]
- [计数器指标图表]
- [数据库负载图表]
- [主要维度表]

时间范围筛选器

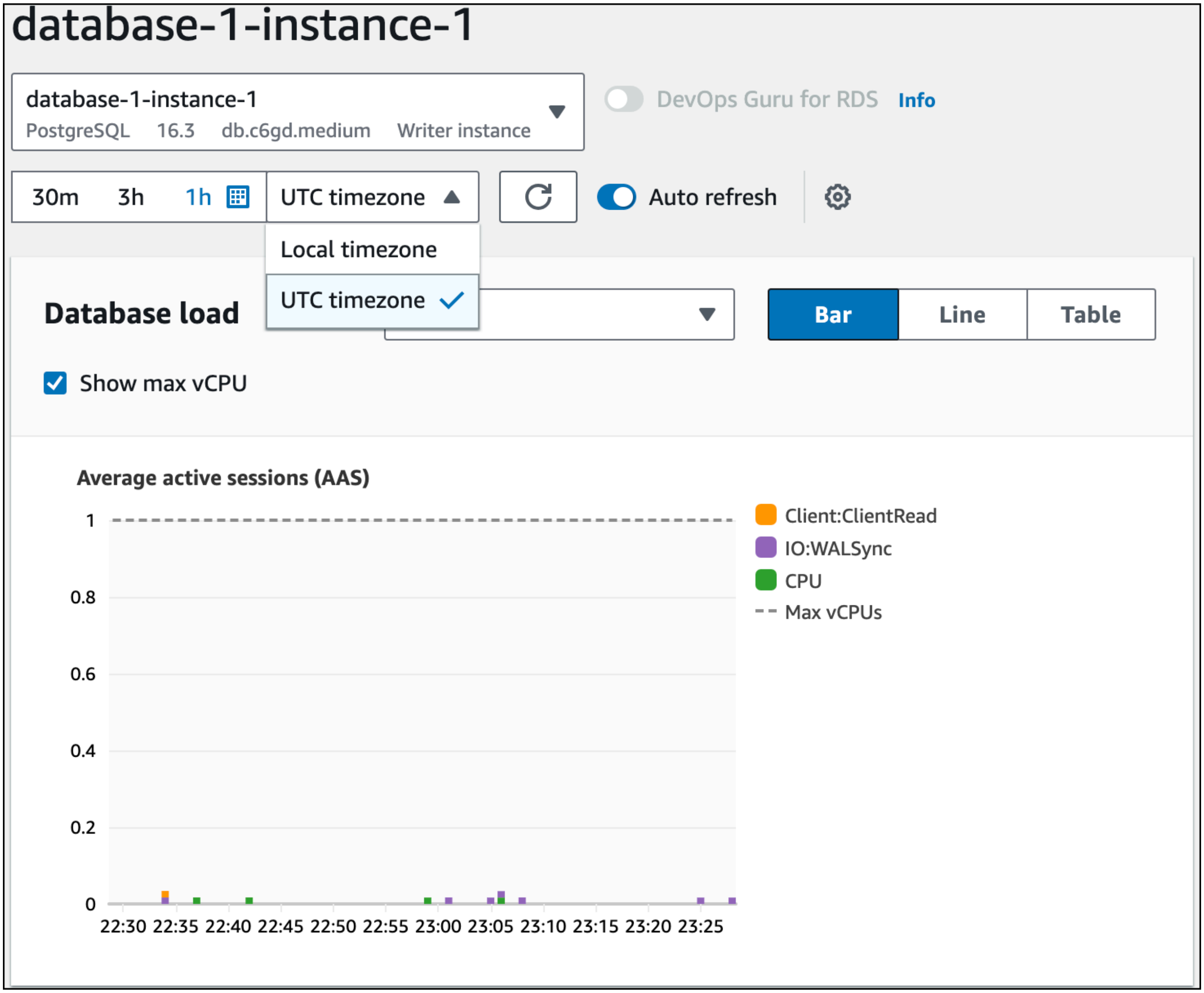
默认情况下，Performance Insights 控制面板将显示最近一小时的数据库负载。您可以将此范围调整为短至 5 分钟或长达 2 年。您也可以选择自定义相对范围。



您可以选择一个具有开始和结束日期和时间的绝对范围。以下示例显示的时间范围从 2024 年 9 月 25 日午夜开始，到 2024 年 9 月 28 日晚上 11:59 结束。



默认情况下，性能详情控制面板的时区为协调世界时（UTC）。也可以选择本地时区。



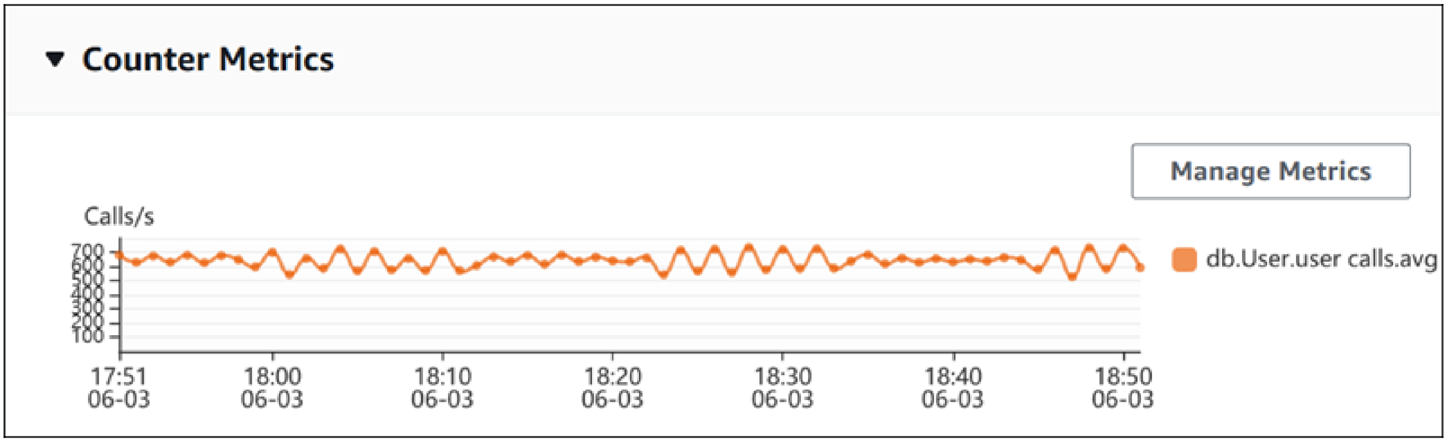
计数器指标图表

使用计数器指标，您可以自定义 Performance Insights 控制面板来包括最多 10 个其他图表。这些图表显示了所选的数十个操作系统和数据库性能指标。您可将此信息与数据库负载相关联，以帮助识别和分析性能问题。

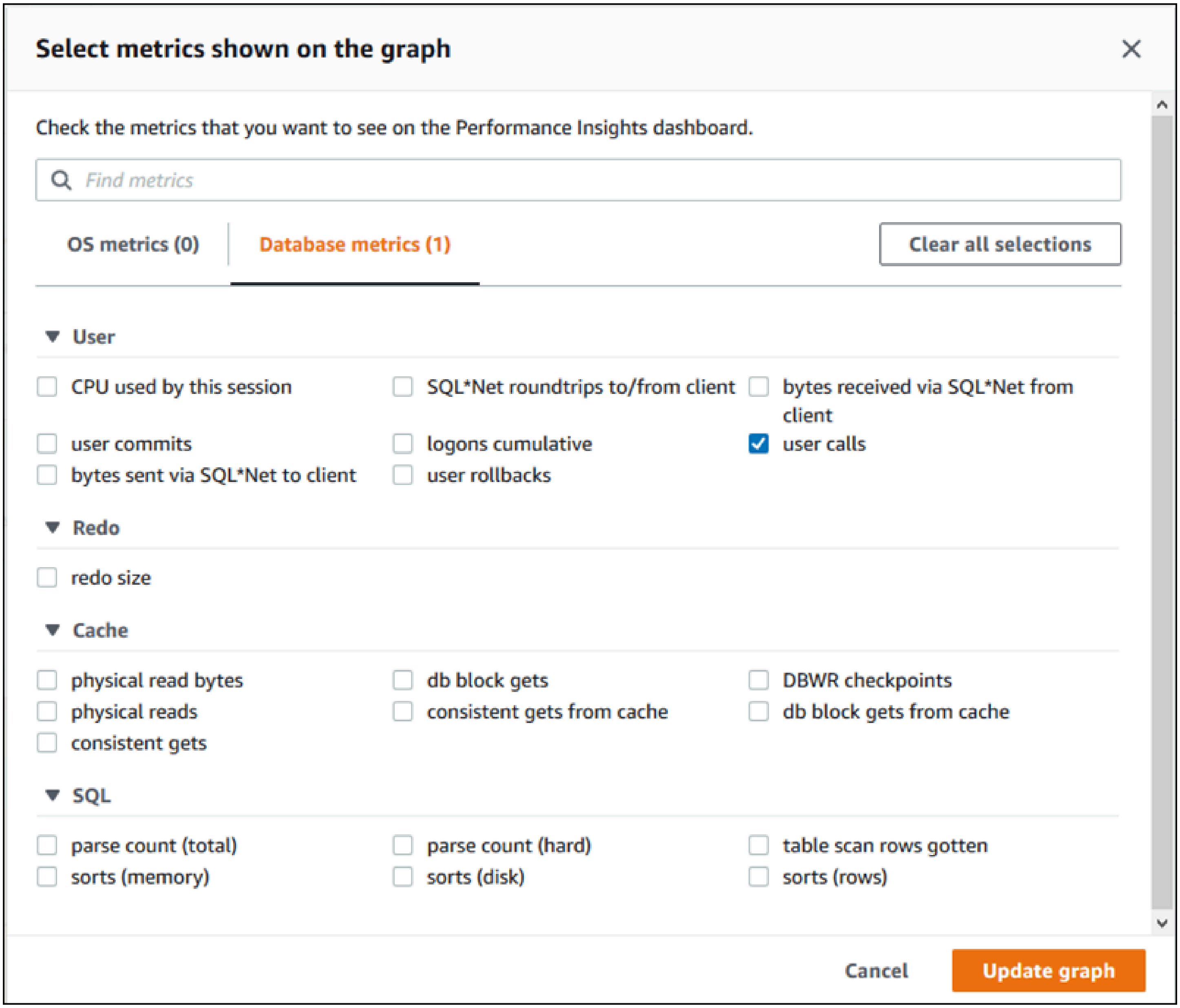
计数器指标图表显示了性能计数器的数据。原定设置指标取决于数据库引擎：

- MySQL 和 MariaDB – `db.SQL.Innodb_rows_read.avg`
- Oracle – `db.User.user calls.avg`
- Microsoft SQL Server – `db.Databases.Active Transactions(_Total).avg`

- PostgreSQL – `db.Transactions.xact_commit.avg`



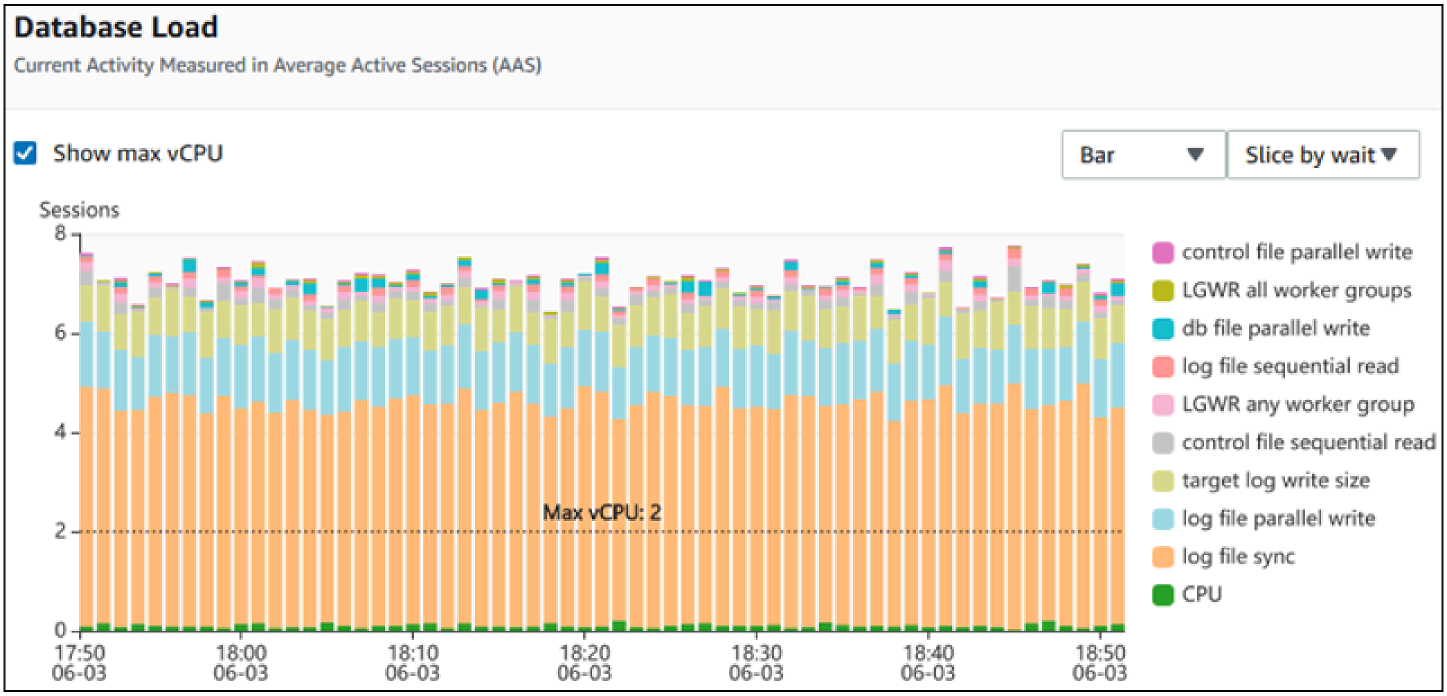
要更改性能计数器，请选择 Manage Metrics（管理指标）。您可以选择多个操作系统指标或数据库指标，如以下屏幕截图所示。要查看任何指标的详细信息，请将鼠标悬停在相应指标名称上。



有关可以为每个数据库引擎添加的计数器指标的说明，请参阅 性能详情计数器指标。

数据库负载图表

数据库负载图表显示数据库负载与最大 vCPU 线表示的数据库实例容量的比较情况。预设情况下，堆叠折线图将以每单位时间的平均活动会话数表示数据库负载。数据库负载按等待状态进行切片（分组）。

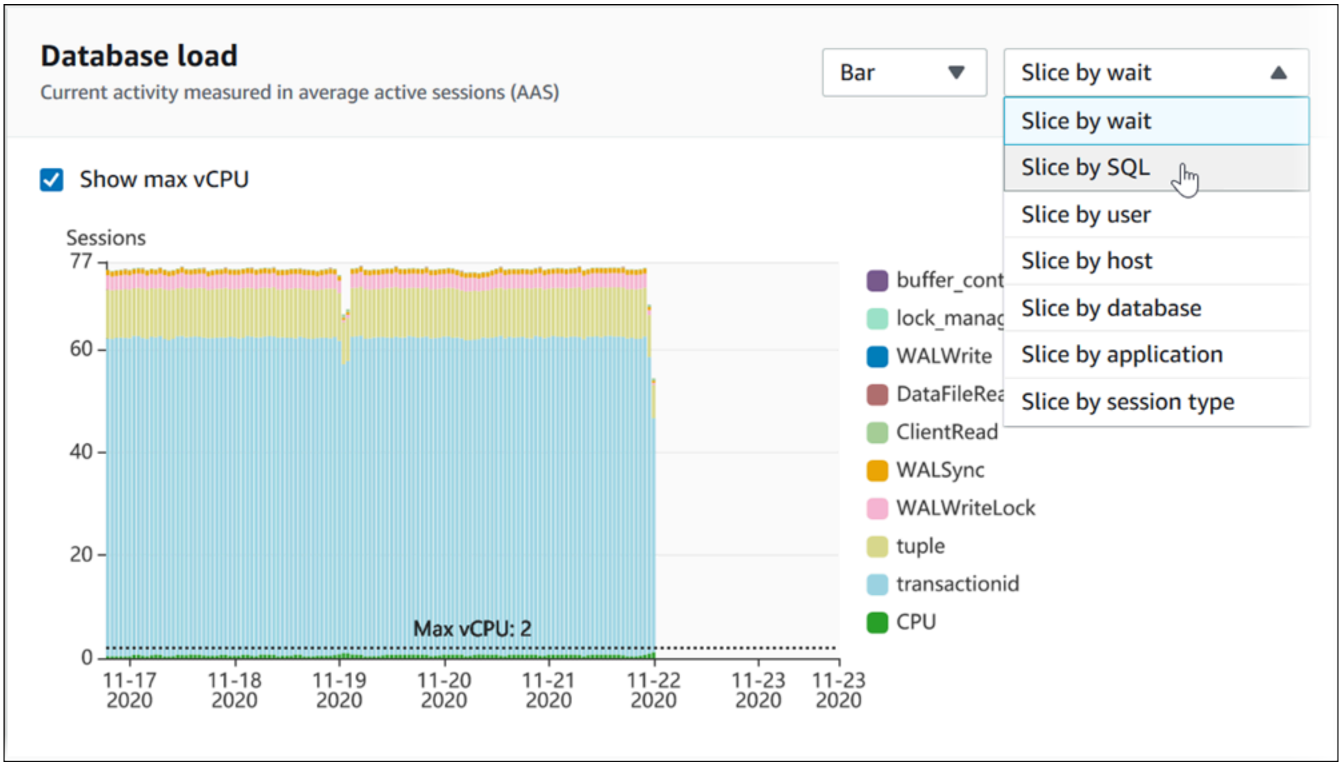


按维度切片的数据库负载

您可以选择按任何受支持维度分组的活动会话显示负载。下表显示了不同引擎支持的维度。

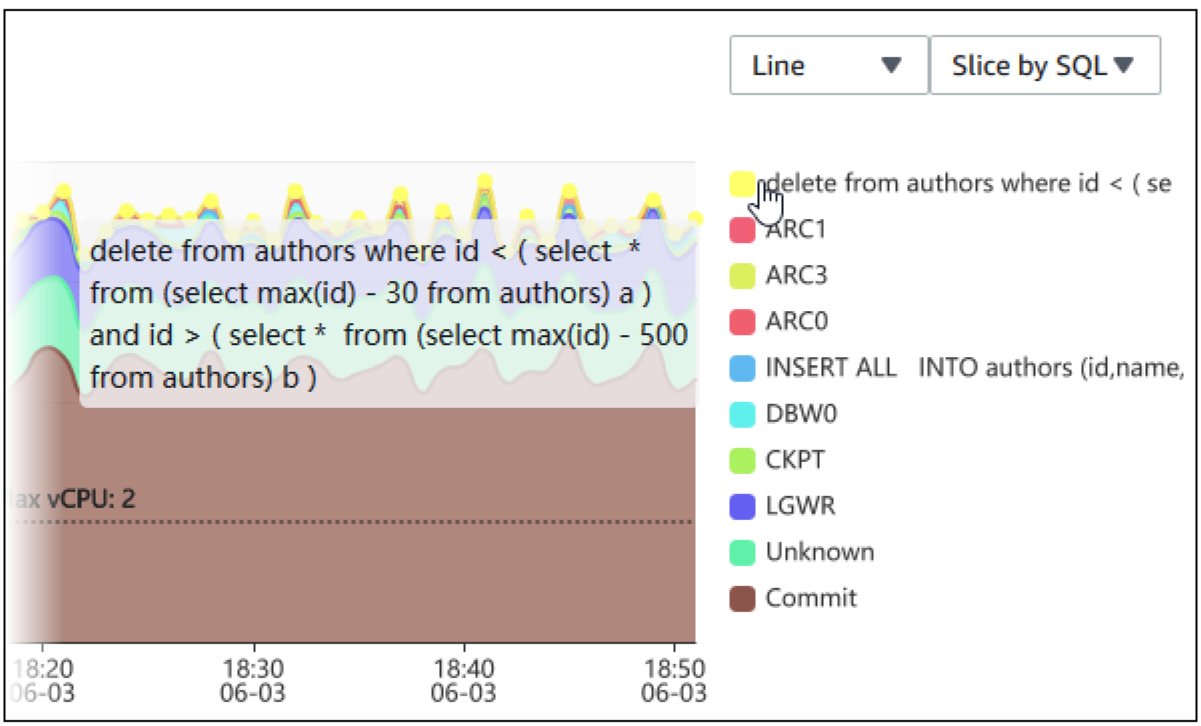
维度	Oracle	SQL Server	PostgreSQL	MySQL
Host	是	是	是	是
SQL	是	是	是	是
User	是	是	是	是
等待	是	是	是	是
计划	是	否	否	否
应用程序	否	否	是	否
数据库	否	否	是	是
会话类型	否	否	是	否

下图显示了 PostgreSQL 数据库实例的维度。

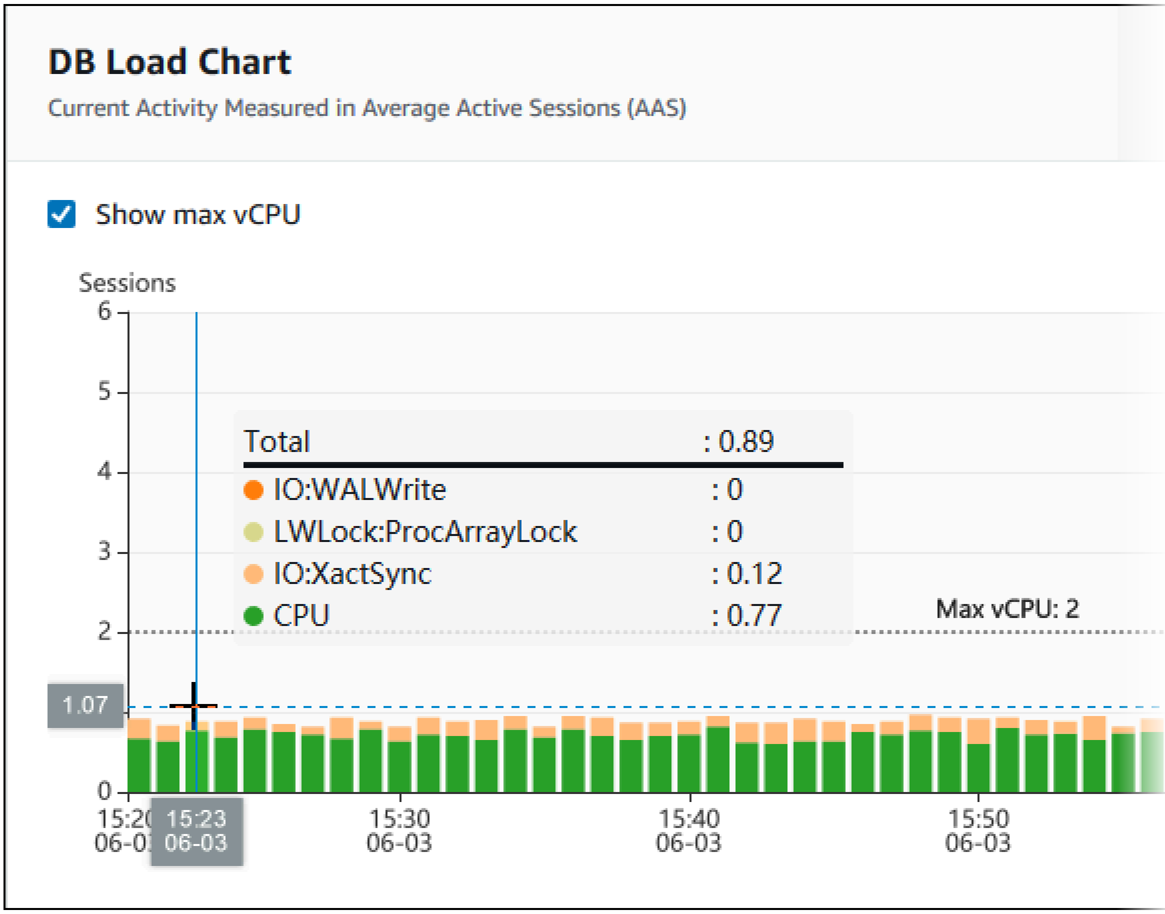


维度项目的数据库负载详细信息

要查看维度中数据库负载项目的详细信息，请将光标悬停在相应项目名称上。下图显示了 SQL 语句的详细信息。

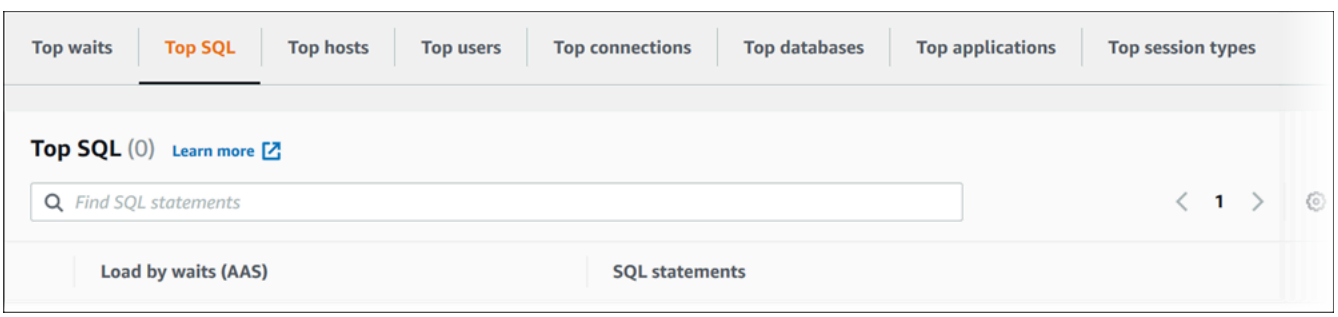


要在图例中查看任何项目在选定时间段内的详细信息，请将鼠标悬停在相应项目上。



主要维度表

主要维度表将按不同的维度切割数据库负载。维度是数据库负载不同特征的类别或“切片依据”。如果维度为 SQL，则 Top SQL（主要 SQL）显示了对数据库负载影响最大的 SQL 语句。



请选择以下任何一个维度选项卡。

选项卡	描述	支持的引擎
主要 SQL	当前正在运行的 SQL 语句	全部
主要等待	数据库后端正在等待的事件	全部
主要主机	所连接客户端的主机名	全部
主要用户	登录到数据库的用户	全部
主要数据库	客户端所连接的数据库的名称	仅限 PostgreSQL、MySQL、MariaDB 和 SQL Server
主要应用程序	连接到数据库的应用程序的名称	仅限 PostgreSQL 和 SQL Server
主要会话类型	当前会话的类型	仅限 PostgreSQL

要了解如何使用 Top SQL（主要 SQL）选项卡分析查询，请参阅 “Top SQL”（主要 SQL）选项卡概览。

访问性能详情控制面板

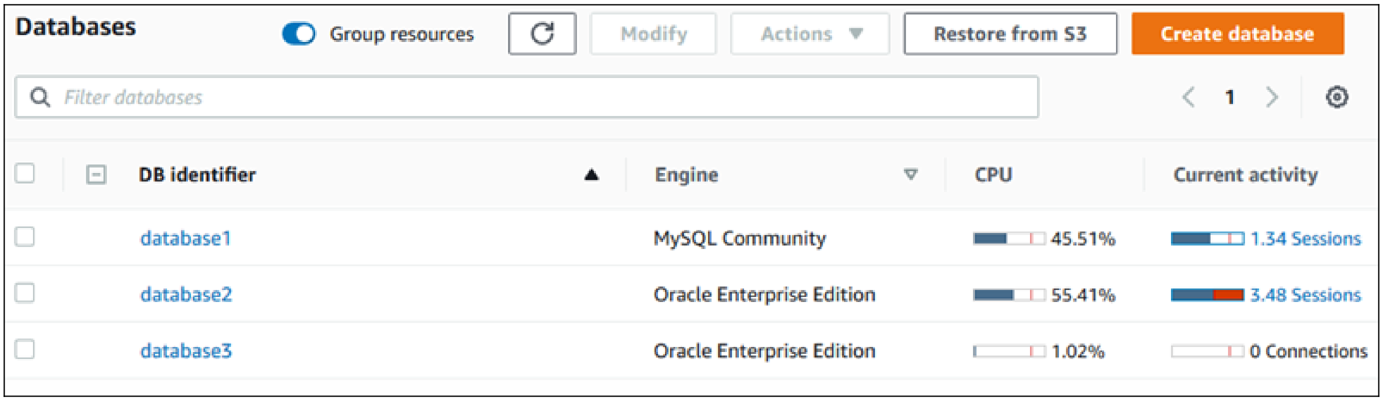
Amazon RDS 在性能详情控制面板中提供性能详情和 CloudWatch 指标的合并视图。

要访问性能详情控制面板，请使用以下过程。

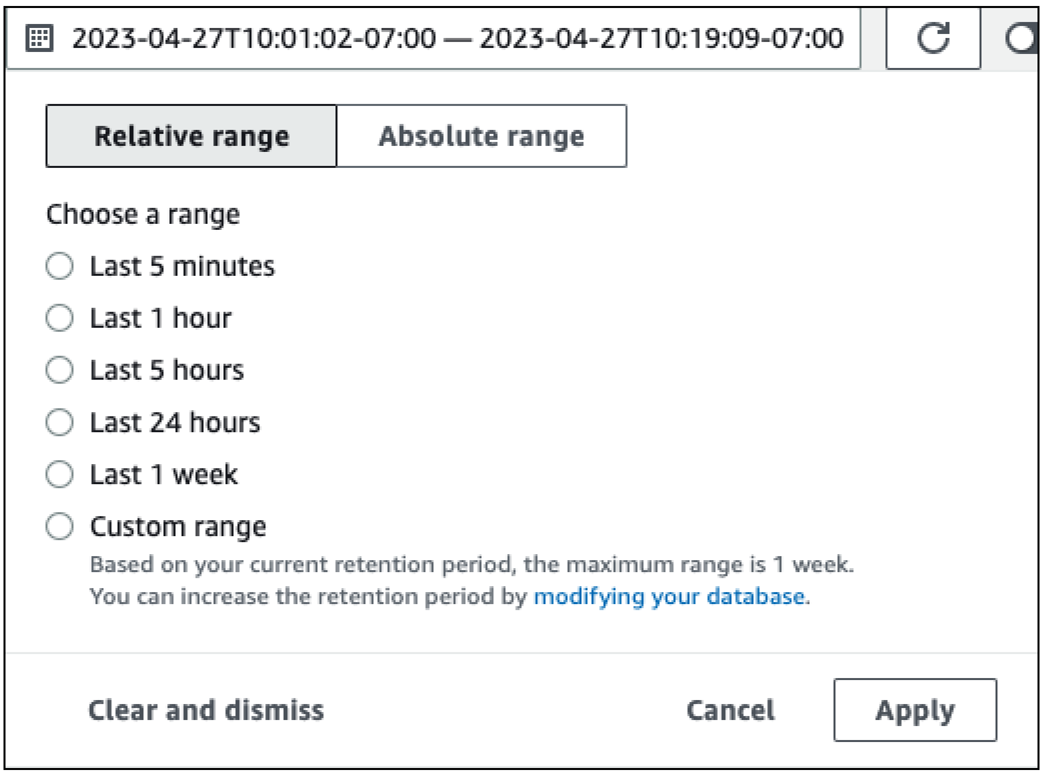
在 Amazon 管理控制台中查看性能详情控制面板

1. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在左侧导航窗格中，选择性能详情。
3. 选择一个数据库实例。

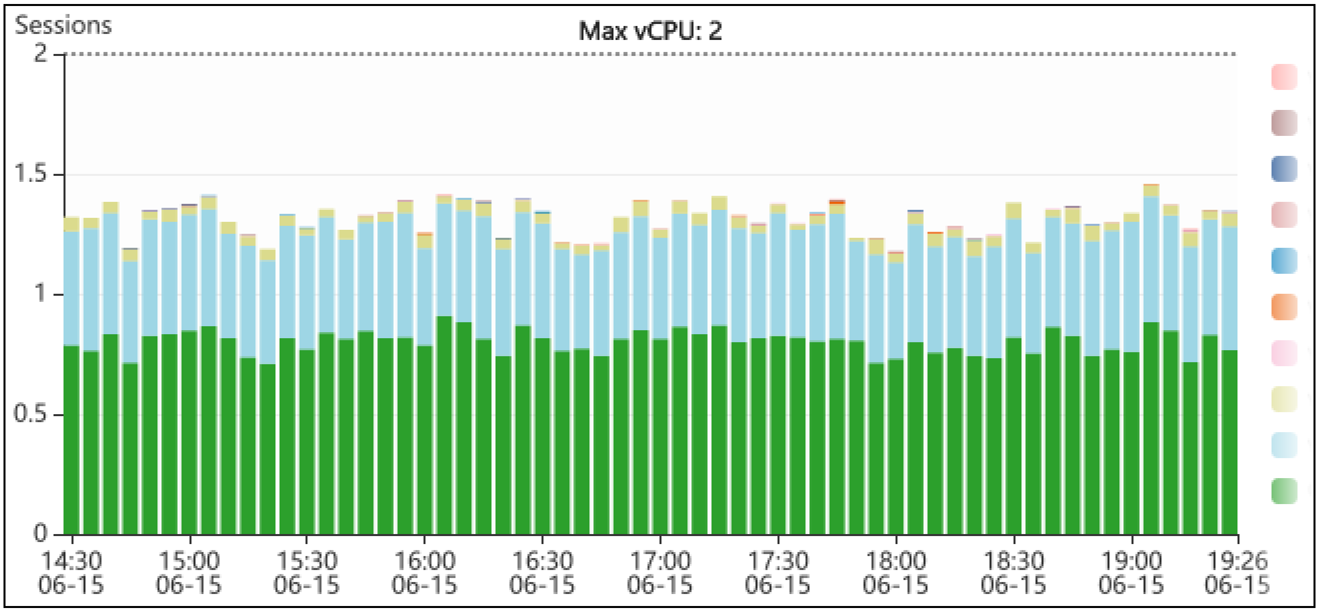
   对于已开启性能详情的数据库实例，还可以通过选择数据库实例列表中的会话项目来访问控制面板。在当前活动下，会话项目显示在过去五分钟内平均活跃会话中的数据库负载。条形图显示负载量。当条形图为空时，数据库实例处于空闲状态。随着负载的增加，条形图会以蓝色填充。当负载超过数据库实例类上的虚拟 CPU (vCPU) 数量时，条形图变为红色，表示可能出现瓶颈。



4. （可选）在右上角选择日期或时间范围，并指定不同的相对或绝对时间间隔。现在，您可以指定时间段，并生成数据库性能分析报告。该报告提供了已发现的见解和建议。有关更多信息，请参阅在性能详情中创建性能分析报告。

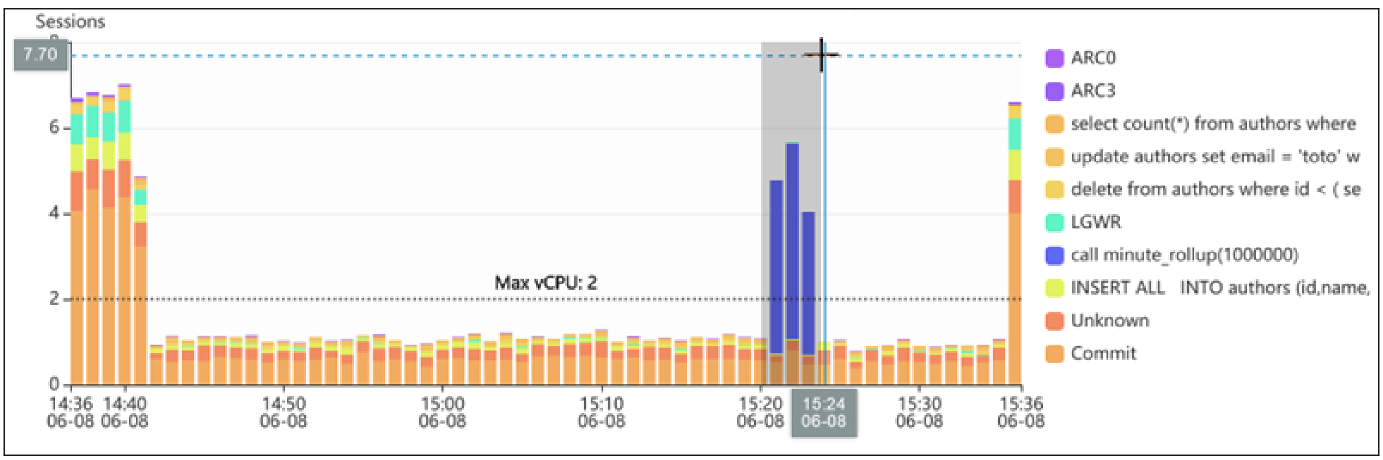


在以下屏幕截图中，数据库负载间隔为 5 小时。

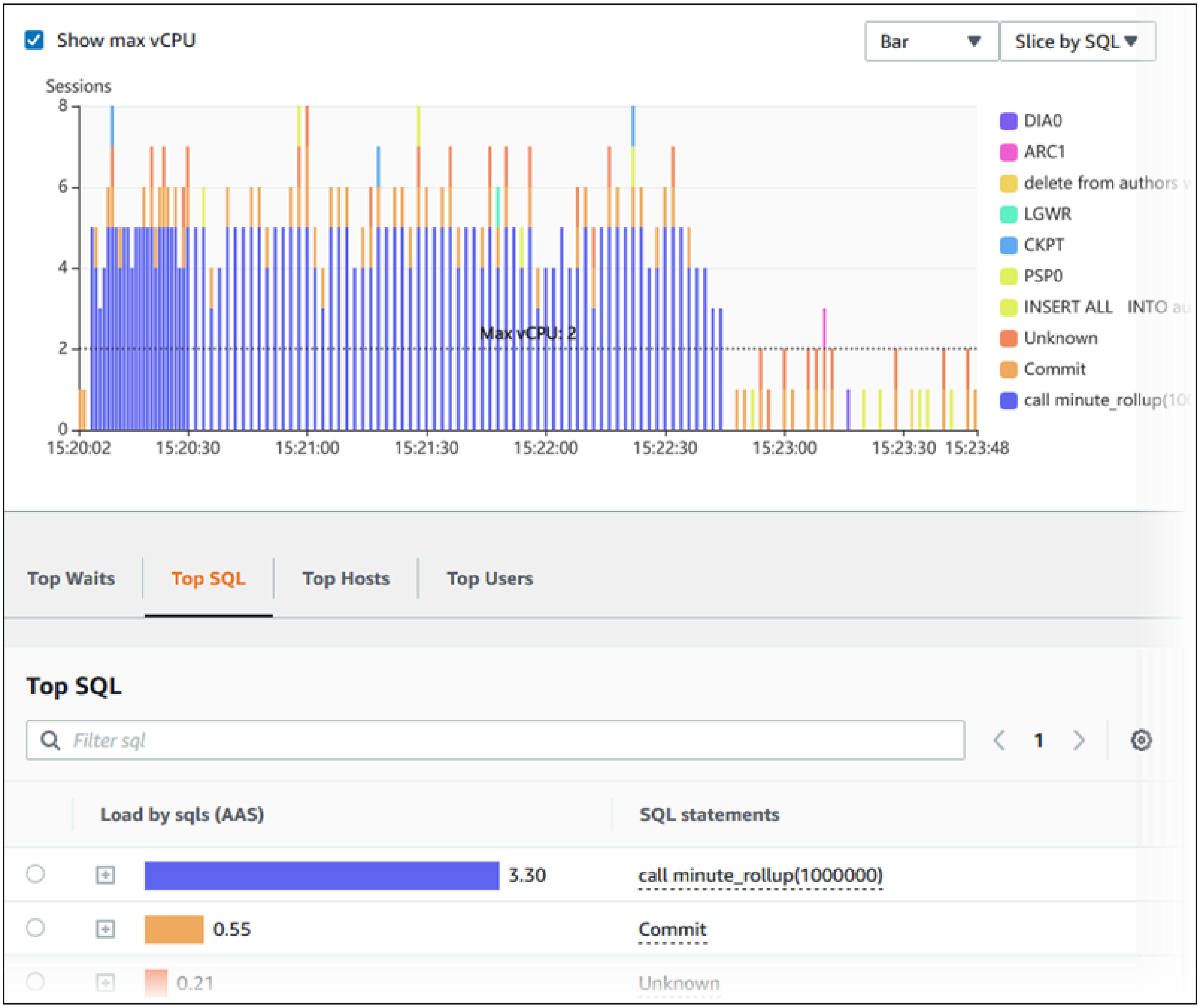


5. （可选）要放大数据库负载图表的一部分，请选择开始时间并拖动到所需时间段的结尾。

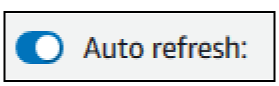
所选区域将在数据库负载图表中突出显示。



释放鼠标时，数据库负载图表上的所选 Amazon将放大，并重新计算 Top dimensions（热门维度）表。



6. （可选）要自动刷新数据，请选择自动刷新。



性能详情控制面板自动用新数据进行刷新。刷新速率取决于所显示的数据量：

- 5 分钟则每 10 秒刷新一次。
- 1 小时则每 5 分钟刷新一次。
- 5 小时则每 5 分钟刷新一次。
- 24 小时则每 30 分钟刷新一次。
- 1 周则每天刷新一次。

- 1 个月则每天刷新一次。

按等待事件分析数据库负载

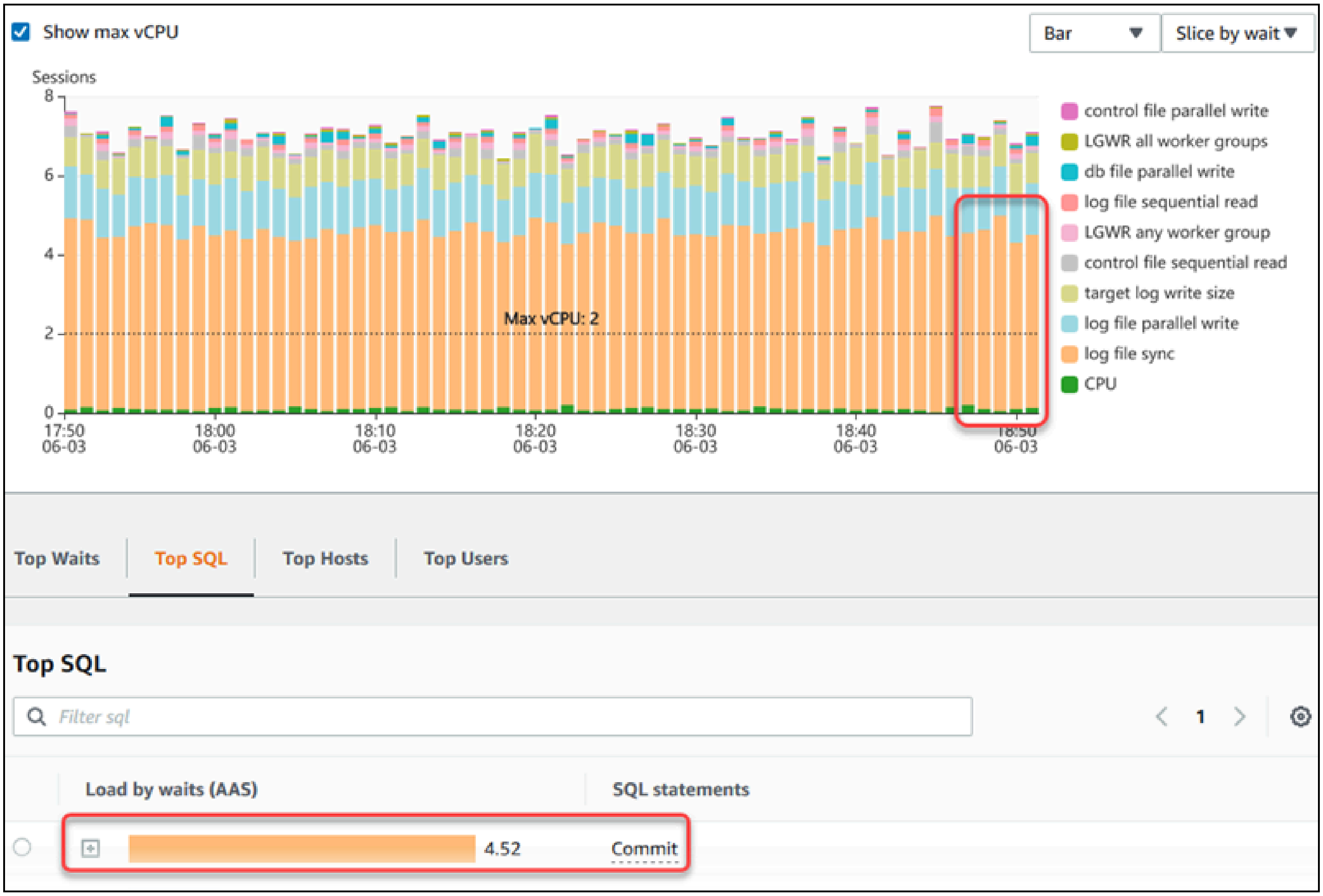
如果数据库负载图表显示了一个瓶颈，您可以找出负载的来源。为此，请查看数据库负载图表下方的主要负载项目。选择特定项目 (如 SQL 查询或用户) 以深入了解该项目并查看有关该项目的详细信息。

按等待状态和主要 SQL 查询分组的数据库负载是默认 Performance Insights 控制面板视图，此组合通常提供了最多的性能问题见解。按等待状态分组的数据库负载显示了数据库中是否存在任何资源瓶颈或并发瓶颈。在这种情况下，“Top Load Items”表的 SQL 选项卡显示了增大该负载的查询。

诊断性能问题的典型工作流程如下：

1. 查看数据库负载图表并了解是否存在数据库负载的事件越过了 Max CPU 线。
2. 如果有，请查看数据库负载图表并确定负主要责任的等待状态。
3. 通过以下方式确定导致负载的摘要查询：查看“Top Load Items”表上的 SQL 选项卡中的哪个查询对于导致这些等待状态所起的作用最大。可通过 DB Load by Wait (按等待状态排列的数据库负载) 列加以识别。
4. 在 SQL 选项卡中选择这些摘要查询之一以展开它并查看它包含的子查询。

例如，在下面的控制面板中，日志文件同步等待状态占大部分数据库负载。LGWR 所有工作线程组等待状态也很高。主要 SQL 图表显示导致日志文件同步等待状态的内容：频繁的 `COMMIT` 语句。在这种情况下，降低提交频率将会减少数据库负载。



分析一段时间内的数据库性能

通过创建一段时间的性能分析报告，可以通过按需分析来分析数据库性能。可以查看性能分析报告，发现任何性能问题，例如资源瓶颈或数据库实例中的查询更改。Performance Insights 控制面板支持您选择时间段并创建性能分析报告。您还可以向报告添加一个或多个标签。

要使用此功能，您必须使用付费套餐保留期。有关更多信息，请参阅 性能详情的定价和数据留存

该报告可供在性能分析报告 - 新增选项卡中进行选择和查看。该报告包含见解、相关指标和解决性能问题的建议。在 Performance Insights 保留期内，该报告可供查看。

如果报告分析时段的开始时间在保留期之外，则报告将被删除。您也可以在保留期结束之前删除报告。

要检测性能问题并为数据库实例生成分析报告，必须开启 Performance Insights。有关开启 Performance Insights 的更多信息，请参阅为 Amazon RDS 打开和关闭性能详情。

有关此功能的区域、数据库引擎和实例类支持信息，请参阅支持性能详情功能的 Amazon RDS 数据库引擎、区域和实例类

在以下各节中，您可以创建、查看、删除性能分析报告和向其中添加标签。

主题

- 在性能详情中创建性能分析报告
- 在性能详情中查看性能分析报告
- 在性能详情中向性能分析报告添加标签
- 在性能详情中删除性能分析报告

在性能详情中创建性能分析报告

您可以在 Performance Insights 控制面板中创建特定时段的性能分析报告。您可以选择一个时间段并将一个或多个标签添加到分析报告中。

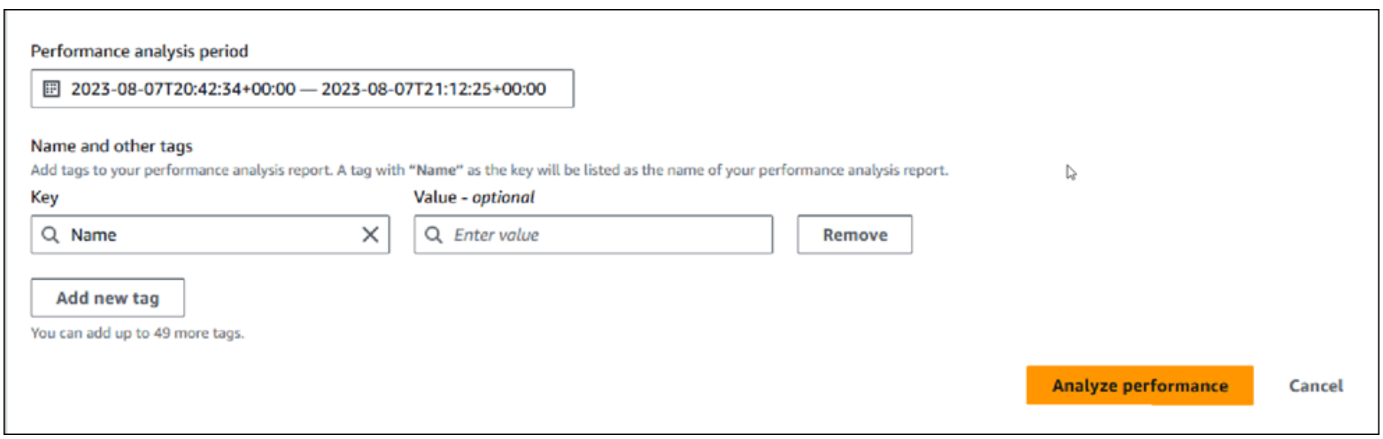
分析时段从 5 分钟到 6 天不等。在分析开始之前，必须有至少 24 小时的性能数据。

有关此功能的区域、数据库引擎和实例类支持信息，请参阅支持性能详情功能的 Amazon RDS 数据库引擎、区域和实例类

创建一段时间内的性能分析报告

1. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在左侧导航窗格中，选择性能详情。
3. 选择一个数据库实例。
4. 在性能详情控制面板上的数据库负载部分中选择分析性能。

   将显示用于设置时间段和向性能分析报告中添加一个或多个标签的字段。

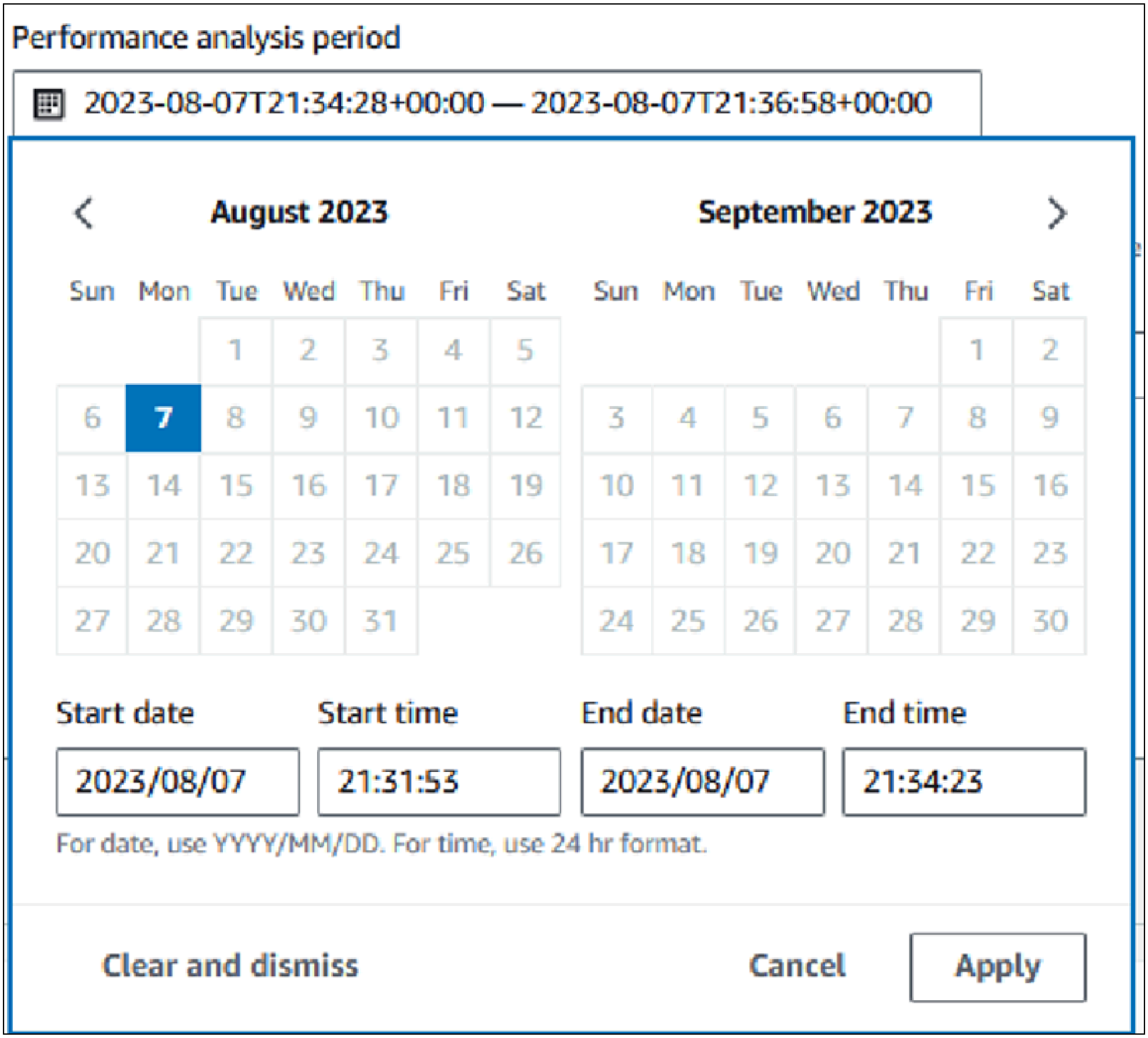


5. 选择时间段。如果在右上角的相对范围或绝对范围中设置时间段，您只能输入或选择该时间段内的分析报告日期和时间。如果您选择该时间段之外的分析时段，则会显示一条错误消息。

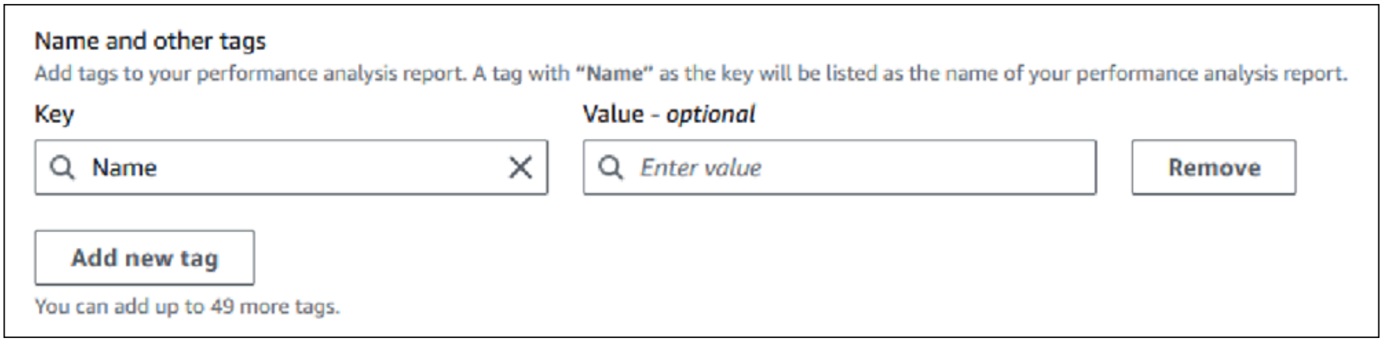
要设置时间段，您可以执行以下任一操作：

- 按下并拖动数据库负载图表上的任何滑块。

  性能分析时段框显示选定的时间段，数据库负载图表突出显示选定的时间段。
- 在性能分析时段框中，选择开始日期、开始时间、结束日期以及结束时间。



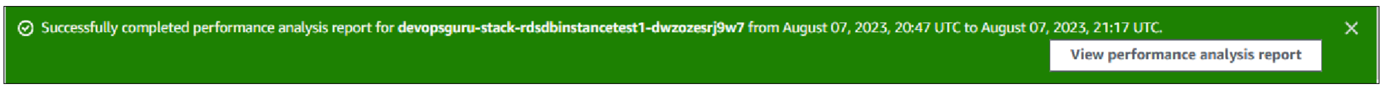
6. （可选）输入键和值 - 可选，以便为报告添加标签。



7. 选择分析性能。

   无论报告生成是成功还是失败，横幅都会显示一条消息。该消息还提供了查看报告的链接。

   以下示例显示了带有报告创建成功消息的横幅。

   该报告可供在性能分析报告 - 新增选项卡中查看。

您可以使用 Amazon CLI 创建性能分析报告。有关如何使用 Amazon CLI 创建报告的示例，请参阅创建一段时间的性能分析报告。

在性能详情中查看性能分析报告

性能分析报告 - 新增选项卡列出了为数据库实例创建的所有报告。对于每个报告，显示以下内容：

- ID：报告的唯一标识符。
- 名称：添加到报告中的标签键。
- 报告创建时间：您创建报告的时间。
- 分析开始时间：报告中分析的开始时间。
- 分析结束时间：报告中分析的结束时间。

查看性能分析报告

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在左侧导航窗格中，选择 Performance Insights。
3. 选择要查看其分析报告的数据库实例。

4. 在性能详情控制面板中，向下滚动并选择性能分析报告 - 新增选项卡。

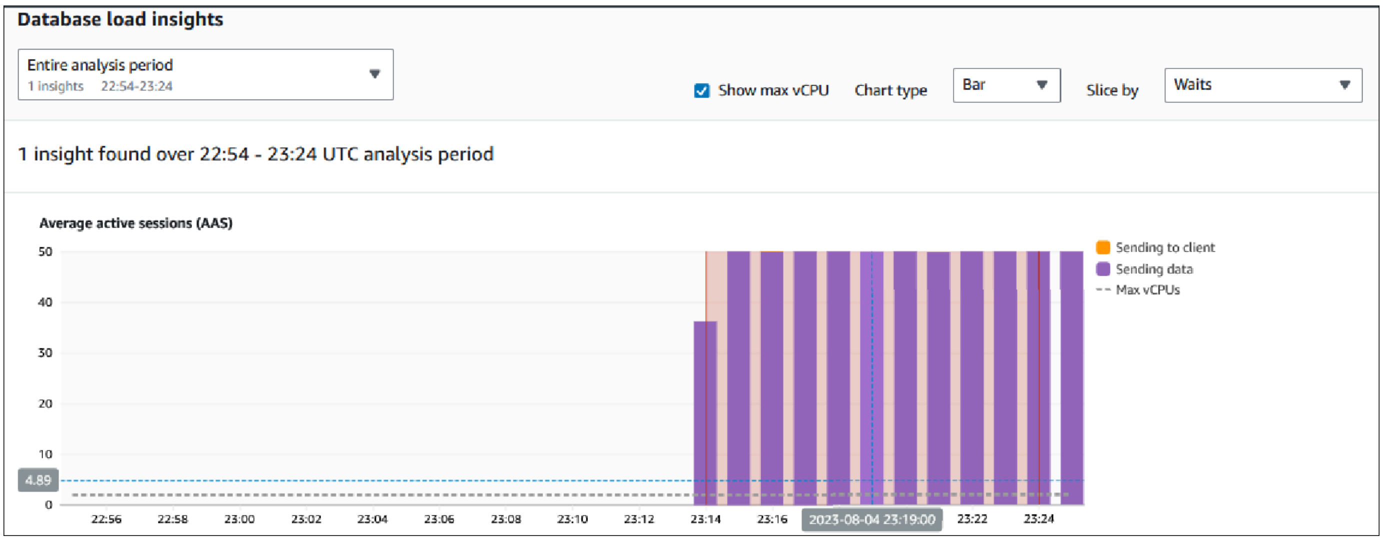
   将显示不同时间段的所有分析报告。

5. 选择您要查看的报告的 ID。

   如果确定了多个见解，则原定设置情况下，数据库负载图表会显示整个分析时段。如果报告确定了一个见解，则原定设置情况下，数据库负载图表会显示该见解。

   控制面板还在标签部分中列出了报告的标签。

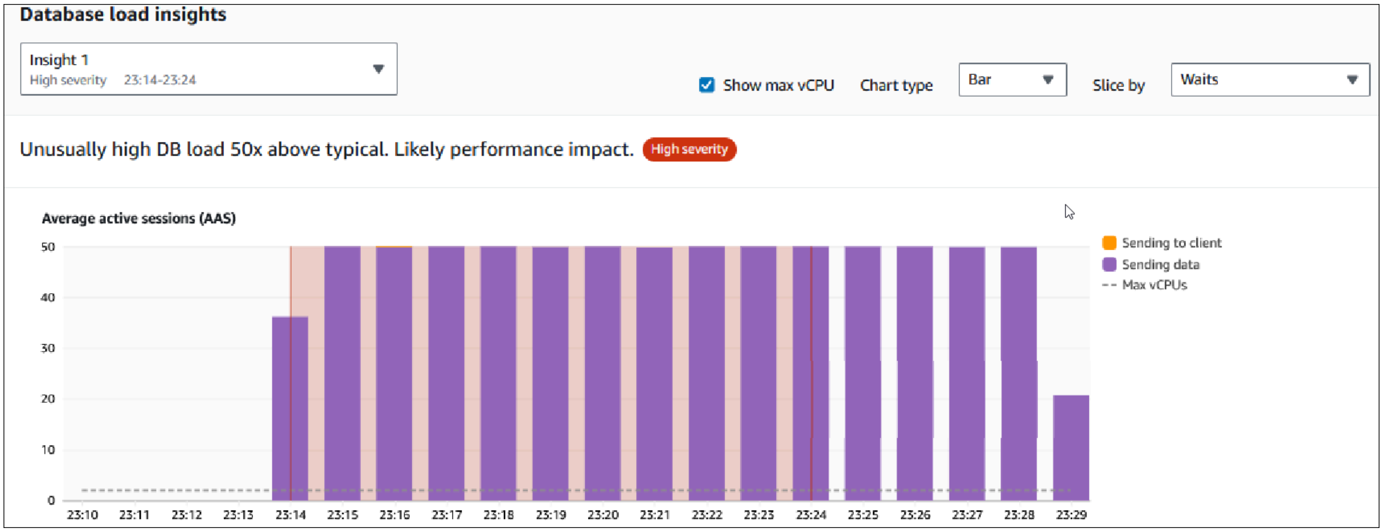
   以下示例显示了报告的整个分析时段。

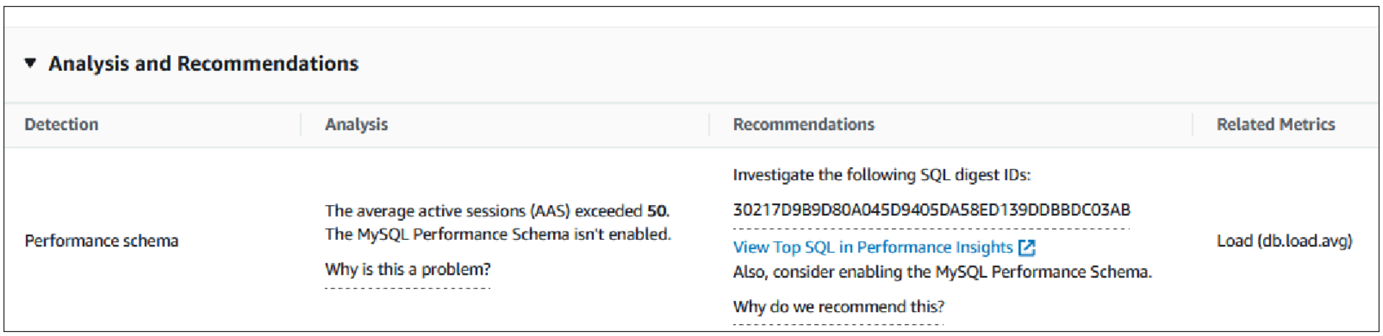


6. 如果在报告中发现多个见解，请在数据库负载见解列表中选择要查看的见解。

   控制面板显示见解消息、数据库负载图表（突出显示见解的时间段、分析和建议）以及报告标签列表。

   以下示例显示了报告中的数据库负载见解。





在性能详情中向性能分析报告添加标签

您可以在创建或查看报告时添加标签。您最多可以为一个报告添加 50 个标签。

您需要权限才能添加标签。有关 Performance Insights 的访问策略的更多信息，请参阅为 Performance Insights 配置访问策略。

要在创建报告时添加一个或多个标签，请参阅过程在性能详情中创建性能分析报告中的步骤 6。

在查看报告时添加一个或多个标签

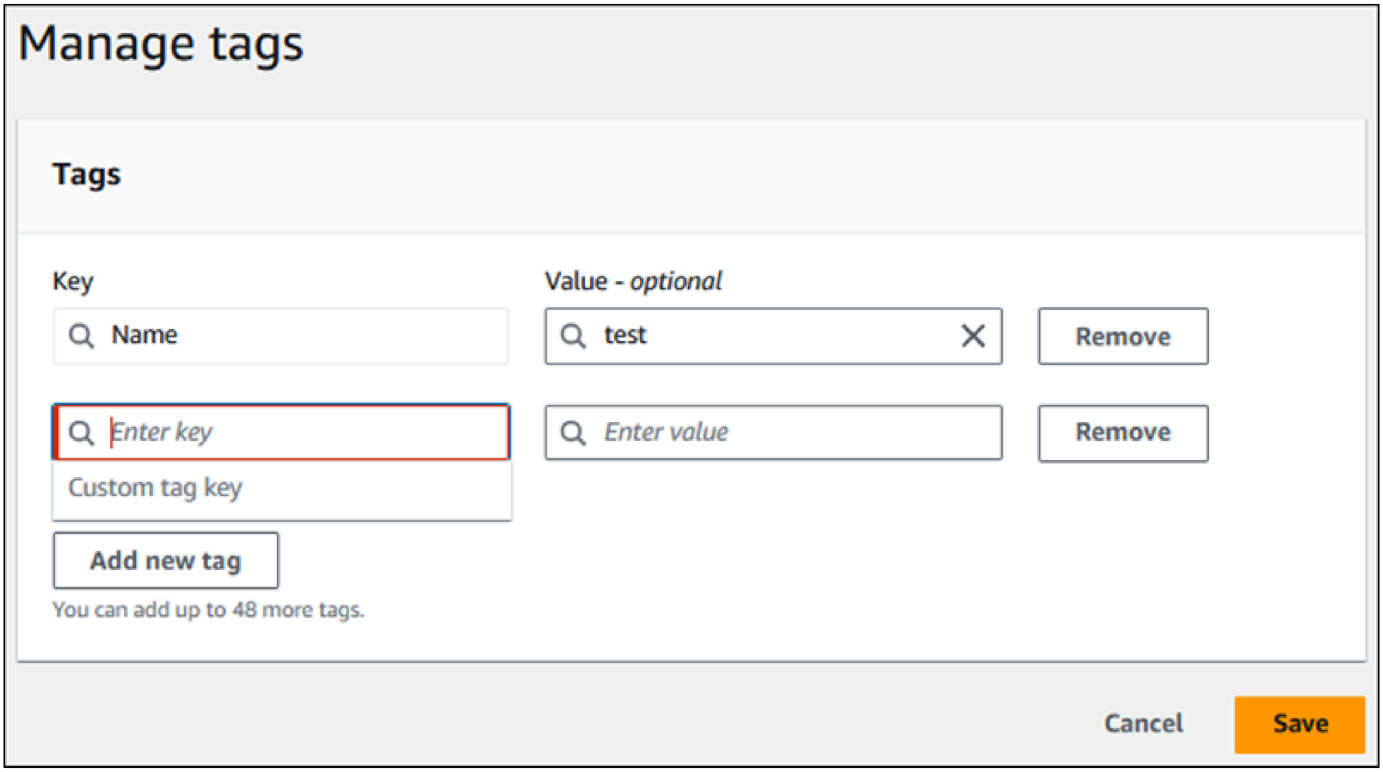
1. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在左侧导航窗格中，选择 Performance Insights。
3. 选择一个数据库实例。

   将显示该数据库实例的 Performance Insights 控制面板。
4. 向下滚动并选择性能分析报告 - 新增选项卡。
5. 选择要为其添加标签的报告。

控制面板显示报告。

6. 向下滚动到标签，然后选择管理标签。
7. 选择添加新标签。
8. 输入键和值 - 可选，然后选择添加新标签。

以下示例提供了为所选报告添加新标签的选项。



为报告创建了一个新标签。

报告的标签列表显示在控制面板上的标签部分。如果您想从报告中删除标签，请选择标签旁边的移除。

在性能详情中删除性能分析报告

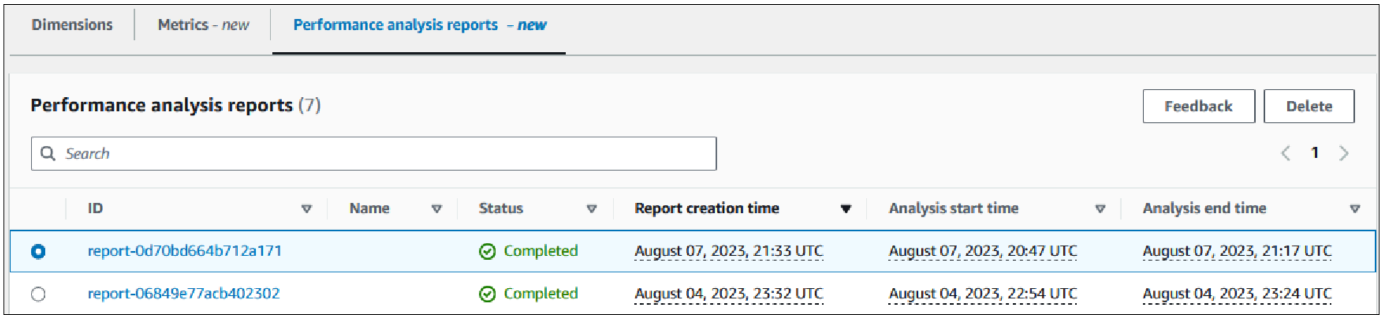
您可以从性能分析报告选项卡中显示的报告列表中删除报告，也可以在查看报告时删除报告。

删除报告

1. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在左侧导航窗格中，选择 Performance Insights。

3. 选择一个数据库实例。

   将显示该数据库实例的 Performance Insights 控制面板。
4. 向下滚动并选择性能分析报告 - 新增选项卡。
5. 选择要删除的报告，然后选择右上角的删除。



   将显示一个确认窗口。选择确认后，将删除报告。
6. （可选）选择您要删除的报告的 ID。

   在报告页面中，选择右上角的删除。

   将显示一个确认窗口。选择确认后，将删除报告。

使用性能详情中的“主要 SQL”选项卡分析查询

在 Amazon RDS Performance Insights 控制面板中，您可以在 Top dimensions（主要维度）表中的 Top SQL（主要 SQL）选项卡下找到有关运行中查询和最近查询的信息。您可以使用此信息来优化查询。

主题

- [“Top SQL”（主要 SQL）选项卡概览]
- [在 Performance Insights 控制面板中访问更多 SQL 文本]
- [在 Performance Insights 控制面板中查看 SQL 统计数据]

“Top SQL”（主要 SQL）选项卡概览

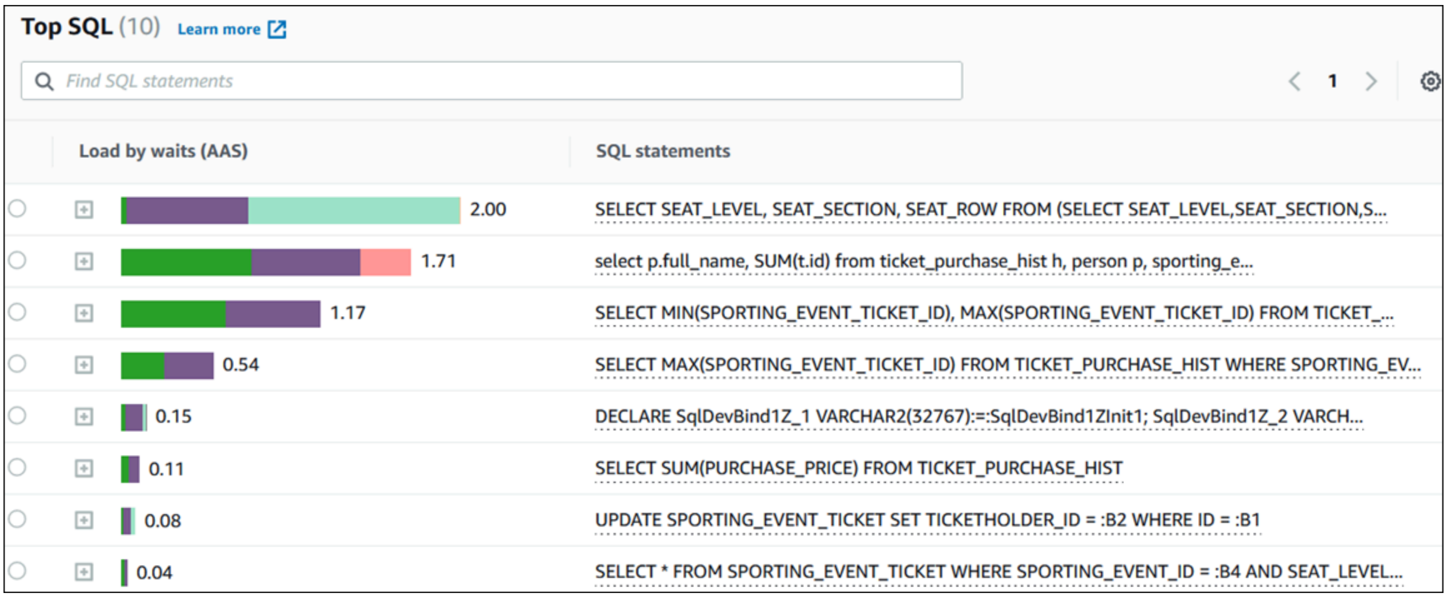
原定设置情况下，Top SQL（主要 SQL）选项卡将显示对数据库负载影响最大的 25 个 SQL 查询。为了帮助优化查询，您可以分析查询文本和 SQL 统计数据等信息。您还可以选择想要显示在 Top SQL（主要 SQL）选项卡中的统计数据。

主题

- SQL 文本
- SQL 统计数据
- 按等待状态排列的负载 (AAS)
- 查看 SQL 信息
- 选择统计数据首选项

SQL 文本

原定设置情况下，Top SQL（主要 SQL）表中的每行为每条语句显示 500 字节的文本。

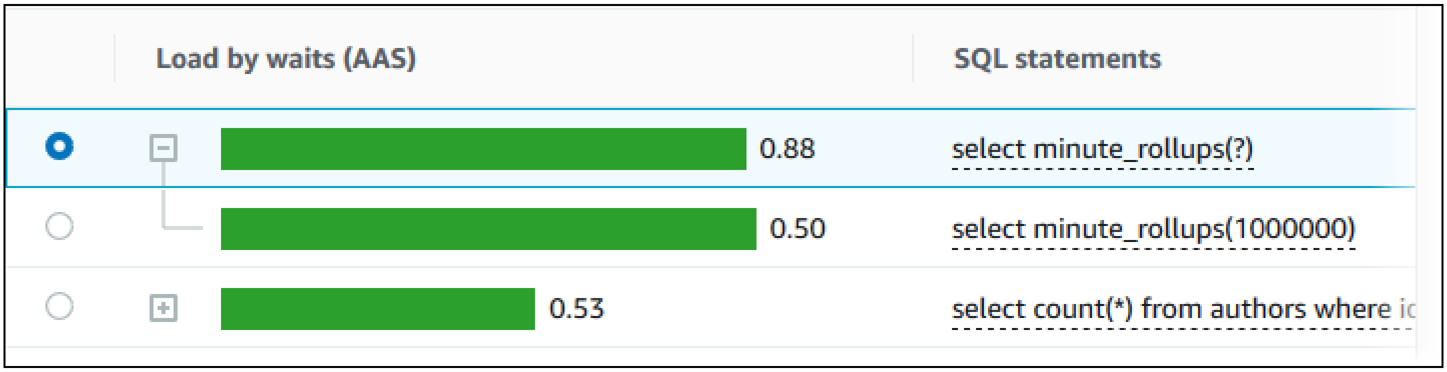


要了解如何查看超过默认 500 字节的 SQL 文本，请参阅在 Performance Insights 控制面板中访问更多 SQL 文本。

SQL 摘要是多个结构上相似但可能具有不同文本值的实际查询的组合。摘要用问号替换硬编码值。例如，摘要可能为 `SELECT * FROM emp WHERE lname= ?`。此摘要可能包含以下子查询：

```
SELECT * FROM emp WHERE lname = 'Sanchez'
SELECT * FROM emp WHERE lname = 'Olagappan'
SELECT * FROM emp WHERE lname = 'Wu'
```

要查看摘要中的文字 SQL 语句，请选择查询，然后选择加号 (+)。在以下示例中，所选查询为摘要。



> **Note**
>
> SQL 摘要将相似的 SQL 语句进行分组，但不会编辑敏感信息。

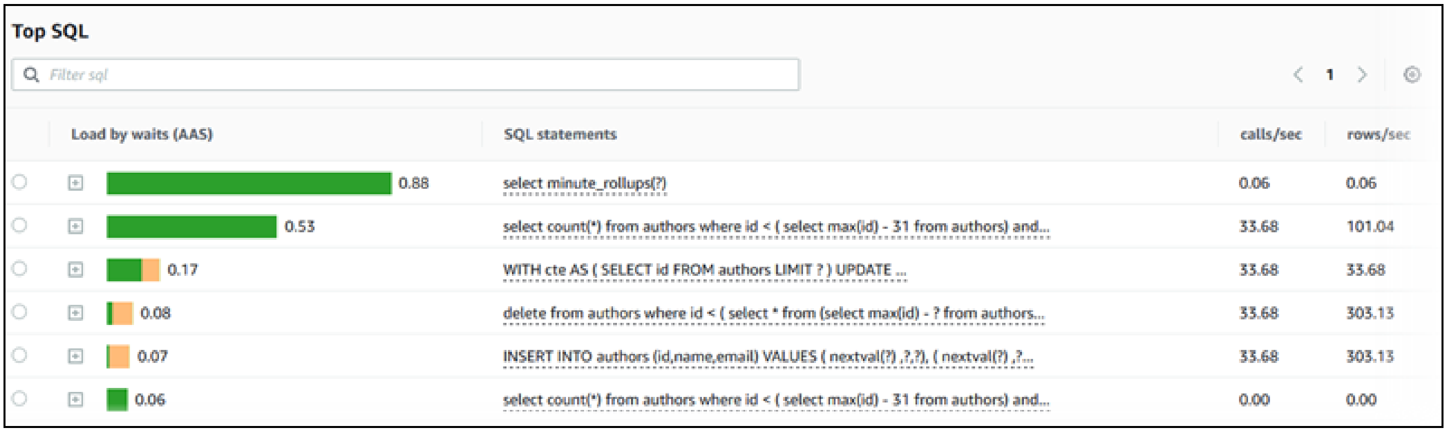
Performance Insights 可以将 Oracle SQL 文本显示为 Unknown（未知）。文本会在以下情况下具有此状态：

- 除 SYS 以外的 Oracle 数据库用户处于活动状态，但当前未执行 SQL。例如，当并行查询完成时，查询协调器将等待帮助程序进程发送会话统计数据。在等待期间，查询文本将显示 Unknown（未知）。
- 对于标准版 2 上的 RDS for Oracle 实例，Oracle 资源管理器会限制并行线程的数量。执行此工作的后台进程会导致查询文本显示为 Unknown（未知）。

SQL 统计数据

SQL 统计数据是关于 SQL 查询的性能相关指标。例如，性能详情可能会显示每秒执行数或每秒处理的行数。性能详情仅收集最常见查询的统计数据。通常，它们与 Performance Insights 控制面板中显示的按负载列出的主要查询匹配。

Top SQL（主要 SQL）表中的每一行显示了 SQL 语句或摘要的相关统计数据，如以下示例所示。



Performance Insights 可以报告 SQL 统计数据的 `0.00` 和 `-`（未知）。这种情况在以下条件下发生：

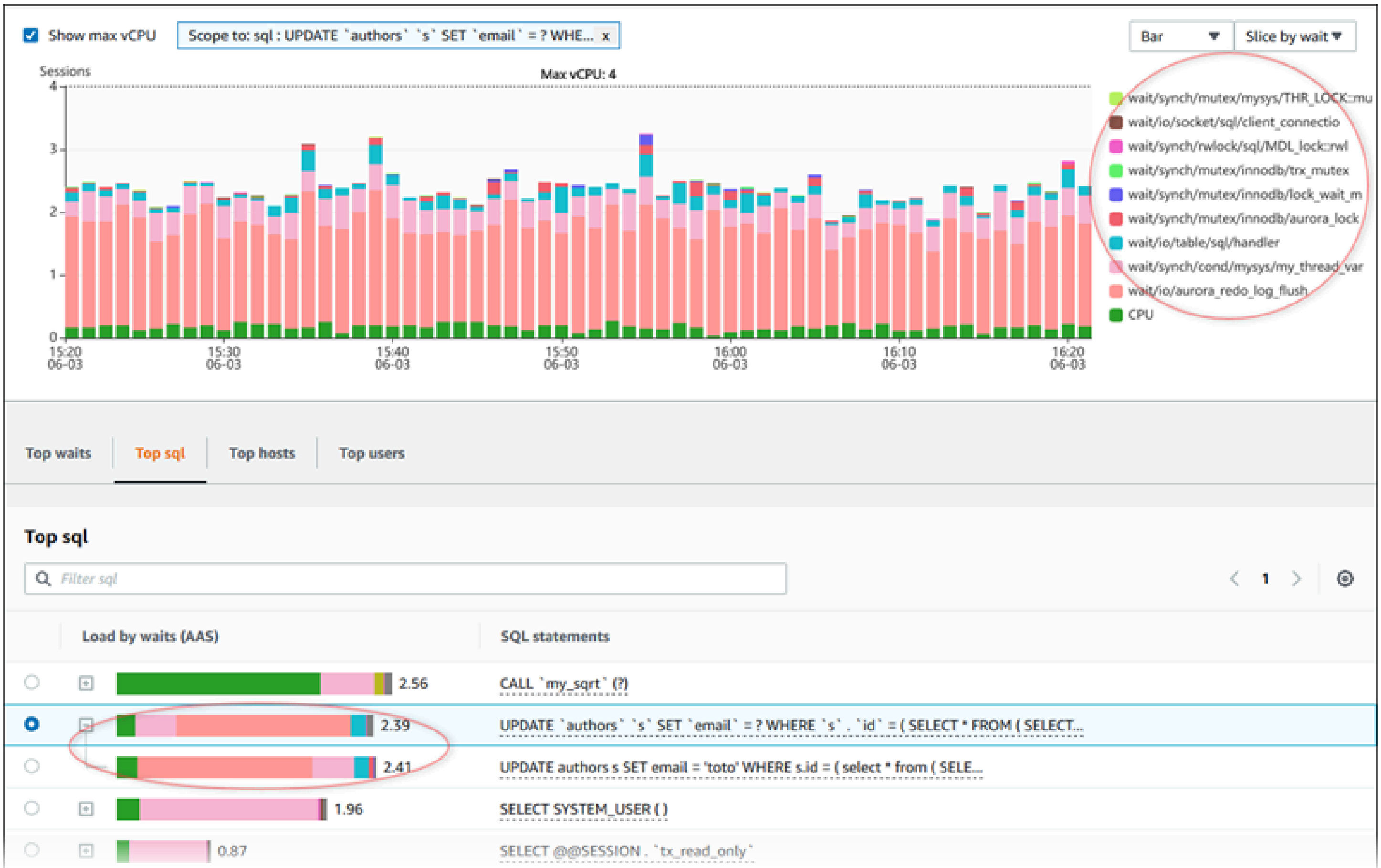
- 只存在一个样本。例如，Performance Insights 根据 `pg_stat_statements` 视图的多个样本，计算 RDS PostgreSQL 的更改速率。当工作负载运行时间较短时，Performance Insights 可能只收集一个样本，这意味着它无法计算更改速率。未知值用短划线 (`-`) 表示。
- 两个样本具有相同的值。由于未发生更改，Performance Insights 无法计算更改速率，因此它将速率报告为 `0.00`。
- RDS PostgreSQL 语句缺乏有效的标识符。PostgreSQL 仅在解析和分析之后才会为语句创建标识符。因此，语句可以存在于 PostgreSQL 内部内存结构中，而无需标识符。由于 Performance Insights 每秒对内部内存结构进行一次采样，因此低延迟查询可能只出现在单个样本中。如果查询标识符对该样本不可用，则 Performance Insights 无法将此语句与其统计数据进行关联。未知值用短划线 (`-`) 表示。

有关 Amazon RDS 引擎的 SQL 统计数据的说明，请参阅 Performance Insights 的 SQL 统计数据。

按等待状态排列的负载 (AAS)

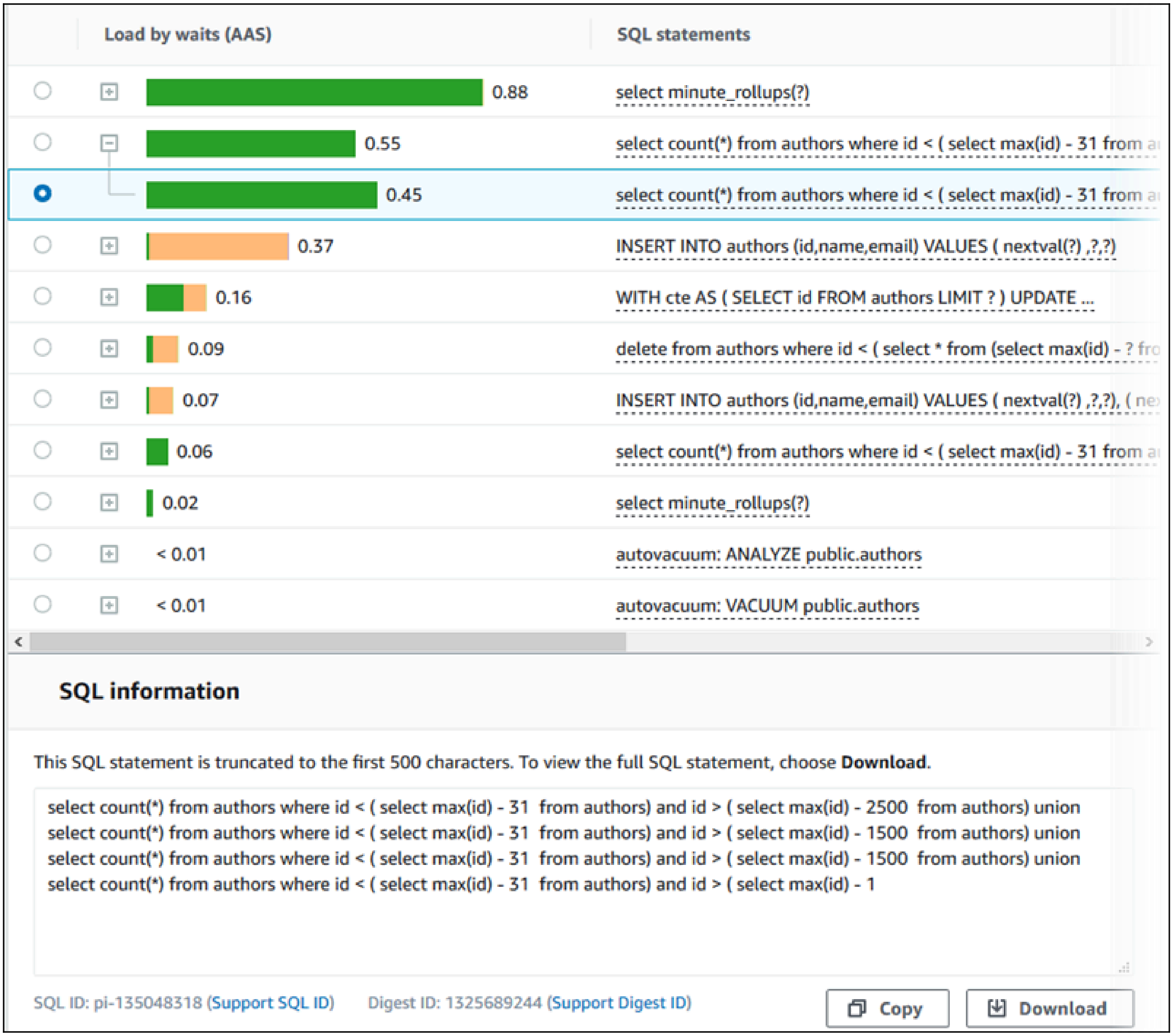
在 Top SQL（主要 SQL）中，按等待状态排列的负载 (AAS) 列说明了与每个主要负载项目关联的数据库负载的百分比。此列按当前在数据库负载图表中选择的分组方式反映该项目的负载。有关平均活动会话数（AAS）的更多信息，请参阅平均活动会话数。

例如，您可以按等待状态对数据库负载图表进行分组。您可以检查主要负载项目表中的 SQL 查询。在这种情况下，系统将对 DB Load by Waits (按等待状态排列的数据库负载) 栏进行大小调整、分段和颜色编码，以显示该查询在导致给定等待状态方面所起的作用大小，它还会显示哪些等待状态正在影响选定的查询。



查看 SQL 信息

在 Top SQL（主要 SQL）表中，您可以打开一条语句以查看其信息。信息将显示在底部窗格中。

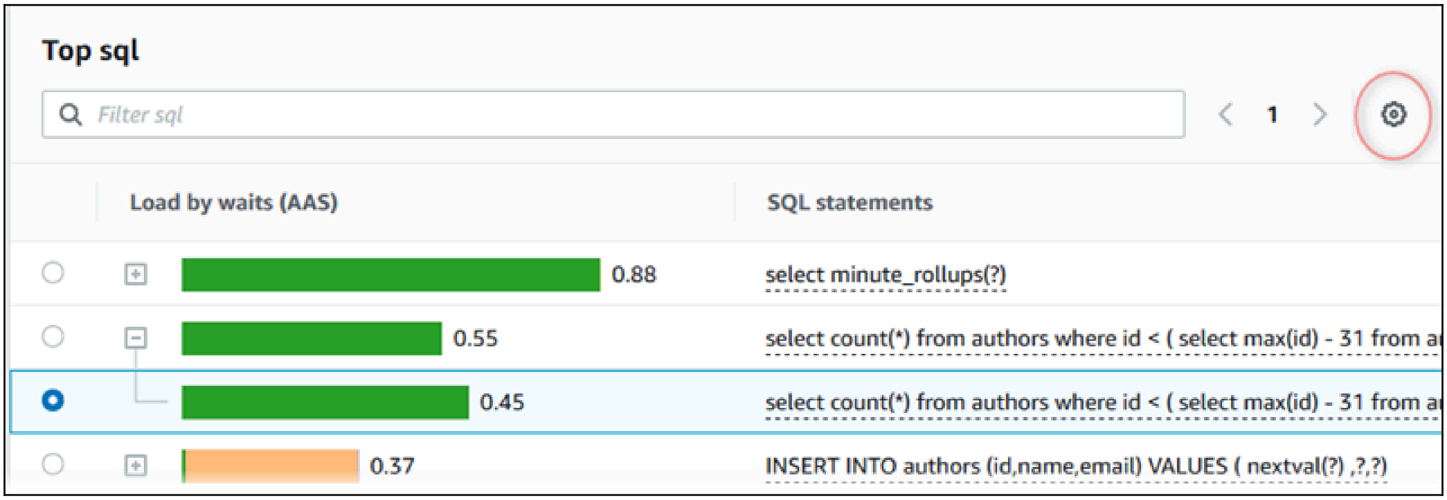


与 SQL 语句关联的以下类型的标识符 (ID)：

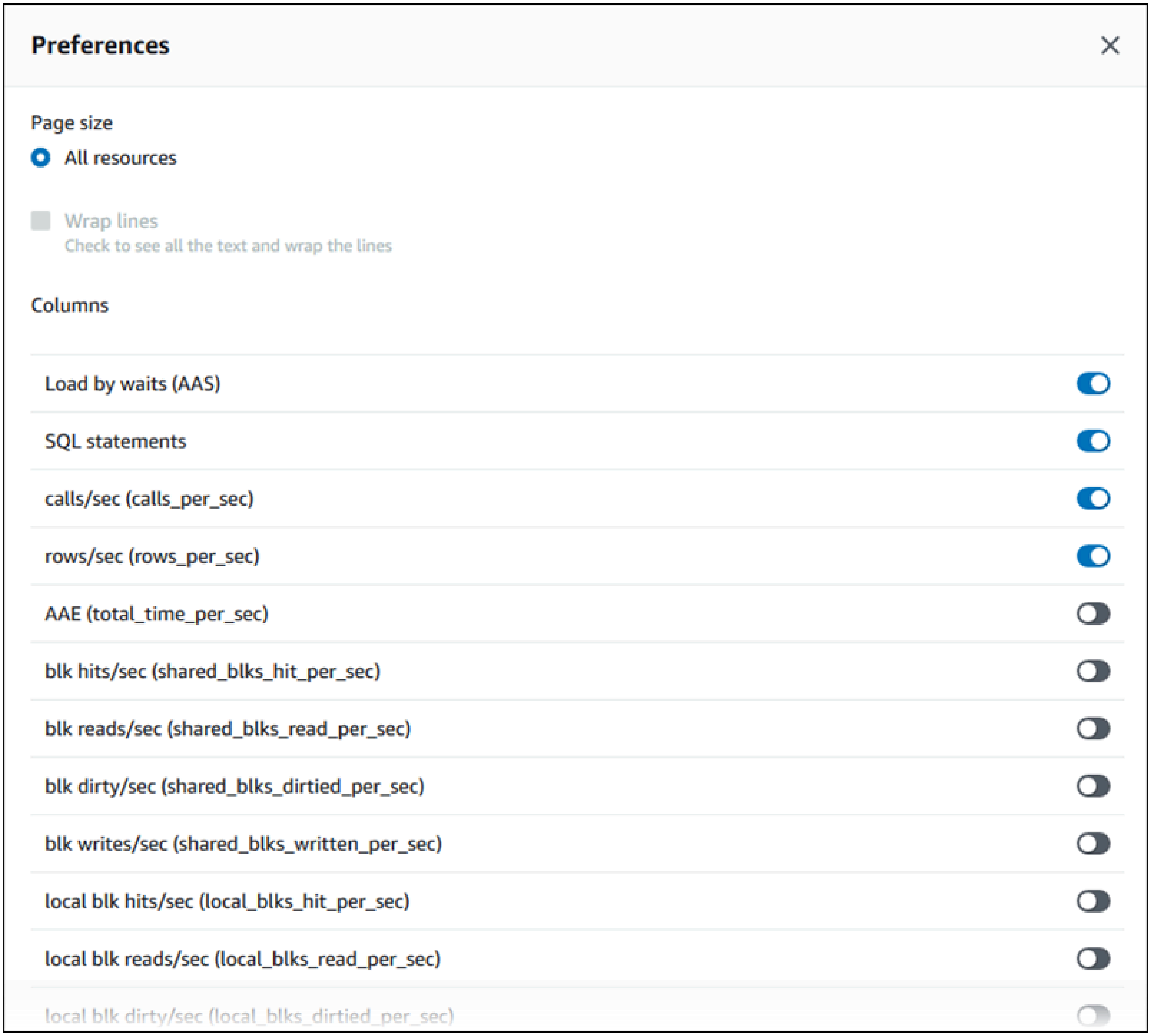
- 支持 SQL ID – SQL ID 的哈希值。此值仅用于当您处理 Amazon Support 时引用 SQL ID。AmazonSupport 无法访问您实际的 SQL ID 和 SQL 文本。
- 支持摘要 ID – 摘要 ID 的哈希值。此值仅用于当您处理 Amazon Support 时引用摘要 ID。AmazonSupport 无法访问您实际的摘要 ID 和 SQL 文本。

选择统计数据首选项

您可以通过选择 Preferences（首选项）图标来控制 Top SQL（主要 SQL）选项卡中显示的统计数据。



当您选择 Preferences (首选项) 图标时，Preferences (首选项) 窗口将打开。以下屏幕截图是 Preferences（首选项）窗口的示例。



要启用您希望在 Top SQL（主要 SQL）选项卡中显示的统计数据，请使用鼠标滚动到窗口底部，然后选择 Continue（继续）。

有关 Amazon RDS 引擎每秒或每次调用统计数据的更多信息，请参阅Performance Insights 的 SQL 统计数据中引擎特定的 SQL 统计数据部分

在 Performance Insights 控制面板中访问更多 SQL 文本

预设情况下，Top SQL（主要 SQL）表中的每行为每个 SQL 语句显示 500 字节的 SQL 文本。

当 SQL 语句超过 500 字节时，您可以在 Top SQL（主要 SQL）表下的 SQL text（SQL 文本）部分中查看更多文本。在这种情况下，SQL text（SQL 文本）中显示的文本的最大长度为 4KB。此限制由控制台引入，并受数据库引擎设置的限制的约束。要保存 SQL text（SQL 文本）中显示的文本，请选择 Download（下载）。

主题

- [Amazon RDS 引擎 的文本大小限制]
- [为 Amazon RDS for PostgreSQL 数据库实例设置 SQL 文本限制]
- [在 Performance Insights 控制面板中查看和下载 SQL 文本]

Amazon RDS 引擎 的文本大小限制

下载 SQL 文本时，数据库引擎将确定文本的最大长度。您可以下载最多为以下每个引擎限制的 SQL 文本。

数据库引擎	已下载文本的最大长度
Amazon RDS for MySQL 和 MariaDB	当启用性能架构时，长度固定为 4096 字节。如果未启用性能架构，则长度固定为 65535 字节。
Amazon RDS for Microsoft SQL Server	4,096 个字符
Amazon RDS for Oracle	1,000 字节

性能详情控制台的 SQL text（SQL 文本）部分最多显示引擎返回的最大长度的文本。例如，如果 MySQL 最多返回 1 KB 到 Performance Insights，则只能收集并显示 1 KB，即使原始查询较大。因此，当您在 SQL text（SQL 文本）中查看查询或下载查询时，性能详情将返回相同的字节数。

如果您使用 Amazon CLI 或 API，则性能详情没有控制台强制实施的 4KB 限制。`DescribeDimensionKeys` 和 `GetResourceMetrics` 返回最多 500 字节。

> **Note**
>
> `GetDimensionKeyDetails` 将返回完整的查询，但大小受引擎限制约束。

为 Amazon RDS for PostgreSQL 数据库实例设置 SQL 文本限制

Amazon RDS for PostgreSQL 以不同的方式处理文本。您可以使用数据库实例参数 `track_activity_query_size` 设置文本大小限制。该参数具有以下特征：

默认文本大小

在 Amazon RDS for PostgreSQL 版本 9.6 中，`track_activity_query_size` 参数的默认设置为 1024 字节。在 Amazon RDS for PostgreSQL 版本 10 或更高版本中，默认值为 4096 字节。

最大文本大小

对于 Amazon RDS for PostgreSQL 版本 12 及更低版本，`track_activity_query_size` 的限制为 102400 字节。版本 13 及更高版本的最大值为 1 MB。

如果引擎返回 1 MB 至 Performance Insights，则控制台仅显示前 4 KB。如果您下载查询，您将得到完整的 1 MB。在这种情况下，查看和下载返回不同的字节数。有关 `track_activity_query_size` 数据库实例参数的更多信息，请参阅 PostgreSQL 文档中的[运行时统计数据](#)。

要增加 SQL 文本大小，请增加 `track_activity_query_size` 限制。要修改参数，请更改与 Amazon RDS for PostgreSQL 数据库实例关联的参数组中的参数设置。

在实例使用默认参数组时更改设置

1. 为相应数据库引擎和数据库引擎版本创建新的数据库实例参数组。
2. 在新参数组中设置参数。
3. 将新参数组与数据库实例相关联。

有关设置数据库实例参数的信息，请参阅 [在 Amazon RDS 中修改数据库参数组中的参数](#)。

在 Performance Insights 控制面板中查看和下载 SQL 文本

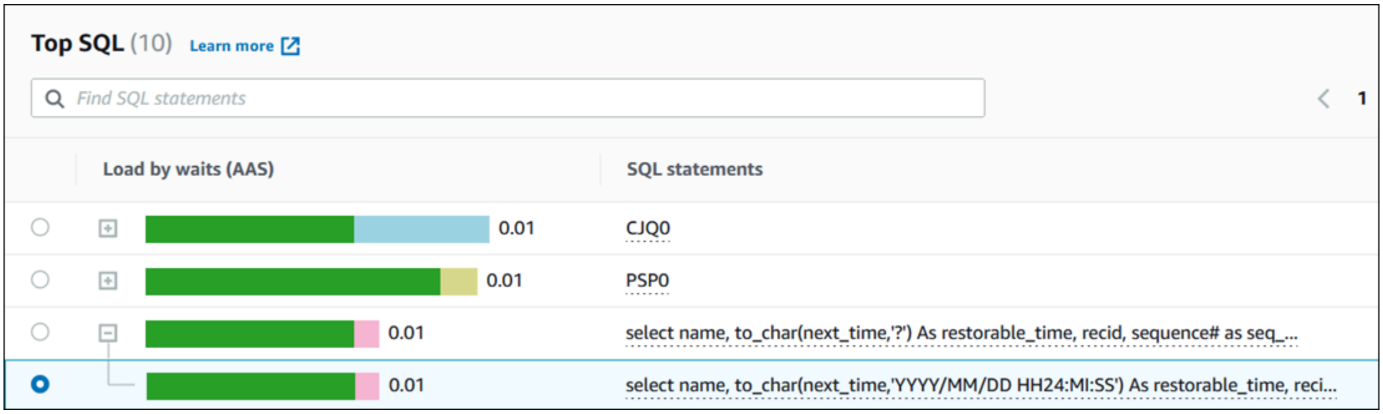
在 Performance Insights 控制面板中，您可以查看或下载 SQL 文本。

在 Performance Insights 控制面板中查看更多 SQL 文本

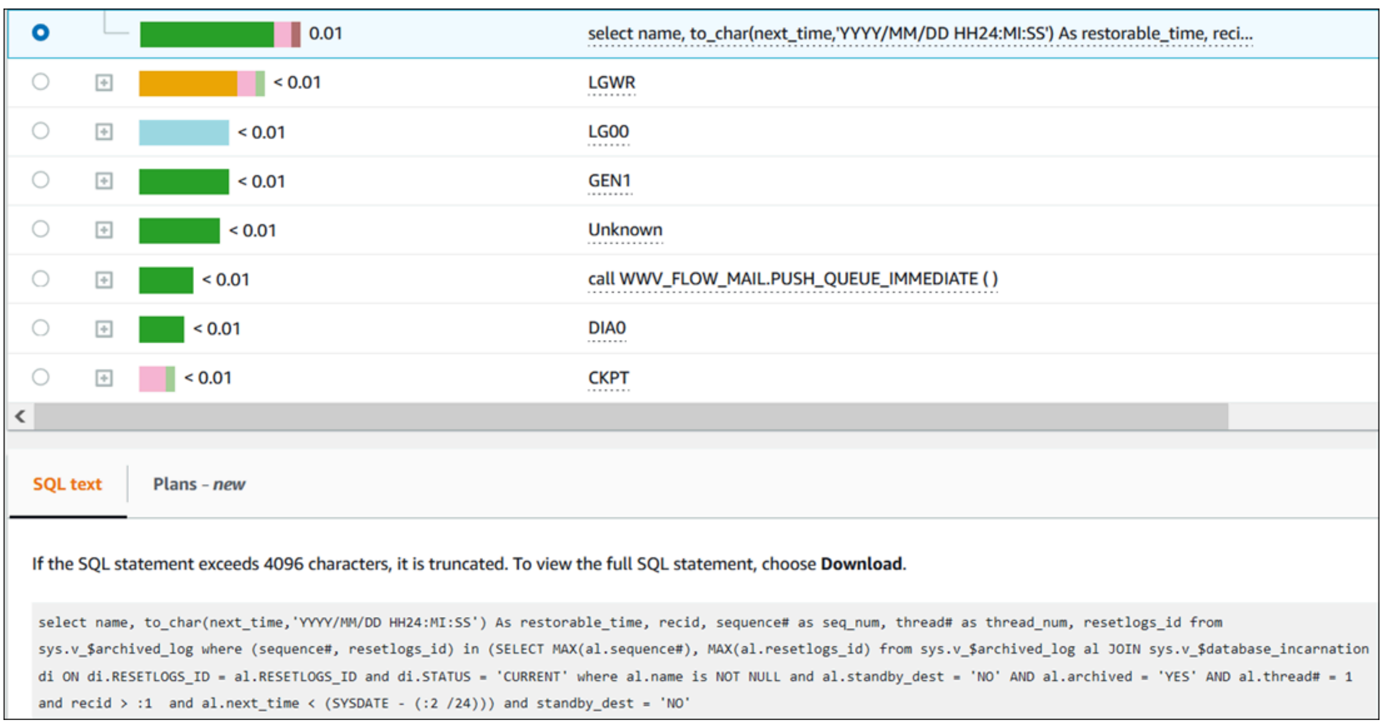
1. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Performance Insights。

3. 选择一个数据库实例。
4. 在性能详情控制面板中向下滚动到主要 SQL 选项卡。
5. 选择加号来展开 SQL 摘要，然后选择该摘要的子查询之一。

   具有大于 500 字节的文本的 SQL 语句如下图所示。



6. 向下滚动至 SQL text（SQL 文本）选项卡。



   Performance Insights 控制面板可以为每个 SQL 语句最多显示 4096 字节。

7. （可选）选择复制来复制所显示的 SQL 语句，或选择下载来下载 SQL 语句以查看不超过数据库引擎限制的 SQL 文本。

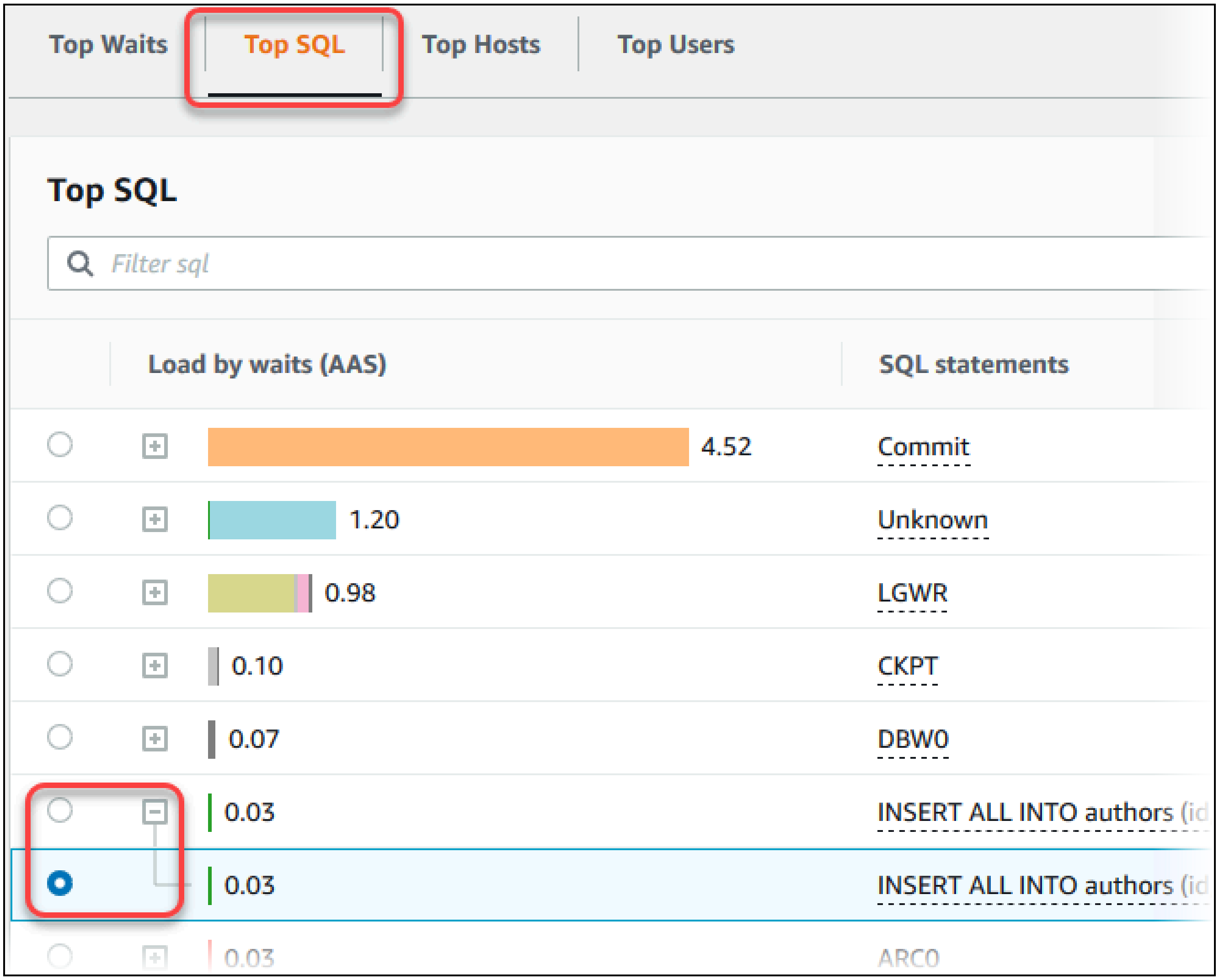
> **Note**
>
> 要复制或下载 SQL 语句，请禁用弹出窗口阻止程序。

在 Performance Insights 控制面板中查看 SQL 统计数据

在 Performance Insights 控制面板中，SQL 统计数据可在 Database load（数据库负载）图表的 Top SQL（主要 SQL）选项卡中找到。

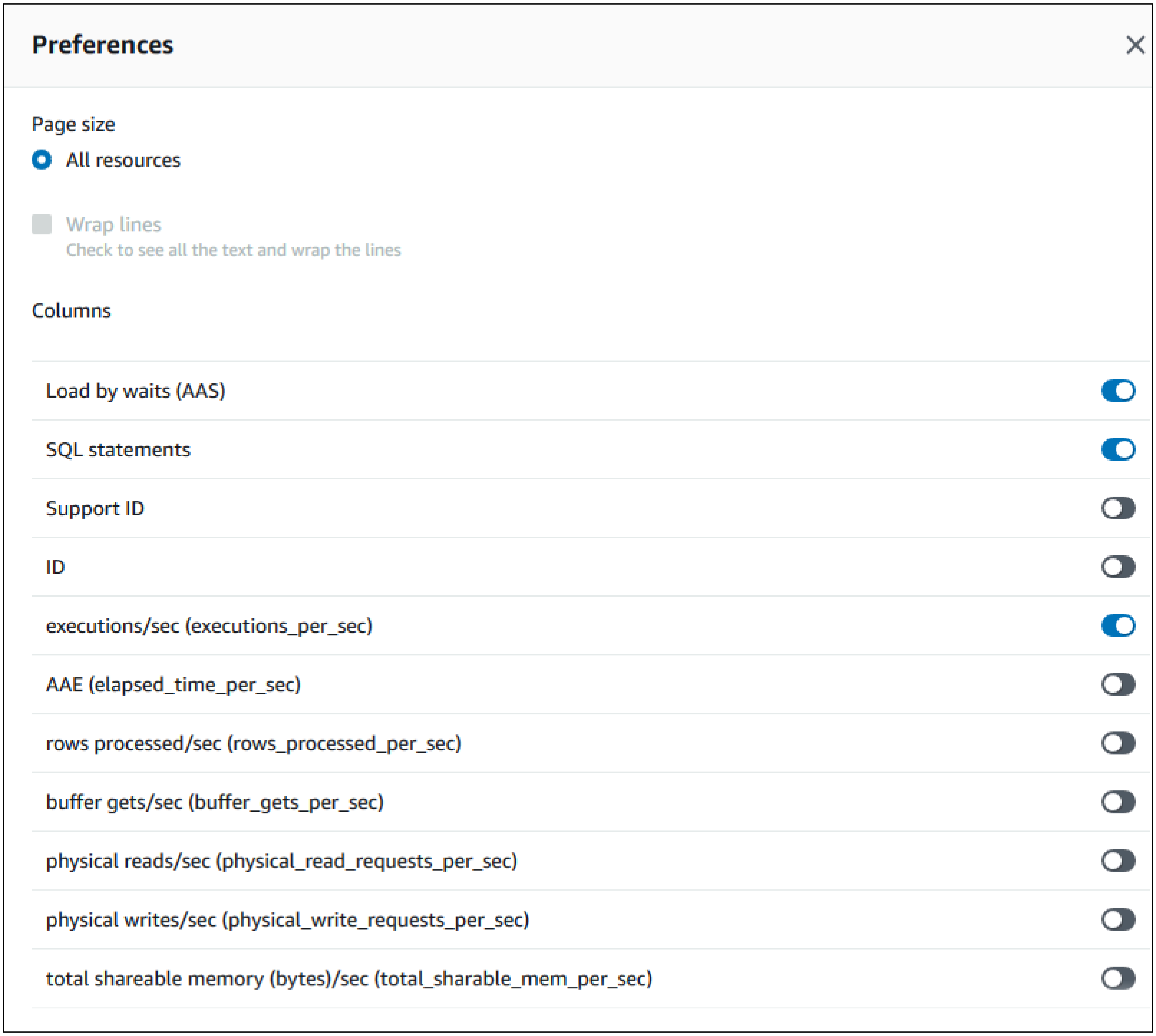
查看 SQL 统计数据

1. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在左侧导航窗格中，选择 Performance Insights。
3. 在页面顶部，选择要查看 SQL 统计数据的数据库。
4. 滚动到页面底部并选择 Top SQL（主要 SQL）选项卡。
5. 选择单独的语句或摘要查询。

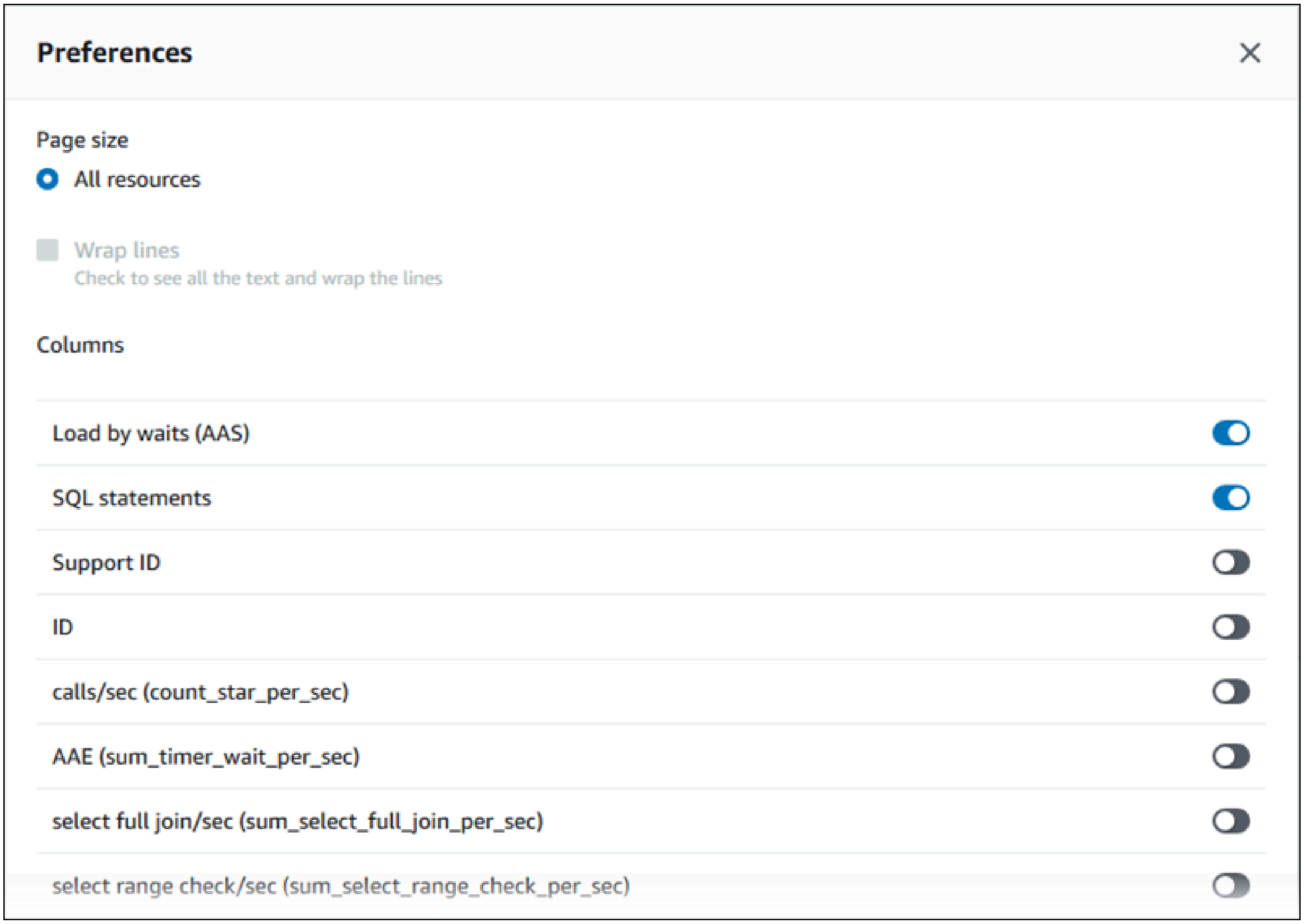


6. 通过选择图表右上角的齿轮图标来选择要显示的统计数据。有关 Amazon RDS 引擎的 SQL 统计数据的说明，请参阅 Performance Insights 的 SQL 统计数据。

   以下示例显示 Oracle 数据库实例的统计数据首选项。



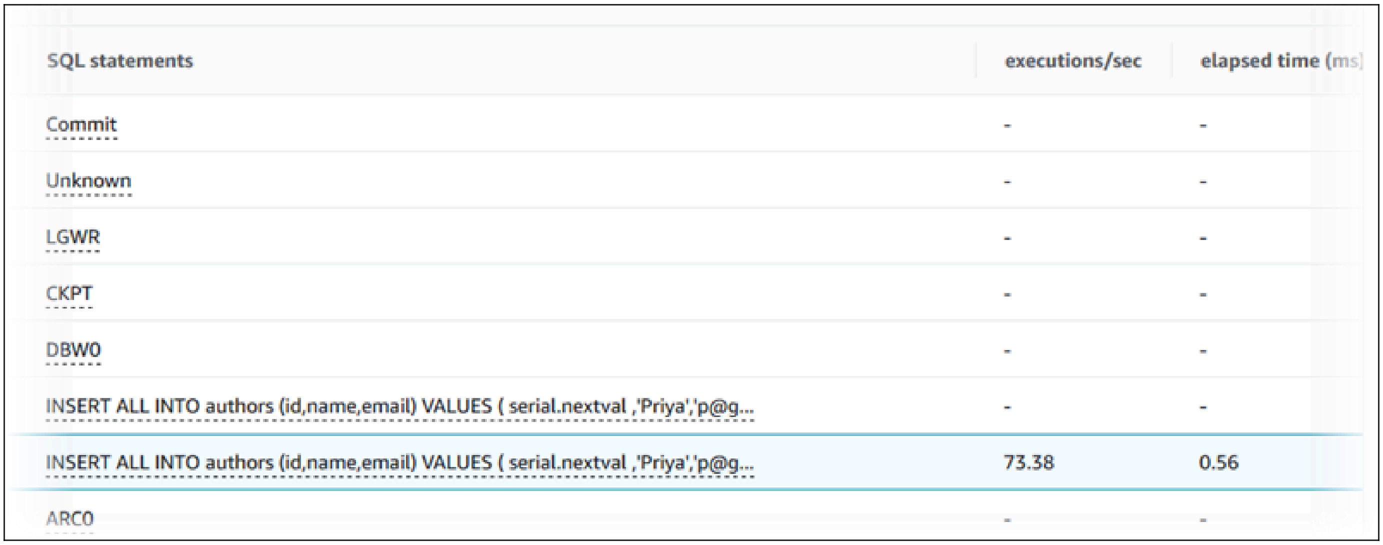
以下示例显示 MariaDB 和 MySQL 数据库实例的首选项。



7. 选择“Save（保存）”以保存首选项。

   Top SQL（主要 SQL）表将刷新。

   以下示例显示 Oracle SQL 查询的统计数据。



SQL statements	executions/sec	elapsed time (ms)
Commit	-	-
Unknown	-	-
LGWR	-	-
CKPT	-	-
DBW0	-	-
INSERT ALL INTO authors (id,name,email) VALUES (serial.nextval ,'Priya','p@g...	-	-
INSERT ALL INTO authors (id,name,email) VALUES (serial.nextval ,'Priya','p@g...	73.38	0.56
ARC0	-	-

分析 Oracle 最大 PDB 负载

分析 Oracle 容器数据库（CDB）负载时，您可能想了解哪些可插拔数据库（PDB）对数据库负载的影响最大。您可能还希望比较运行类似查询的各个 PDB 的性能，以微调性能。有关 Oracle CDB 的更多信息，请参阅 RDS for Oracle 数据库架构。

在 Amazon RDS Performance Insights 控制面板中，您可以在维度选项卡的最大 PDB 选项卡下查找有关可插拔数据库（PDB）的信息。

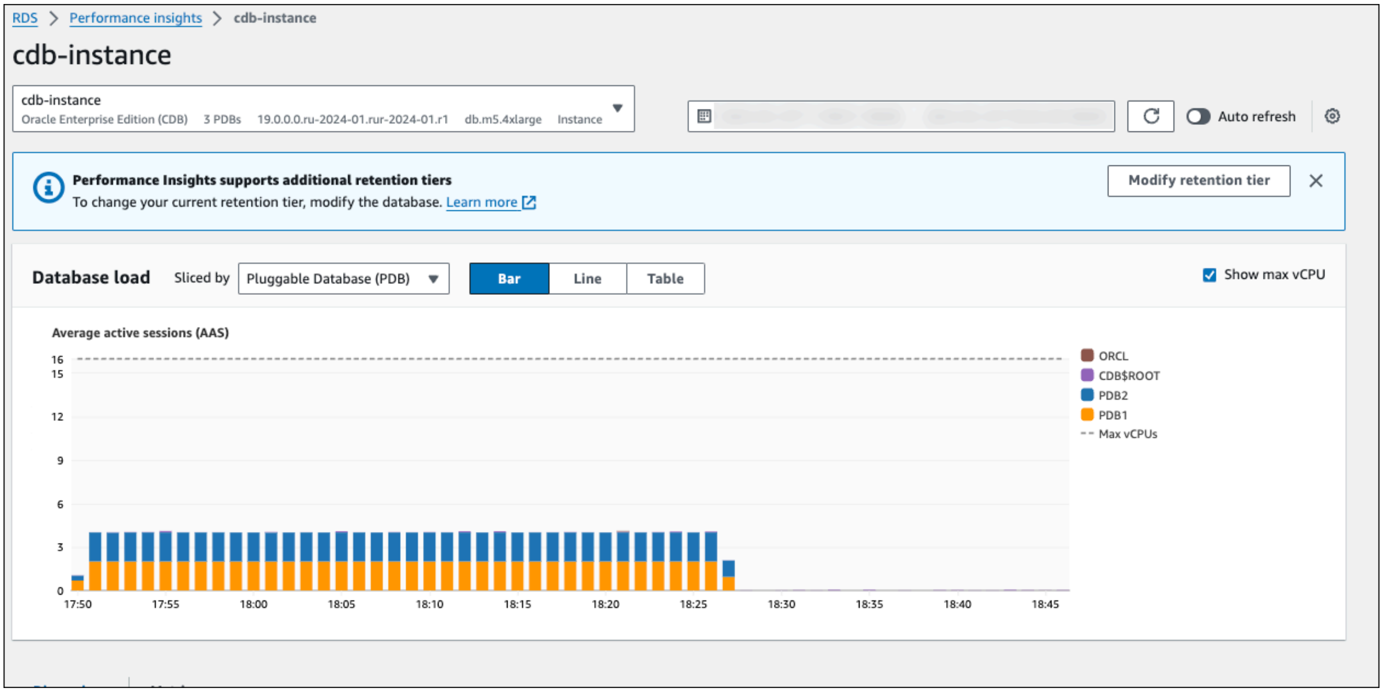
有关此功能的区域、数据库引擎和实例类支持信息，请参阅 支持性能详情功能的 Amazon RDS 数据库引擎、区域和实例类。

分析 Oracle CDB 中的最大 PDB 负载

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在左侧导航窗格中，选择 Performance Insights。
3. 选择 Oracle CDB 实例。

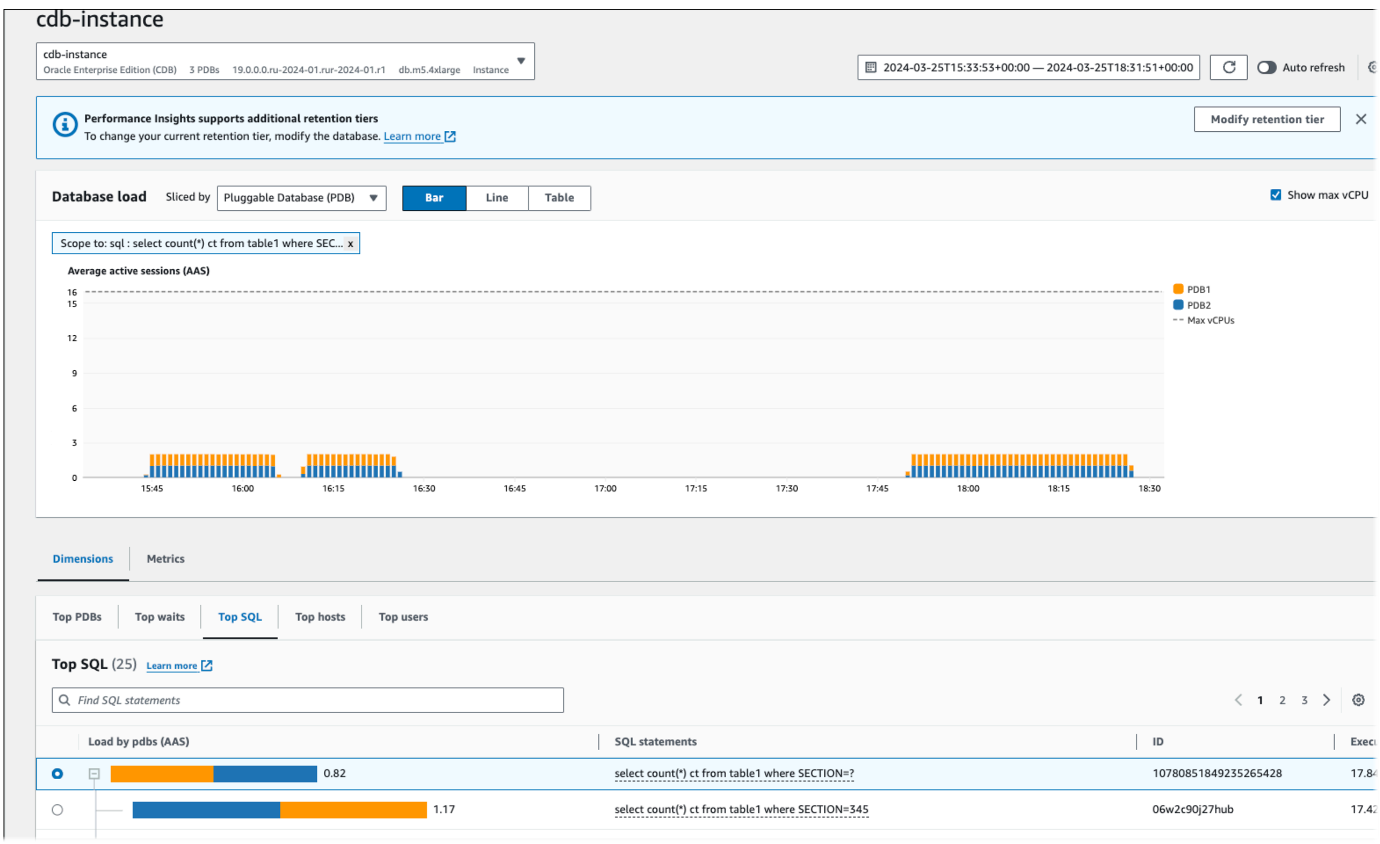
   将显示该数据库实例的 Performance Insights 控制面板。
4. 在数据库负载（DB 负载）部分，选择“划分依据”旁边的可插拔数据库（PDB）。

   平均活动会话图表显示了负载最大的 PDB。PDB 标识符显示在彩色编码方块的右侧。每个标识符都唯一标识一个 PDB。

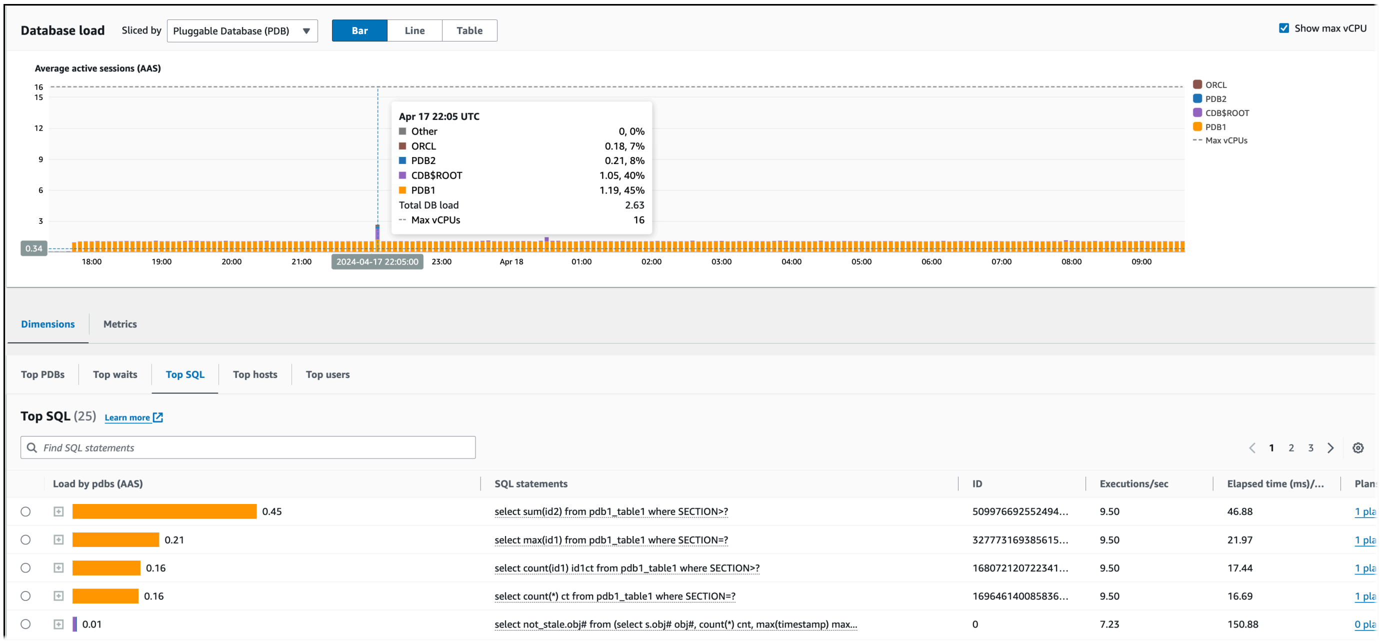


5. 向下滚动至 Top SQL（热门 SQL）选项卡。

在以下示例中，您可以查看相同的 SQL 查询及其驱动到多个 PDB 的负载。



在以下示例中，单个 PDB 处理的负载高于 CDB 中的其他 PDB。



有关 Oracle CDB 的更多信息，请参阅 CDB 和 PDB。

使用 Amazon RDS 的性能详情控制面板分析执行计划

在 Amazon RDS 性能详情控制面板中，您可以找到有关 Oracle 和 SQL Server 数据库实例的执行计划的信息。您可以使用这些信息来了解哪些计划对数据库负载的贡献最大。

要分析 Oracle 或 SQL Server 执行计划，请参阅以下主题。

分析执行计划

- 使用 Amazon RDS 的性能详情控制面板分析 Oracle 执行计划
- 使用 Amazon RDS 的性能详情控制面板分析 SQL Server 执行计划

分析 Amazon RDS 的执行计划概述

您可以使用 Amazon RDS 性能详情控制面板，来了解哪些计划对 Oracle 和 SQL Server 数据库实例的数据库负载贡献最大。

例如，给定时间的主要 SQL 语句可能正在使用下表所示的计划。

主要 SQL	规划
SELECT SUM(amount_sold) FROM sales WHERE prod_id = 10	计划 A
SELECT SUM(amount_sold) FROM sales WHERE prod_id = 521	计划 B
SELECT SUM(s_total) FROM sales WHERE region = 10	计划 A
SELECT * FROM emp WHERE emp_id = 1000	计划 C
SELECT SUM(amount_sold) FROM sales WHERE prod_id = 72	计划 A

使用 Performance Insights 的计划功能，您可以执行以下操作：

- 了解主要 SQL 查询使用哪些计划。

  例如，您可能会发现，大多数数据库负载是由使用计划 A 和计划 B 的查询生成的，只有很小一部分查询使用计划 C。
- 比较同一查询的不同计划。

  在前面的示例中，除产品 ID 外，三个查询都相同。其中两个查询使用计划 A，另一个查询使用计划 B。若要查看两个计划的差异，您可以使用 Performance Insights。
- 了解查询何时切换到新计划。

  您可能会看到查询使用了计划 A，然后在某个时间切换到计划 B。此时数据库是否发生了变化？ 例如，如果表为空，优化程序可能会选择全表扫描。如果表加载了一百万行，优化程序可能会切换至索引范围扫描。
- 深入了解成本最高的计划的具体步骤。

  例如，对于长时间运行的查询，可能会在 equi-join 中显示缺少联接条件。此缺失条件会强制进行笛卡尔联接，将两个表的所有行联接起来。

您可以使用 Performance Insights 的计划捕获功能来执行上述任务。正如您可以按等待事件和主要 SQL 对查询进行切片一样，您也可以按计划维度对查询进行切片。

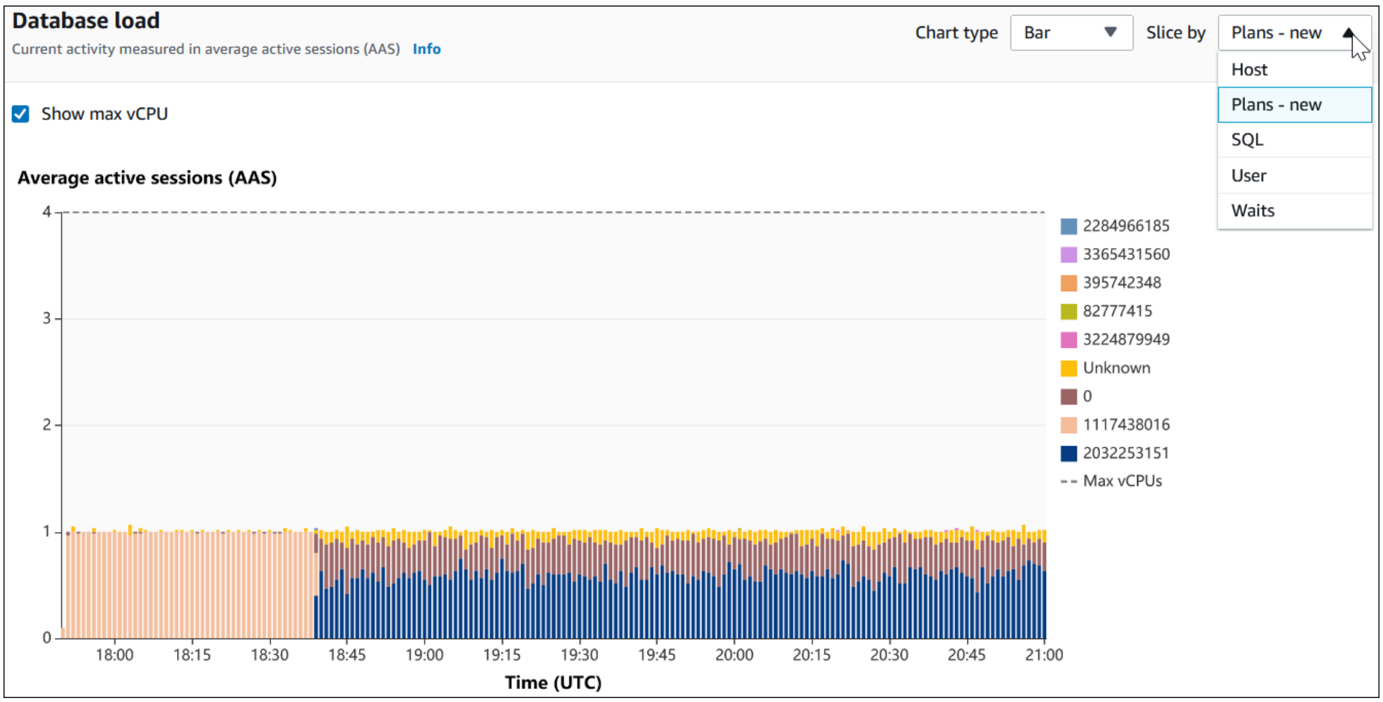
使用 Amazon RDS 的性能详情控制面板分析 Oracle 执行计划

分析 Oracle Database 上的数据库负载时，您可能想了解哪些计划对数据库负载的影响最大。您可以使用性能详情的计划捕获特征，来确定哪些计划对数据库负载的贡献最大。

使用控制台分析 Oracle 执行计划

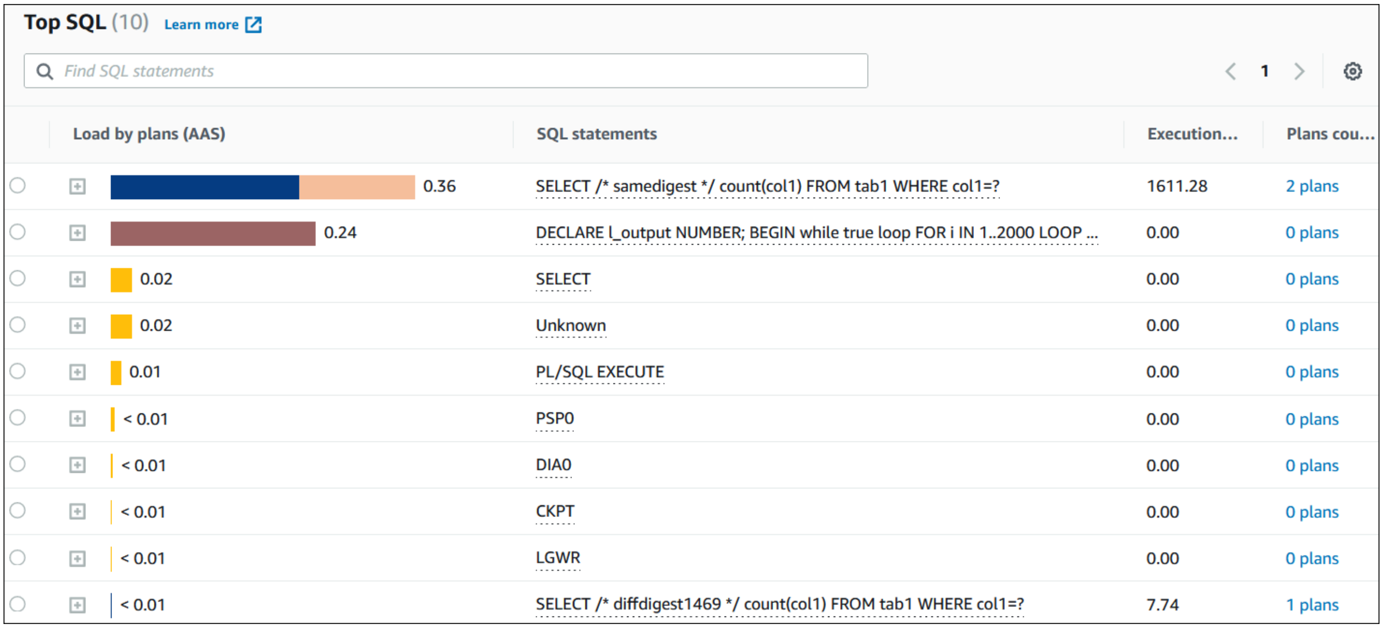
1. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择性能详情。
3. 选择 Oracle 数据库实例。将为该数据库实例显示 Performance Insights 控制面板。
4. 在 Database load (DB load)（数据库负载）部分，选择 Slice by（切片依据）旁边的 Plans（计划）。

   平均活动会话图表显示了主要 SQL 语句使用的计划。计划哈希值显示在彩色方块的右侧。每个哈希值唯一标识每个计划。

5. 向下滚动至 Top SQL（主要 SQL）选项卡。

   在以下示例中，主要 SQL 摘要具有两个计划。您可以通过语句中的问号来判断其是否为摘要。



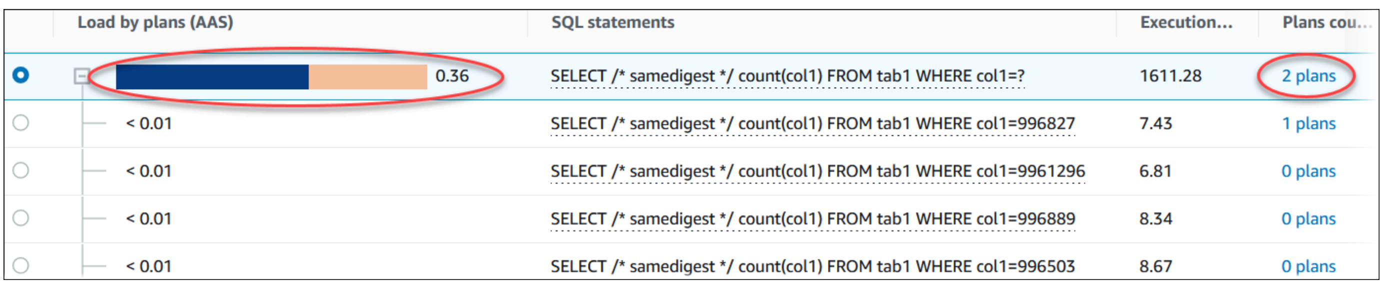
Top SQL (10) Learn more

Find SQL statements

Load by plans (AAS)	SQL statements	Execution...	Plans cou...
0.36	SELECT /* samedigest */ count(col1) FROM tab1 WHERE col1=?	1611.28	2 plans
0.24	DECLARE l_output NUMBER; BEGIN while true loop FOR i IN 1..2000 LOOP ...	0.00	0 plans
0.02	SELECT	0.00	0 plans
0.02	Unknown	0.00	0 plans
0.01	PL/SQL EXECUTE	0.00	0 plans
< 0.01	PSP0	0.00	0 plans
< 0.01	DIA0	0.00	0 plans
< 0.01	CKPT	0.00	0 plans
< 0.01	LGWR	0.00	0 plans
< 0.01	SELECT /* diffdigest1469 */ count(col1) FROM tab1 WHERE col1=?	7.74	1 plans

6. 选择摘要，将其扩展到其组件语句中。

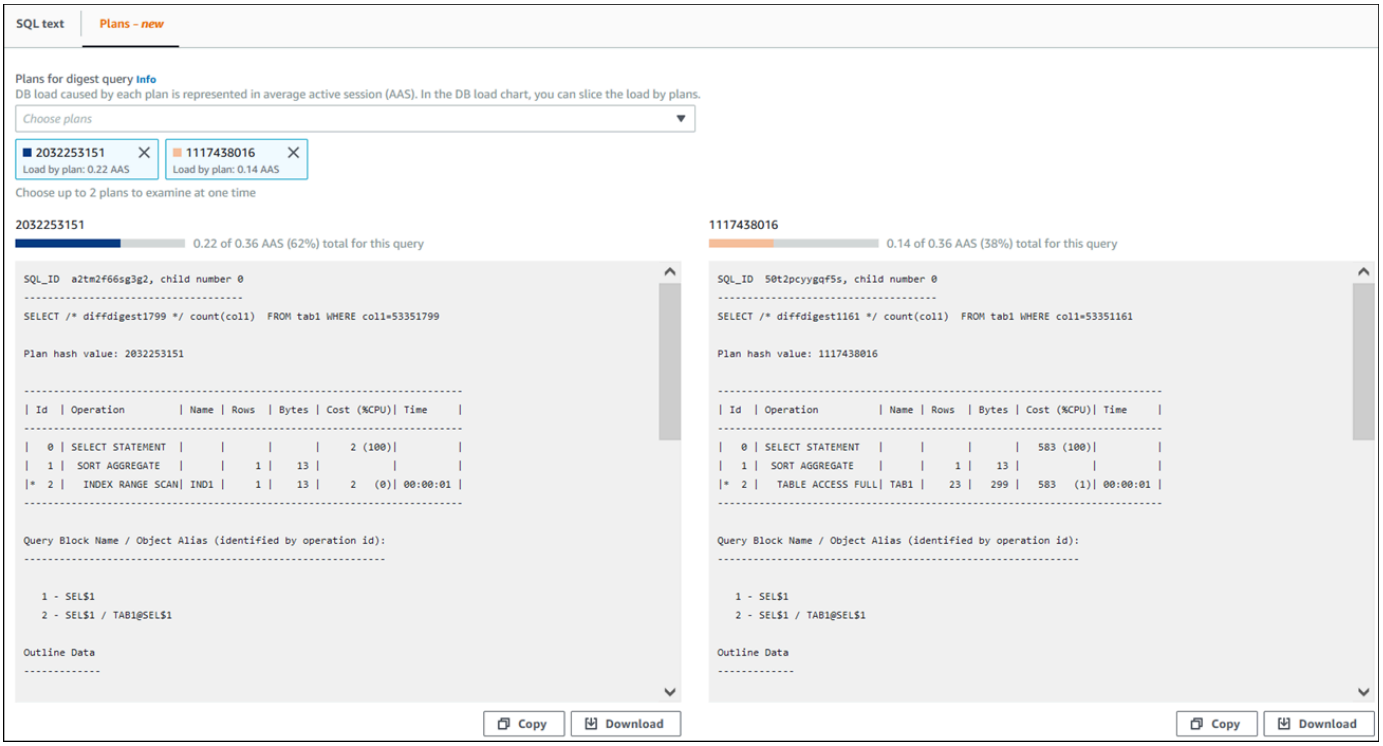
   在以下示例中，SELECT 语句为摘要查询。摘要中的组件查询使用两种不同的计划。计划的颜色对应于数据库负载图表。摘要中的计划总数在显示第二列。



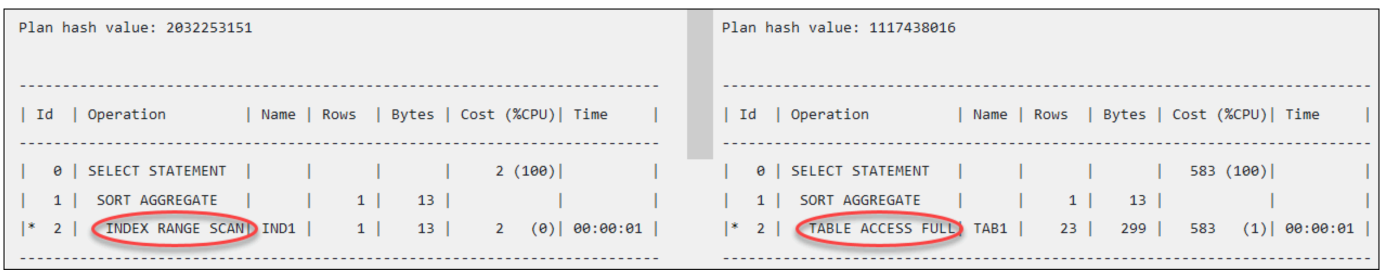
Load by plans (AAS)	SQL statements	Execution...	Plans cou...
0.36	SELECT /* samedigest */ count(col1) FROM tab1 WHERE col1=?	1611.28	2 plans
< 0.01	SELECT /* samedigest */ count(col1) FROM tab1 WHERE col1=996827	7.43	1 plans
< 0.01	SELECT /* samedigest */ count(col1) FROM tab1 WHERE col1=9961296	6.81	0 plans
< 0.01	SELECT /* samedigest */ count(col1) FROM tab1 WHERE col1=996889	8.34	0 plans
< 0.01	SELECT /* samedigest */ count(col1) FROM tab1 WHERE col1=996503	8.67	0 plans

7. 向下滚动，然后从 Plans for digest query（摘要查询计划）列表中选择两个要比较的 Plans（计划）。

   您可以一次查看一个或两个查询计划。以下屏幕截图比较了摘要中的两个计划，哈希值分别为 2032253151 和 1117438016。在以下示例中，运行此摘要查询的平均活动会话中有 62% 使用左侧的计划，而 38% 使用右侧的计划。



在此示例中，两个计划存在明显不同。计划 2032253151 中的步骤 2 使用索引扫描，计划 1117438016 中的步骤 2 使用全表扫描。对于具有大量行的表，使用索引扫描对单行进行查询通常更快。



8. （可选）选择 Copy（复制）将计划复制到剪贴板，或者选择 Download（下载）将计划保存到硬盘。

使用 Amazon RDS 的性能详情控制面板分析 SQL Server 执行计划

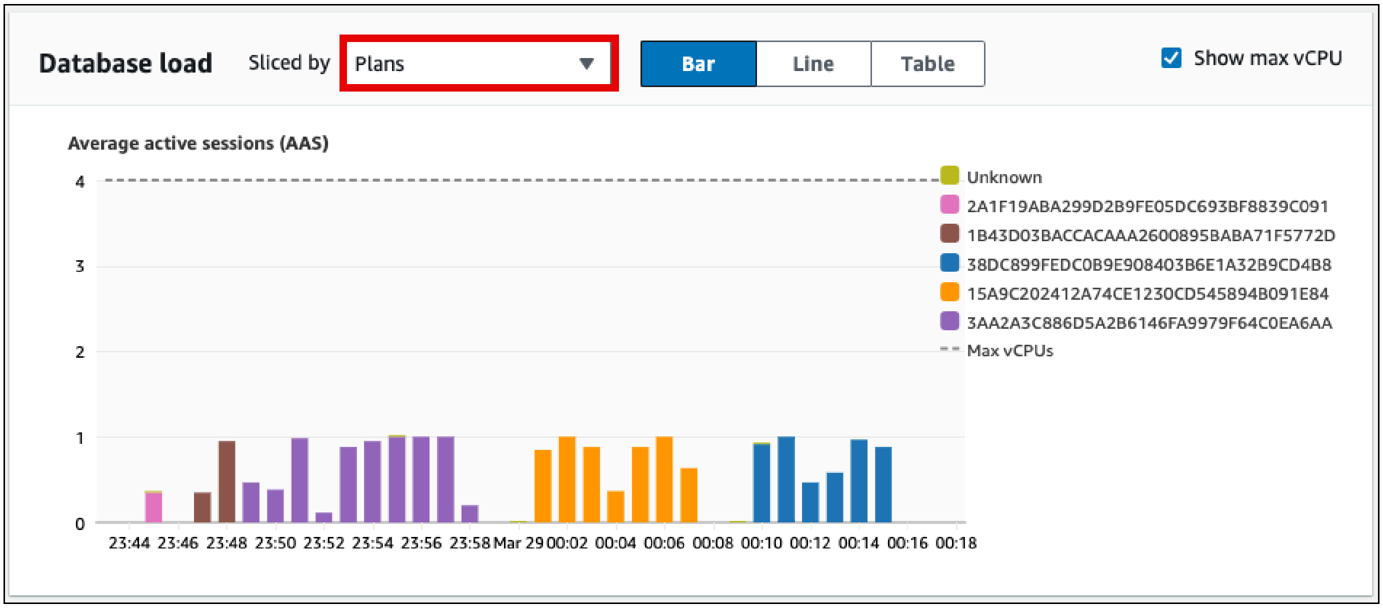
分析 SQL Server 数据库上的数据库负载时，您可能想了解哪些计划对数据库负载的贡献最大。您可以使用性能详情的计划捕获特征，来确定哪些计划对数据库负载的贡献最大。

使用控制台分析 SQL Server 执行计划

1. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。

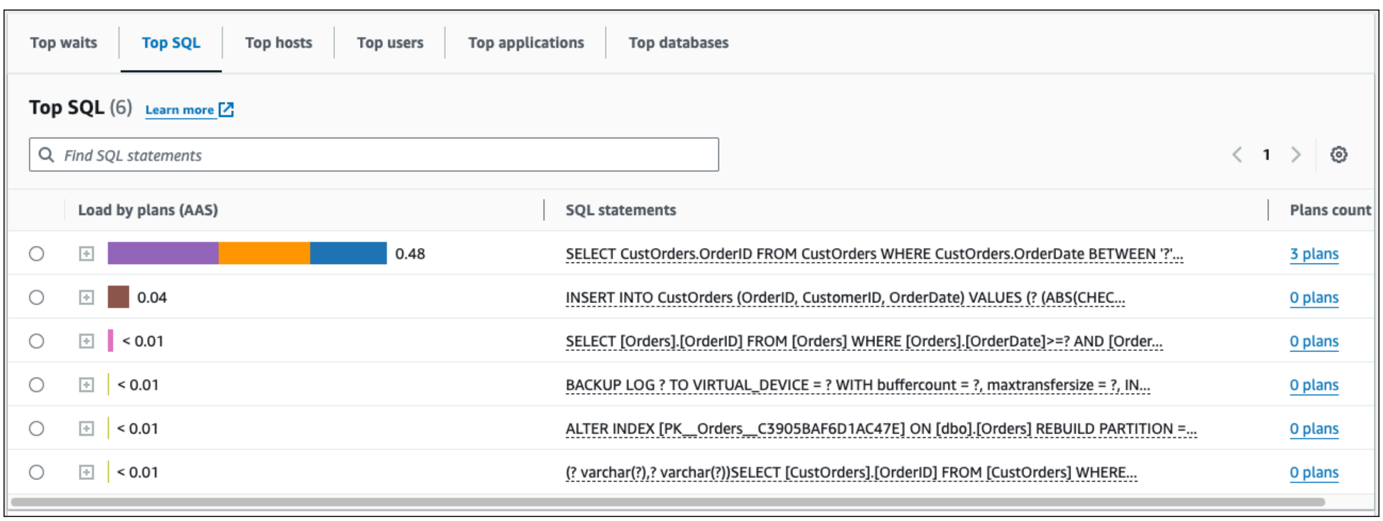
2. 在导航窗格中，选择性能详情。
3. 选择 SQL Server 数据库实例。将为该数据库实例显示 Performance Insights 控制面板。
4. 在 Database load (DB load)（数据库负载）部分，选择 Slice by（切片依据）旁边的 Plans（计划）。

   平均活动会话图表显示了主要 SQL 语句使用的计划。计划哈希值显示在彩色方块的右侧。每个哈希值唯一标识每个计划。



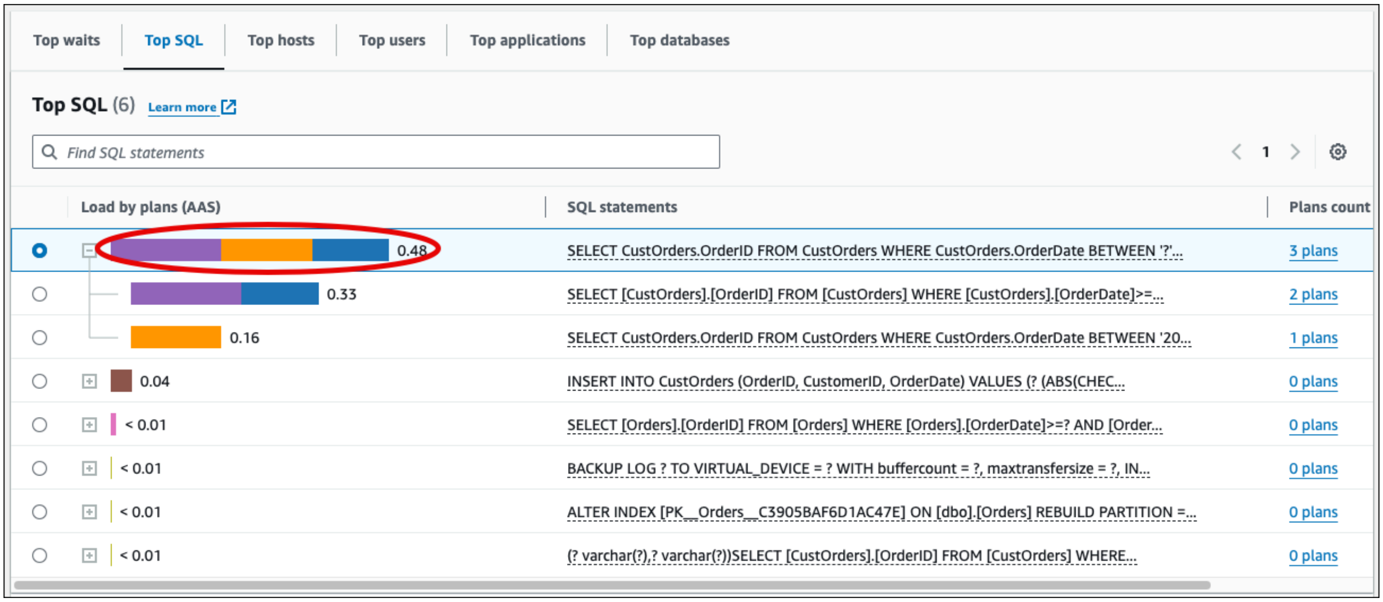
5. 向下滚动至 Top SQL（主要 SQL）选项卡。

   在以下示例中，主要 SQL 摘要具有三个计划。SQL 语句中存在问号表示该语句是摘要。要查看完整的 SQL 语句，请在 SQL 语句列中选择一个值。



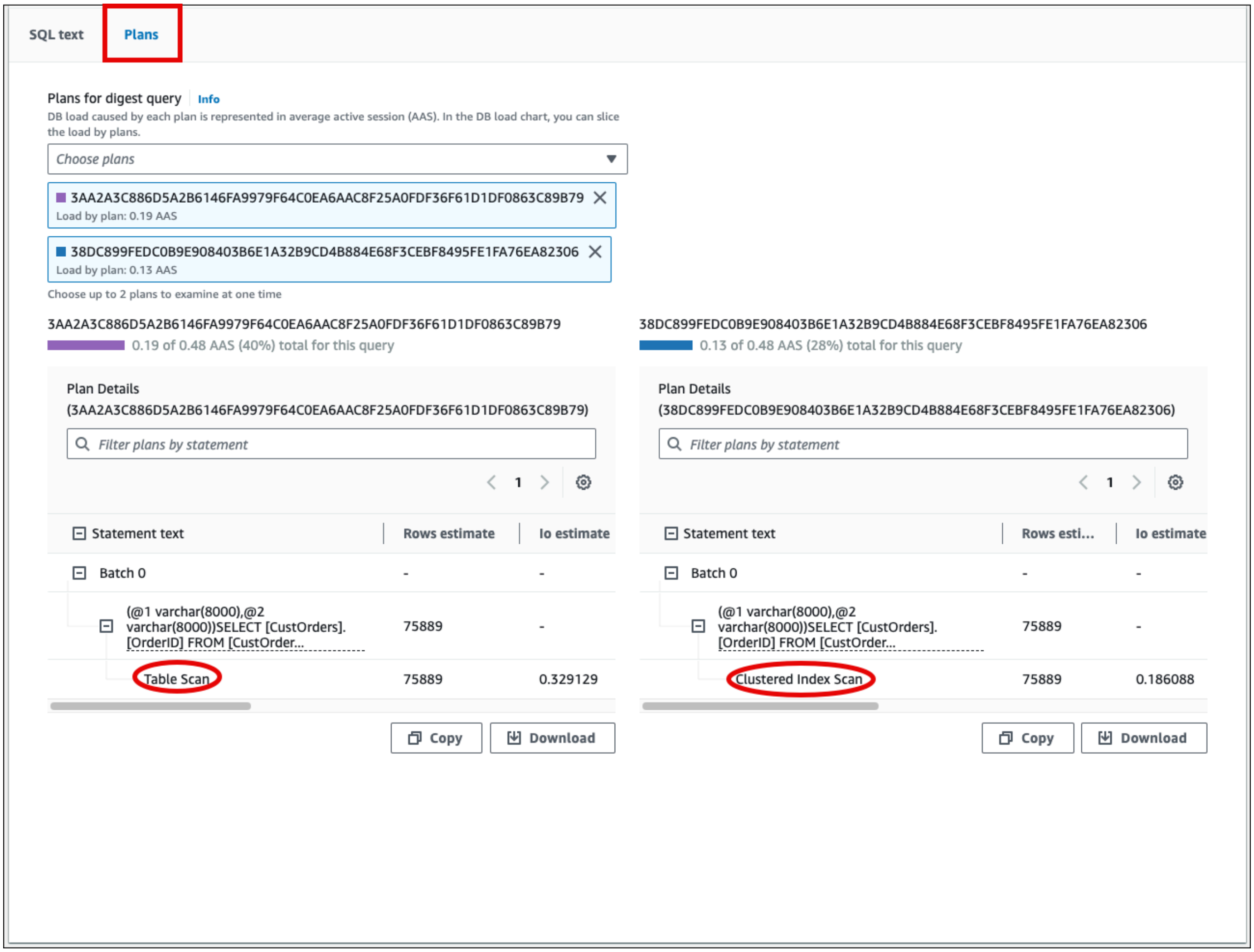
6. 选择摘要，将其扩展到其组件语句中。

在以下示例中，SELECT 语句为摘要查询。摘要中的组件查询使用三种不同的执行计划。分配给计划的颜色对应于数据库负载图表。



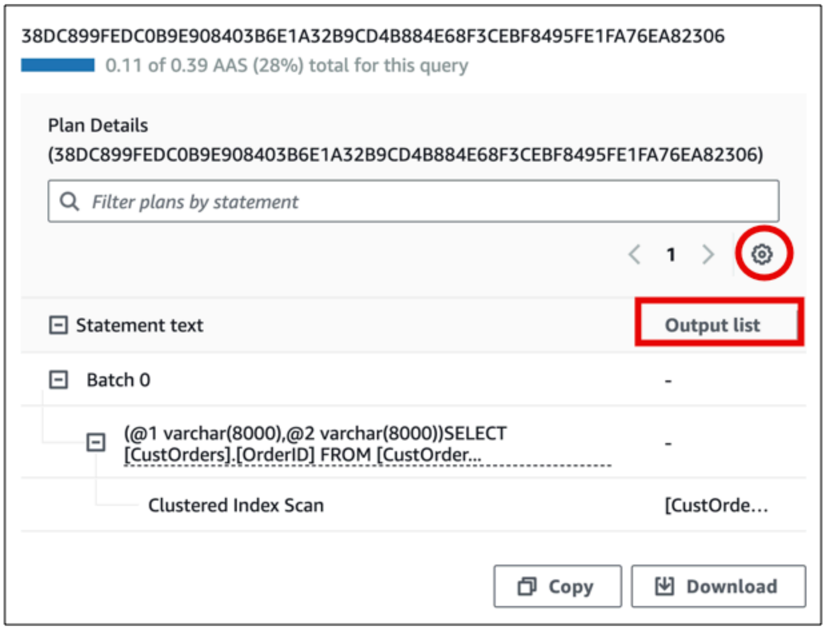
7. 向下滚动，然后从 Plans for digest query（摘要查询计划）列表中选择两个要比较的 Plans（计划）。

   您可以一次查看一个或两个查询计划。以下屏幕截图比较了摘要中的两个计划。在以下示例中，运行此摘要查询的平均活动会话中有 40% 使用左侧的计划，而 28% 使用右侧的计划。



在前面的示例中，这些计划在一个重要方面有所不同。左侧计划中的步骤 2 使用表扫描，而右侧的计划使用聚集索引扫描。对于具有大量行的表，使用聚集索引扫描，检索单行的查询几乎总是更快。

8. （可选）选择“计划详细信息”表上的设置图标来自定义列的可见性和顺序。以下屏幕截图显示了“计划详细信息”表，其第二列是输出列表列。



9. （可选）选择 Copy（复制）将计划复制到剪贴板，或者选择 Download（下载）将计划保存到硬盘。

> **Note**
>
> 性能详情使用分层树表来显示估计的执行计划。该表包括每条语句的部分执行信息。有关“计划详细信息”表中各列的更多信息，请参阅 SQL Server 文档中的 SET SHOWPLAN_ALL。要显示估计的执行计划的完整执行信息，请选择下载来下载该计划，然后将该计划上传到 SQL Server Management Studio。有关使用 SQL Server Management Studio 显示估计的执行计划的更多信息，请参阅 SQL Server 文档中的显示估计的执行计划。

查看性能详情主动建议

Amazon RDS 性能详情可监控特定指标，并自动通过分析指定资源可能存在问题的级别来创建阈值。当新的指标值在给定时间段内超过预定义的阈值时，性能详情会生成主动建议。此建议有助于防止数据库性能将来受到影响。要获得这些主动建议，您必须开启性能详情，并设置付费套餐保留期。

有关开启 Performance Insights 的更多信息，请参阅为 Amazon RDS 打开和关闭性能详情。有关性能详情的定价和数据留存的更多信息，请参阅性能详情的定价和数据留存。

要了解主动建议支持的区域、数据库引擎和实例类别，请参阅支持性能详情功能的 Amazon RDS 数据库引擎、区域和实例类。

您可以在建议详细信息页面中查看主动建议的详细分析和建议的调查。

有关建议的更多信息，请参阅 来自 Amazon RDS 的建议。

查看主动建议的详细分析

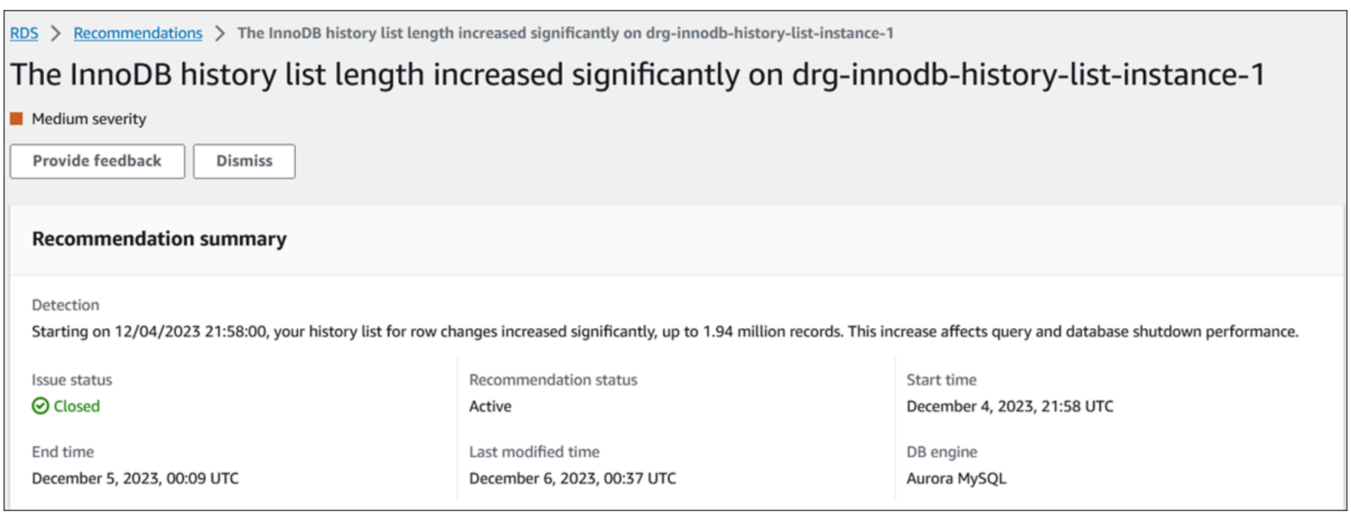
1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，执行以下任何一项操作：
   - 选择建议。

     建议页面显示按您账户中所有资源的严重性排序的建议列表。
   - 选择数据库，然后在数据库页中为资源选择建议。

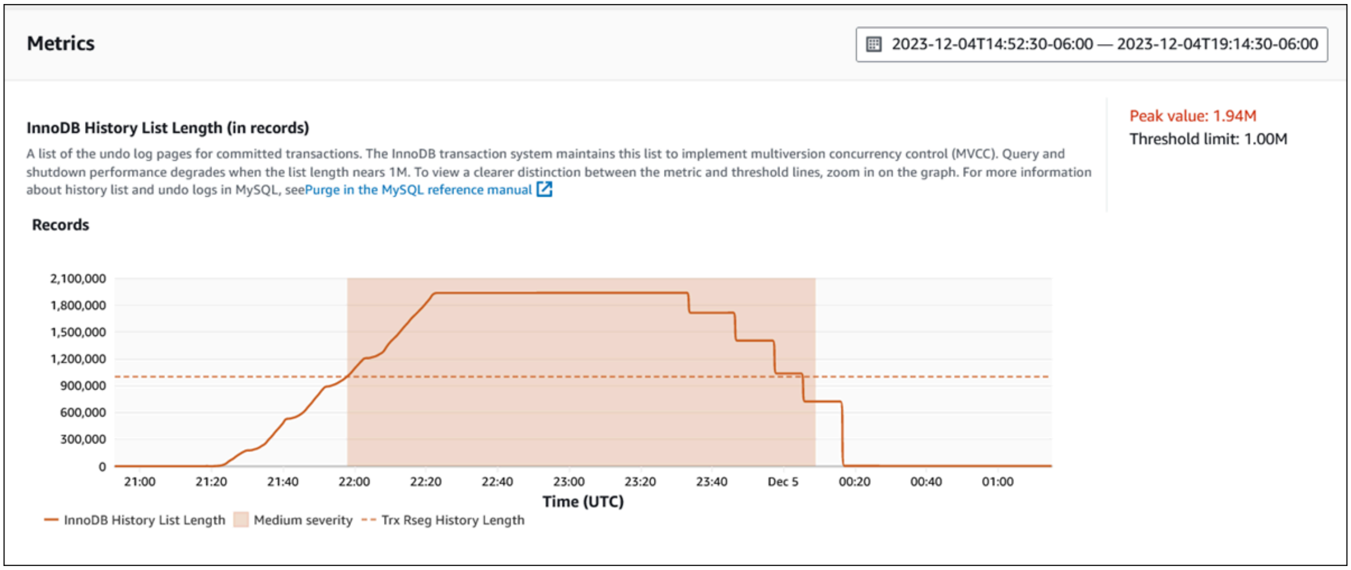
     建议选项卡显示所选资源的建议及其详细信息。
3. 查找主动建议，然后选择查看详细信息。

   此时会显示建议详细信息页面。标题提供了受影响资源的名称以及检测到的问题和严重性。

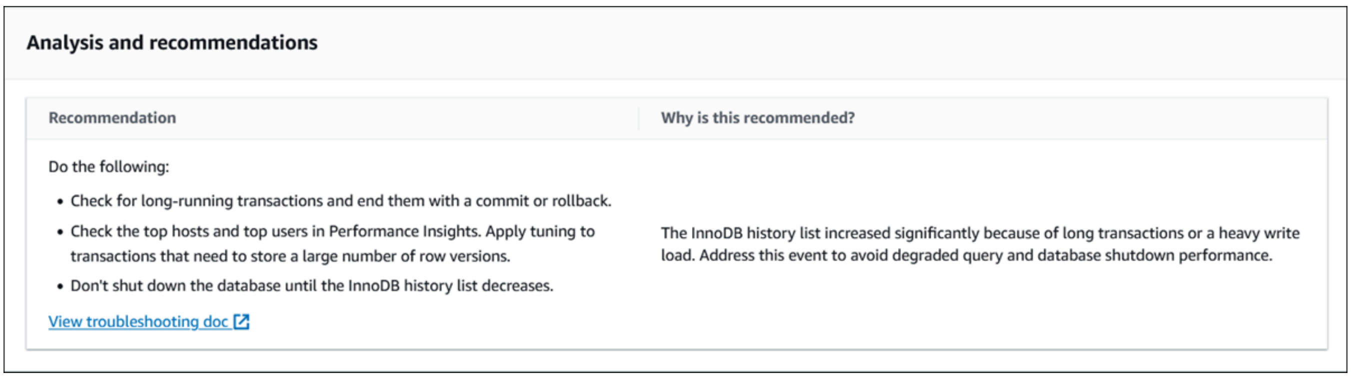
   以下是建议详细信息页面上的组件：
   - 建议摘要 - 检测到的问题、建议和问题状态、问题开始和结束时间、建议修改时间以及引擎类型。

   - 指标 - 检测到的问题的图表。每个图表都显示由资源的基线行为确定的阈值，以及从问题开始时间报告的指标数据。



- 分析和建议 - 建议和所提建议的理由。



您可以查看问题的原因，然后执行建议的操作来解决问题，或者选择右上角的忽略来忽略建议。

使用 Performance Insights API for Amazon RDS 检索指标

开启性能详情后，API 将提供实例性能的可见性。Amazon CloudWatch Logs 日志为Amazon服务提供售卖监控指标的权威源。

Performance Insights 提供了按平均活动会话 (AAS) 衡量的数据库负载的特定于域的视图。对 API 使用者而言，此指标看起来像是二维时间序列数据集。数据的时间维度提供所查询时间范围的每个时间点的数据库负载数据。每个时间点将分解与所请求维度相关的整体负载，如相应时间点测量的 `SQL`、`Wait-event`、`User` 或 `Host`。

Amazon RDS Performance Insights 用于监控您的 Amazon RDS 数据库实例，使您可以分析数据库性能和排查数据库性能问题。查看 Performance Insights 数据的一种方法是在 Amazon Web Services

Management Console中。Performance Insights 还提供公有 API，以便您可以查询自己的数据。您可以使用 API 来执行以下操作：

- 将数据卸载到数据库中
- 将 Performance Insights 数据添加到现有监控控制面板
- 构建监控工具

要使用 Performance Insights API，请在您的 Amazon RDS 数据库实例之一上启用 Performance Insights。有关启用 Performance Insights 的信息，请参阅 为 Amazon RDS 打开和关闭性能详情。有关 Performance Insights API 的更多信息，请参阅 Amazon RDS Performance Insights API 参考。

Performance Insights API 提供以下操作。

Performance Insights 操作	Amazon CLI command	描述
CreatePerformanceAnalysisReport	aws pi create-performance-analysis-report	为数据库实例创建特定时间段的性能分析报告。结果为 AnalysisReportId ，这是报告的唯一标识符。
DeletePerformanceAnalysisReport	aws pi delete-performance-analysis-report	删除性能分析报告。
DescribeDimensionKeys	aws pi describe-dimension-keys	对于特定的时间段，检索指标的前 N 个维度键。
GetDimensionKeyDetails	aws pi get-dimension-key-details	检索数据库实例或数据源的指定维度组的属性。例如，如果您指定了 SQL ID，并且有维度详细信息，则 GetDimensionKeyDetails 将检索与此 ID 关联的维度 db.sql.statement 的全文。此操作很有用，因为 GetResourceMetrics 和 DescribeD

Performance Insights 操作	Amazon CLI command	描述
		imensionKeys 不支持检索大型 SQL 语句文本。
GetPerformanceAnalysisReport	aws pi get-performance-analysis-report	检索报告，包括报告的见解。结果包括报告状态、报告 ID、报告时间详情、见解和建议。
GetResourceMetadata	aws pi get-resource-metadata	检索不同功能的元数据。例如，元数据可以表明特定数据库实例上的某个功能已打开或关闭。
GetResourceMetrics	aws pi get-resource-metrics	检索一组数据来源在一段时间内的 Performance Insights 指标。您可以提供特定维度组和维度，并为每个组提供聚合和筛选条件。
ListAvailableResourceDimensions	aws pi list-available-resource-dimensions	检索特定实例上每个特定指标类型可查询的维度。
ListAvailableResourceMetrics	aws pi list-available-resource-metrics	检索指定指标类型的所有可用指标，指定数据库实例可用该指标进行查询。
ListPerformanceAnalysisReports	aws pi list-performance-analysis-reports	检索数据库实例的所有可用分析报告。这些报告根据每个报告的开始时间列出。
ListTagsForResource	aws pi list-tags-for-resource	列出添加到资源的所有元数据标签。该列表包含标签的名称和值。
TagResource	aws pi tag-resource	将元数据标签添加到 Amazon RDS 资源中。该标签包含名称和值。

Performance Insights 操作	Amazon CLI command	描述
UntagResource	aws pi untag-resource	从资源中删除元数据标签。

有关检索性能详情的时间序列指标和 Amazon CLI 示例的更多信息，请参阅以下主题。

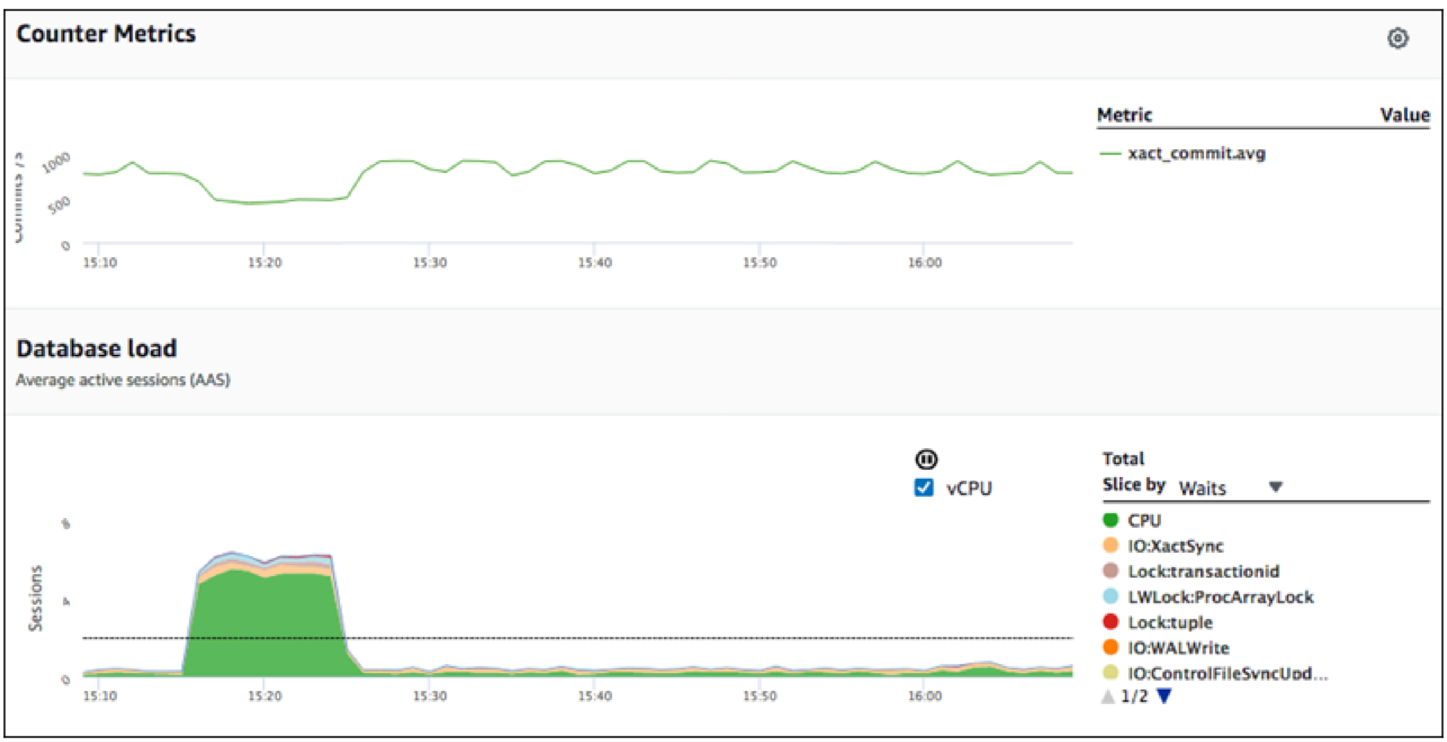
主题

- 检索性能详情的时间序列指标
- Performance Insights 的 Amazon CLI 示例

检索性能详情的时间序列指标

`GetResourceMetrics` 操作从 Performance Insights 数据中检索一个或多个时间序列指标。`GetResourceMetrics` 需要指标和时间段，并返回包含数据点列表的响应。

例如，Amazon Web Services Management Console使用 `GetResourceMetrics` 来填充 Counter Metrics (计数器指标) 图表和 Database Load (数据库负载) 图表，如下图所示。



GetResourceMetrics 返回的所有指标都是标准的时间序列指标，但 db.load 除外。此指标显示在 Database Load (数据库负载) 图表中。db.load 指标不同于其他时间序列指标，因为您可以将它分为称为维度的子组件。在上图中，按组成 db.load 的等待状态对 db.load 进行细分和分组。

> **Note**
>
> GetResourceMetrics 也可以返回 db.sampleload 指标，但 db.load 指标在大多数情况下是合适的。

有关 GetResourceMetrics 返回的计数器指标的信息，请参阅性能详情计数器指标。

指标支持以下计算：

- 平均值 – 指标在一段时间内的平均值。在指标名称后面附加 .avg。
- 最小值 – 指标在一段时间内的最小值。在指标名称后面附加 .min。
- 最大值 – 指标在一段时间内的最大值。在指标名称后面附加 .max。
- 总计 – 指标值在一段时间内的总计。在指标名称后面附加 .sum。
- 样本数 – 在一段时间内收集指标的次数。在指标名称后面附加 .sample_count。

例如，假定在 300 秒（5 分钟）时段内收集指标，并且每分钟收集一次指标。各分钟的值为 1、2、3、4 和 5。在本例中，返回以下计算：

- 平均值 – 3
- 最小值 – 1
- 最大值 – 5
- 总计 – 15
- 样本数 – 5

有关使用 get-resource-metrics Amazon CLI 命令的信息，请参阅 get-resource-metrics。

对于 --metric-queries 选项，请指定一个或多个要获取其结果的查询。每个查询包括必需的 Metric 和可选的 GroupBy 和 Filter 参数。以下是 --metric-queries 选项规范的示例。

```
{
   "Metric": "string",
```

```
    "GroupBy": {
      "Group": "string",
      "Dimensions": ["string", ...],
      "Limit": integer
    },
    "Filter": {"string": "string"
      ...}
```

Performance Insights 的 Amazon CLI 示例

在以下各节中，详细了解性能详情的 Amazon Command Line Interface（Amazon CLI）并使用 Amazon CLI 示例。

主题

- 性能详情的 Amazon CLI 的内置帮助
- 检索计数器指标
- 检索首要等待事件的数据库负载平均值
- 检索首要 SQL 的数据库负载平均值
- 检索按 SQL 筛选的数据库负载平均值
- 检索 SQL 语句的全文
- 创建一段时间的性能分析报告
- 检索性能分析报告
- 列出数据库实例的所有性能分析报告
- 删除性能分析报告
- 向性能分析报告中添加标签
- 列出性能分析报告的所有标签
- 从性能分析报告中删除标签

性能详情的 Amazon CLI 的内置帮助

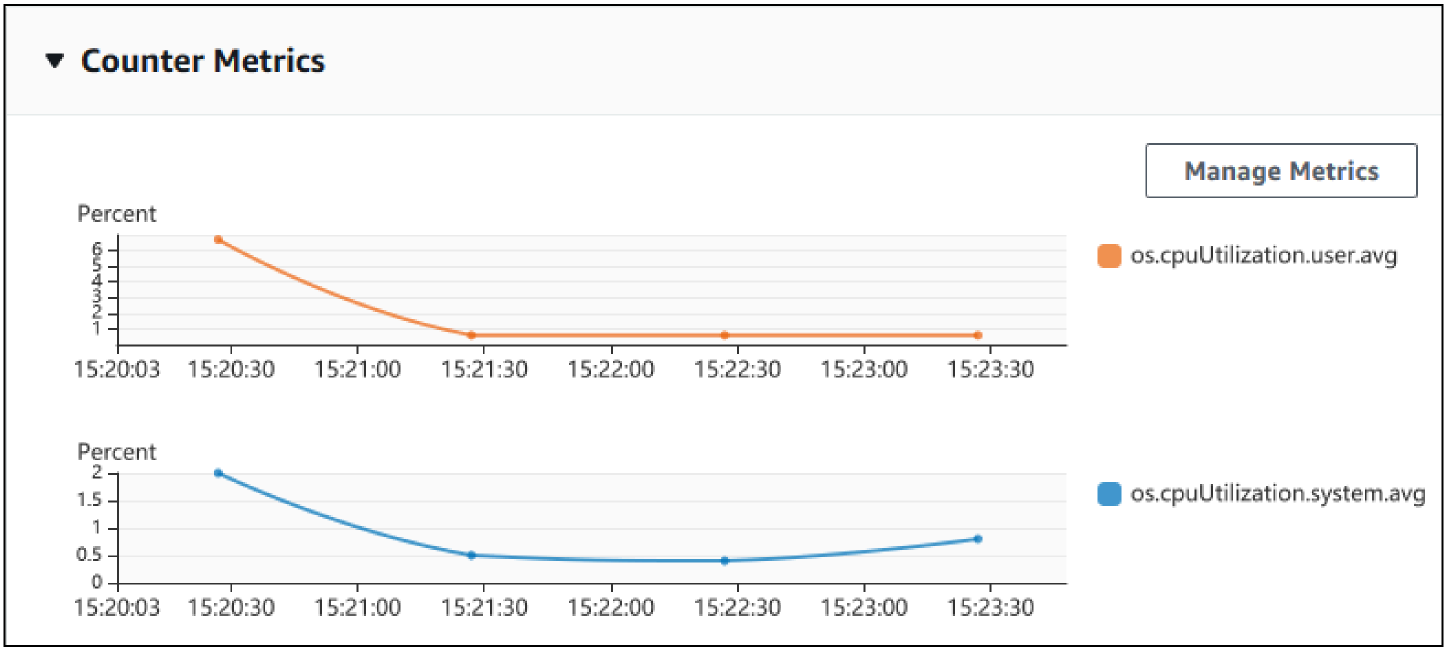
您可以使用 Amazon CLI 查看 Performance Insights 数据。可以通过在命令行上输入以下内容来查看 Performance Insights 的 Amazon CLI 命令的帮助。

```
aws pi help
```

如果尚未安装 Amazon CLI，请参阅《Amazon CLI 用户指南》中的[安装 Amazon CLI](#) 来了解安装信息。

检索计数器指标

以下屏幕截图显示 Amazon Web Services Management Console 中的两个计数器指标图表。



以下示例显示如何收集 Amazon Web Services Management Console 用于生成两个计数器指标图表的相同数据。

对于 Linux、macOS 或 Unix：

```
aws pi get-resource-metrics \
   --service-type RDS \
   --identifier db-ID \
   --start-time 2018-10-30T00:00:00Z \
   --end-time   2018-10-30T01:00:00Z \
   --period-in-seconds 60 \
   --metric-queries '[{"Metric": "os.cpuUtilization.user.avg"  },
                      {"Metric": "os.cpuUtilization.idle.avg"}]'
```

对于 Windows：

```
aws pi get-resource-metrics ^
   --service-type RDS ^
   --identifier db-ID ^
   --start-time 2018-10-30T00:00:00Z ^
```

```
    --end-time   2018-10-30T01:00:00Z ^
    --period-in-seconds 60 ^
    --metric-queries '[{"Metric": "os.cpuUtilization.user.avg"  },
                       {"Metric": "os.cpuUtilization.idle.avg"}]'
```

还可以通过为 `--metrics-query` 选项指定文件来使命令更易于读取。以下示例为该选项使用名为 query.json 的文件。此文件具有以下内容。

```
[
    {
        "Metric": "os.cpuUtilization.user.avg"
    },
    {
        "Metric": "os.cpuUtilization.idle.avg"
    }
]
```

运行以下命令来使用此文件。

对于 Linux、macOS 或 Unix：

```
aws pi get-resource-metrics \
   --service-type RDS \
   --identifier db-ID \
   --start-time 2018-10-30T00:00:00Z \
   --end-time   2018-10-30T01:00:00Z \
   --period-in-seconds 60 \
   --metric-queries file://query.json
```

对于 Windows：

```
aws pi get-resource-metrics ^
   --service-type RDS ^
   --identifier db-ID ^
   --start-time 2018-10-30T00:00:00Z ^
   --end-time   2018-10-30T01:00:00Z ^
   --period-in-seconds 60 ^
   --metric-queries file://query.json
```

上一个示例为各选项指定了以下值：

- `--service-type` – 对于 Amazon RDS 来说为 RDS

- `--identifier` – 数据库实例的资源 ID
- `--start-time` 和 `--end-time` – 要查询的期间的 ISO 8601 `DateTime` 值，支持多种格式

它查询一小时时间范围：

- `--period-in-seconds` – 对于每分钟查询来说为 `60`
- `--metric-queries` – 两个查询的数组，每个查询只用于一个指标。

  指标名称使用点在有用的类别中分类指标，最后一个元素是函数。在示例中，对于每个查询来说，此函数是 `avg`。与 Amazon CloudWatch 一样，支持的函数为 `min`、`max`、`total` 和 `avg`。

响应类似于以下内容。

```
{
    "Identifier": "db-XXX",
    "AlignedStartTime": 1540857600.0,
    "AlignedEndTime": 1540861200.0,
    "MetricList": [
        { //A list of key/datapoints
            "Key": {
                "Metric": "os.cpuUtilization.user.avg" //Metric1
            },
            "DataPoints": [
                //Each list of datapoints has the same timestamps and same number of
 items
                {
                    "Timestamp": 1540857660.0, //Minute1
                    "Value": 4.0
                },
                {
                    "Timestamp": 1540857720.0, //Minute2
                    "Value": 4.0
                },
                {
                    "Timestamp": 1540857780.0, //Minute 3
                    "Value": 10.0
                }
                //... 60 datapoints for the os.cpuUtilization.user.avg metric
            ]
        },
        {
```

```
            "Key": {
                "Metric": "os.cpuUtilization.idle.avg" //Metric2
            },
            "DataPoints": [
                {
                    "Timestamp": 1540857660.0, //Minute1
                    "Value": 12.0
                },
                {
                    "Timestamp": 1540857720.0, //Minute2
                    "Value": 13.5
                },
                //... 60 datapoints for the os.cpuUtilization.idle.avg metric
            ]
        }
    ] //end of MetricList
} //end of response
```

响应具有 Identifier、AlignedStartTime 和 AlignedEndTime。但 --period-in-seconds 值为 60，开始和结束时间已与分钟对齐。如果 --period-in-seconds 为 3600，则开始和结束时间已与小时对齐。

响应中的 MetricList 具有许多条目，每个条目具有 Key 和 DataPoints 条目。每个 DataPoint 具有 Timestamp 和 Value。每个 Datapoints 列表具有 60 个数据点，因为查询针对一小时内的每分钟数据，具有 Timestamp1/Minute1、Timestamp2/Minute2 等，一直到 Timestamp60/Minute60。

因为查询用于两个不同的计数器指标，响应 MetricList 中有两个元素。

检索首要等待事件的数据库负载平均值

以下示例是 Amazon Web Services Management Console用于生成堆栈区域线图的相同查询。此示例检索按前七个等待事件划分负载的最后一个小时的 db.load.avg。命令与 检索计数器指标 中的命令相同。不过，query.json 文件具有以下内容。

```
[
    {
        "Metric": "db.load.avg",
        "GroupBy": { "Group": "db.wait_event", "Limit": 7 }
    }
]
```

运行以下命令。

对于 Linux、macOS 或 Unix：

```
aws pi get-resource-metrics \
   --service-type RDS \
   --identifier db-ID \
   --start-time 2018-10-30T00:00:00Z \
   --end-time   2018-10-30T01:00:00Z \
   --period-in-seconds 60 \
   --metric-queries file://query.json
```

对于 Windows：

```
aws pi get-resource-metrics ^
   --service-type RDS ^
   --identifier db-ID ^
   --start-time 2018-10-30T00:00:00Z ^
   --end-time   2018-10-30T01:00:00Z ^
   --period-in-seconds 60 ^
   --metric-queries file://query.json
```

此示例指定指标 `db.load.avg` 和前七个等待事件的 `GroupBy`。有关此示例的有效值的详细信息，请参阅 Performance Insights API 参考 中的 DimensionGroup。

响应类似于以下内容。

```
{
    "Identifier": "db-XXX",
    "AlignedStartTime": 1540857600.0,
    "AlignedEndTime": 1540861200.0,
    "MetricList": [
        { //A list of key/datapoints
            "Key": {
                //A Metric with no dimensions. This is the total db.load.avg
                "Metric": "db.load.avg"
            },
            "DataPoints": [
                //Each list of datapoints has the same timestamps and same number of
 items
                {
                    "Timestamp": 1540857660.0, //Minute1
```

```
                    "Value": 0.5166666666666667
                },
                {
                    "Timestamp": 1540857720.0, //Minute2
                    "Value": 0.38333333333333336
                },
                {
                    "Timestamp": 1540857780.0, //Minute 3
                    "Value": 0.26666666666666666
                }
                //... 60 datapoints for the total db.load.avg key
            ]
        },
        {
            "Key": {
                //Another key. This is db.load.avg broken down by CPU
                "Metric": "db.load.avg",
                "Dimensions": {
                    "db.wait_event.name": "CPU",
                    "db.wait_event.type": "CPU"
                }
            },
            "DataPoints": [
                {
                    "Timestamp": 1540857660.0, //Minute1
                    "Value": 0.35
                },
                {
                    "Timestamp": 1540857720.0, //Minute2
                    "Value": 0.15
                },
                //... 60 datapoints for the CPU key
            ]
        },
        //... In total we have 8 key/datapoints entries, 1) total, 2-8) Top Wait Events
    ] //end of MetricList
} //end of response
```

在此响应中，`MetricList` 中有八个条目。有一个有关总 `db.load.avg` 的条目，还有七个条目，其中每个条目关于按前七个等待事件之一划分的 `db.load.avg`。与第一个示例不同，因为具有分组维度，所以必须具有一个用于每个指标分组的键。不能像在基本计数器指标使用案例中那样每个指标只有一个键。

检索首要 SQL 的数据库负载平均值

以下示例按前 10 个 SQL 语句对 db.wait_events 进行分组。有两个不同的 SQL 语句组：

- db.sql – 完整的 SQL 语句，例如 select * from customers where customer_id = 123
- db.sql_tokenized – 令牌化的 SQL 语句，例如 select * from customers where customer_id = ?

在分析数据库性能时，将仅参数不同的 SQL 语句视为一个逻辑项目很有用。因此，您在查询时可以使用 db.sql_tokenized。不过，尤其在您对说明计划感兴趣时，查看带参数的完整 SQL 语句和按 db.sql 分组查询有时会更有用。令牌化和完整 SQL 之间存在父-子关系，多个完整 SQL（子级）分组在同一令牌化 SQL（父级）下。

此示例中的命令类似于 检索首要等待事件的数据库负载平均值 中的命令。不过，query.json 文件具有以下内容。

```
[
    {
        "Metric": "db.load.avg",
        "GroupBy": { "Group": "db.sql_tokenized", "Limit": 10 }
    }
]
```

下面的示例使用了 db.sql_tokenized。

对于 Linux、macOS 或 Unix：

```
aws pi get-resource-metrics \
   --service-type RDS \
   --identifier db-ID \
   --start-time 2018-10-29T00:00:00Z \
   --end-time   2018-10-30T00:00:00Z \
   --period-in-seconds 3600 \
   --metric-queries file://query.json
```

对于 Windows：

```
aws pi get-resource-metrics ^
   --service-type RDS ^
   --identifier db-ID ^
   --start-time 2018-10-29T00:00:00Z ^
```

```
    --end-time   2018-10-30T00:00:00Z  ^
    --period-in-seconds 3600 ^
    --metric-queries file://query.json
```

此示例查询 24 小时，以秒为单位的期间为一小时。

此示例指定指标 db.load.avg 和前七个等待事件的 GroupBy。有关此示例的有效值的详细信息，请参阅 Performance Insights API 参考 中的 DimensionGroup。

响应类似于以下内容。

```
{
    "AlignedStartTime": 1540771200.0,
    "AlignedEndTime": 1540857600.0,
    "Identifier": "db-XXX",

    "MetricList": [ //11 entries in the MetricList
        {
            "Key": { //First key is total
                "Metric": "db.load.avg"
            }
            "DataPoints": [ //Each DataPoints list has 24 per-hour Timestamps and a
 value
                {
                    "Value": 1.6964980544747081,
                    "Timestamp": 1540774800.0
                },
                //... 24 datapoints
            ]
        },
        {
            "Key": { //Next key is the top tokenized SQL
                "Dimensions": {
                    "db.sql_tokenized.statement": "INSERT INTO authors (id,name,email)
 VALUES\n( nextval(?)  ,?,?)",
                    "db.sql_tokenized.db_id": "pi-2372568224",
                    "db.sql_tokenized.id": "AKIAIOSFODNN7EXAMPLE"
                },
                "Metric": "db.load.avg"
            },
            "DataPoints": [ //... 24 datapoints
            ]
        },
```

```
        // In total 11 entries, 10 Keys of top tokenized SQL, 1 total key
    ] //End of MetricList
} //End of response
```

此响应的 MetricList 中具有 11 个条目（1 个总计，10 个首要令牌化 SQL），其中每个条目具有 24 个每小时 DataPoints。

对于令牌化 SQL，每个维度列表中具有三个条目：

- db.sql_tokenized.statement – 令牌化 SQL 语句。
- db.sql_tokenized.db_id – 用于引用 SQL 的本机数据库 ID，或 Performance Insights 为您生成的合成 ID（如果本机数据库 ID 不可用）。此示例返回 pi-2372568224 合成 ID。
- db.sql_tokenized.id – Performance Insights 中的查询的 ID。

  在 Amazon Web Services Management Console中，此 ID 称为支持 ID。它如此命名是因为 ID 是 Amazon 支持服务可检查以帮助您解决数据库问题的数据。Amazon 极其重视您的数据的安全性和隐私，几乎所有数据都使用您的 Amazon KMS 密钥进行加密存储。因此，Amazon 中的任何人都无法查看这些数据。在上一个示例中，tokenized.statement 和 tokenized.db_id 都进行了加密存储。如果您的数据库出现问题，Amazon Support 可以通过引用支持 ID 来帮助您。

在查询时，在 Group 中指定 GroupBy 可能很方便。不过，要更精细地控制返回的数据，请指定维度列表。例如，如果所需的所有内容是 db.sql_tokenized.statement，则可将 Dimensions 属性添加到 query.json 文件中。

```
[
    {
        "Metric": "db.load.avg",
        "GroupBy": {
            "Group": "db.sql_tokenized",
            "Dimensions":["db.sql_tokenized.statement"],
            "Limit": 10
        }
    }
]
```

检索按 SQL 筛选的数据库负载平均值

上图显示选择了特定查询，并且首要平均活动会话堆栈区域线图限定为该查询。虽然查询仍针对前七个总体等待事件，已筛选响应的值。筛选条件使它只考虑与特定筛选条件匹配的会话。

此示例中的相应 API 查询类似于 检索首要 SQL 的数据库负载平均值 中的命令。不过，query.json 文件具有以下内容。

```
[
 {
        "Metric": "db.load.avg",
        "GroupBy": { "Group": "db.wait_event", "Limit": 5  },
        "Filter": { "db.sql_tokenized.id": "AKIAIOSFODNN7EXAMPLE" }
    }
]
```

对于 Linux、macOS 或 Unix：

```
aws pi get-resource-metrics \
   --service-type RDS \
   --identifier db-ID \
   --start-time 2018-10-30T00:00:00Z \
   --end-time   2018-10-30T01:00:00Z \
```

```
    --period-in-seconds 60 \
    --metric-queries file://query.json
```

对于 Windows：

```
aws pi get-resource-metrics ^
   --service-type RDS ^
   --identifier db-ID ^
   --start-time 2018-10-30T00:00:00Z ^
   --end-time   2018-10-30T01:00:00Z ^
   --period-in-seconds 60 ^
   --metric-queries file://query.json
```

响应类似于以下内容。


```
{
    "Identifier": "db-XXX",
    "AlignedStartTime": 1556215200.0,
    "MetricList": [
        {
            "Key": {
                "Metric": "db.load.avg"
            },
            "DataPoints": [
                {
                    "Timestamp": 1556218800.0,
                    "Value": 1.4878117913832196
                },
                {
                    "Timestamp": 1556222400.0,
                    "Value": 1.192823803967328
                }
            ]
        },
        {
            "Key": {
                "Metric": "db.load.avg",
                "Dimensions": {
                    "db.wait_event.type": "io",
                    "db.wait_event.name": "wait/io/aurora_redo_log_flush"
                }
            },
            "DataPoints": [
```

```
                {
                    "Timestamp": 1556218800.0,
                    "Value": 1.1360544217687074
                },
                {
                    "Timestamp": 1556222400.0,
                    "Value": 1.058051341890315
                }
            ]
        },
        {
            "Key": {
                "Metric": "db.load.avg",
                "Dimensions": {
                    "db.wait_event.type": "io",
                    "db.wait_event.name": "wait/io/table/sql/handler"
                }
            },
            "DataPoints": [
                {
                    "Timestamp": 1556218800.0,
                    "Value": 0.16241496598639457
                },
                {
                    "Timestamp": 1556222400.0,
                    "Value": 0.05163360560093349
                }
            ]
        },
        {
            "Key": {
                "Metric": "db.load.avg",
                "Dimensions": {
                    "db.wait_event.type": "synch",
                    "db.wait_event.name": "wait/synch/mutex/innodb/
aurora_lock_thread_slot_futex"
                }
            },
            "DataPoints": [
                {
                    "Timestamp": 1556218800.0,
                    "Value": 0.11479591836734694
                },
                {
```

```
                    "Timestamp": 1556222400.0,
                    "Value": 0.013127187864644107
                }
            ]
        },
        {
            "Key": {
                "Metric": "db.load.avg",
                "Dimensions": {
                    "db.wait_event.type": "CPU",
                    "db.wait_event.name": "CPU"
                }
            },
            "DataPoints": [
                {
                    "Timestamp": 1556218800.0,
                    "Value": 0.05215419501133787
                },
                {
                    "Timestamp": 1556222400.0,
                    "Value": 0.05805134189031505
                }
            ]
        },
        {
            "Key": {
                "Metric": "db.load.avg",
                "Dimensions": {
                    "db.wait_event.type": "synch",
                    "db.wait_event.name": "wait/synch/mutex/innodb/lock_wait_mutex"
                }
            },
            "DataPoints": [
                {
                    "Timestamp": 1556218800.0,
                    "Value": 0.017573696145124718
                },
                {
                    "Timestamp": 1556222400.0,
                    "Value": 0.002333722287047841
                }
            ]
        }
    ],
```

```
    "AlignedEndTime": 1556222400.0
} //end of response
```

在此响应中，根据 query.json 文件中指定的令牌化 SQL AKIAIOSFODNN7EXAMPLE 的贡献筛选所有值。键还可能遵循与没有筛选条件的查询不同的顺序，因为前五个等待事件影响了筛选的 SQL。

检索 SQL 语句的全文

以下示例检索数据库实例 `db-10BCD2EFGHIJ3KL4M5NO6PQRS5` 的 SQL 语句的全文。`--group` 是 `db.sql`，`--group-identifier` 是 `db.sql.id`。在此示例中，*`my-sql-id`* 表示通过调用 `pi get-resource-metrics` 或 `pi describe-dimension-keys` 检索的 SQL ID。

运行以下命令。

对于 Linux、macOS 或 Unix：

```
aws pi get-dimension-key-details \
   --service-type RDS \
   --identifier db-10BCD2EFGHIJ3KL4M5NO6PQRS5 \
   --group db.sql \
   --group-identifier my-sql-id \
   --requested-dimensions statement
```

对于 Windows：

```
aws pi get-dimension-key-details ^
   --service-type RDS ^
   --identifier db-10BCD2EFGHIJ3KL4M5NO6PQRS5 ^
   --group db.sql ^
   --group-identifier my-sql-id ^
   --requested-dimensions statement
```

在此示例中，有维度详细信息。因此，Performance Insights 将检索 SQL 语句的全文，而不会将其截断。

```
{
    "Dimensions":[
    {
        "Value": "SELECT e.last_name, d.department_name FROM employees e, departments d
 WHERE e.department_id=d.department_id",
        "Dimension": "db.sql.statement",
        "Status": "AVAILABLE"
```

```
    },
    ...
    ]
}
```

创建一段时间的性能分析报告

以下示例为 `db-loadtest-0` 数据库创建了一个开始时间为 `1682969503`、结束时间为 `1682979503` 的性能分析报告。

```
aws pi create-performance-analysis-report \
        --service-type RDS \
        --identifier db-loadtest-0 \
        --start-time 1682969503 \
        --end-time 1682979503 \
        --region us-west-2
```

响应是报告的唯一标识符 `report-0234d3ed98e28fb17`。

```
{
   "AnalysisReportId": "report-0234d3ed98e28fb17"
}
```

检索性能分析报告

以下示例检索 `report-0d99cc91c4422ee61` 报告的分析报告详细信息。

```
aws pi get-performance-analysis-report \
--service-type RDS \
--identifier db-loadtest-0 \
--analysis-report-id report-0d99cc91c4422ee61 \
--region us-west-2
```

响应提供报告状态、ID、时间详细信息和见解。

```
        {
    "AnalysisReport": {
        "Status": "Succeeded",
        "ServiceType": "RDS",
        "Identifier": "db-loadtest-0",
        "StartTime": 1680583486.584,
```

```
        "AnalysisReportId": "report-0d99cc91c4422ee61",
        "EndTime": 1680587086.584,
        "CreateTime": 1680587087.139,
        "Insights": [
           ... (Condensed for space)
        ]
    }
}
```

列出数据库实例的所有性能分析报告

以下示例列出了 `db-loadtest-0` 数据库的所有可用的性能分析报告。

```
aws pi list-performance-analysis-reports \
--service-type RDS \
--identifier db-loadtest-0 \
--region us-west-2
```

响应列出了所有带有报告 ID、状态和时间段详细信息的报告。

```
{
    "AnalysisReports": [
        {
            "Status": "Succeeded",
            "EndTime": 1680587086.584,
            "CreationTime": 1680587087.139,
            "StartTime": 1680583486.584,
            "AnalysisReportId": "report-0d99cc91c4422ee61"
        },
        {
            "Status": "Succeeded",
            "EndTime": 1681491137.914,
            "CreationTime": 1681491145.973,
            "StartTime": 1681487537.914,
            "AnalysisReportId": "report-002633115cc002233"
        },
        {
            "Status": "Succeeded",
            "EndTime": 1681493499.849,
            "CreationTime": 1681493507.762,
            "StartTime": 1681489899.849,
            "AnalysisReportId": "report-043b1e006b47246f9"
        },
```

```
        {
            "Status": "InProgress",
            "EndTime": 1682979503.0,
            "CreationTime": 1682979618.994,
            "StartTime": 1682969503.0,
            "AnalysisReportId": "report-01ad15f9b88bcbd56"
        }
    ]
}
```

删除性能分析报告

以下示例删除 `db-loadtest-0` 数据库的分析报告。

```
aws pi delete-performance-analysis-report \
--service-type RDS \
--identifier db-loadtest-0 \
--analysis-report-id report-0d99cc91c4422ee61 \
--region us-west-2
```

向性能分析报告中添加标签

以下示例向 `report-01ad15f9b88bcbd56` 报告中添加了一个键为 `name`、值为 `test-tag` 的标签。

```
aws pi tag-resource \
--service-type RDS \
--resource-arn arn:aws:pi:us-west-2:356798100956:perf-reports/RDS/db-loadtest-0/
report-01ad15f9b88bcbd56 \
--tags Key=name,Value=test-tag \
--region us-west-2
```

列出性能分析报告的所有标签

以下示例列出了 `report-01ad15f9b88bcbd56` 报告的所有标签。

```
aws pi list-tags-for-resource \
--service-type RDS \
--resource-arn arn:aws:pi:us-west-2:356798100956:perf-reports/RDS/db-loadtest-0/
report-01ad15f9b88bcbd56 \
--region us-west-2
```

响应列出了添加到报告中的所有标签的值和键：

```
{
    "Tags": [
        {
            "Value": "test-tag",
            "Key": "name"
        }
    ]
}
```

从性能分析报告中删除标签

以下示例从 `report-01ad15f9b88bcbd56` 报告中删除 `name` 标签。

```
aws pi untag-resource \
--service-type RDS \
--resource-arn arn:aws:pi:us-west-2:356798100956:perf-reports/RDS/db-loadtest-0/
report-01ad15f9b88bcbd56 \
--tag-keys name \
--region us-west-2
```

删除标签后，调用 `list-tags-for-resource` API 不会列出此标签。

使用 Amazon CloudTrail 记录 Performance Insights 调用

Performance Insights 与 Amazon CloudTrail 一起运行，后者是一项服务，在 Performance Insights 中提供由用户、角色或 Amazon 服务所执行的操作的记录。CloudTrail 将 Performance Insights 的所有 API 调用作为事件捕获。此捕获包括的调用来自 Amazon RDS 控制台以及来自对性能详情 API 操作的代码调用。

如果您创建跟踪，可以使 CloudTrail 事件持续传送到 Amazon S3 存储桶（包括 Performance Insights 的事件）。如果您不配置跟踪，则仍可在 CloudTrail 控制台中的事件历史记录中查看最新事件。通过使用 CloudTrail 收集的数据，您可以确定特定的信息。此信息包括向 Performance Insights 发出的请求、从中发出请求的 IP 地址、何人发出的请求以及请求的发出时间。它还包括其他详细信息。

要了解有关 CloudTrail 的更多信息，请参阅 Amazon CloudTrail 用户指南。

在 CloudTrail 中使用性能详情信息

在您创建 Amazon 账户时，将在该账户上启用 CloudTrail。当 Performance Insights 中发生活动时，该活动将记录在 CloudTrail 事件中，并与其他 Amazon 服务事件一同保存在 CloudTrail 控制台的

Event history (事件历史记录) 中。您可以在 Amazon 账户中查看、搜索和下载最新事件。有关更多信息，请参阅 Amazon CloudTrail 用户指南中的使用 CloudTrail 事件历史记录查看事件。

要持续记录 Amazon 账户中的事件（包括 Performance Insights 的事件），请创建跟踪。通过跟踪，CloudTrail 可将日志文件传送至 Amazon S3 存储桶。预设情况下，在控制台中创建跟踪时，此跟踪应用于所有 Amazon 区域。此跟踪记录来自 Amazon 分区中的所有 Amazon 区域的事件，并将日志文件传送至您指定的 Amazon S3 存储桶。此外，您可以配置其他 Amazon 服务，进一步分析在 CloudTrail 日志中收集的事件数据并采取行动。有关更多信息，请参阅 Amazon CloudTrail 用户指南中的以下主题：

- 创建跟踪概述
- CloudTrail 支持的服务和集成
- 为 CloudTrail 配置 Amazon SNS 通知
- 从多个区域接收 CloudTrail 日志文件和从多个账户接收 CloudTrail 日志文件

所有 Performance Insights 操作均由 CloudTrail 进行日志记录，您可以在 Performance Insights API 参考中找到这些操作的信息。例如，对 `DescribeDimensionKeys` 和 `GetResourceMetrics` 操作的调用将在 CloudTrail 日志文件中生成条目。

每个事件或日志条目都包含有关生成请求的人员信息。身份信息可帮助您确定以下内容：

- 请求是使用根用户凭证还是 IAM 用户凭证发出的。
- 请求是使用角色还是联合用户的临时安全凭证发出的。
- 请求是否由其他 Amazon 服务发出。

有关更多信息，请参阅 CloudTrail userIdentity 元素。

性能详情日志文件条目

跟踪是一种配置，可用于将事件作为日志文件传送到您指定的 Amazon S3 存储桶。CloudTrail 日志文件包含一个或多个日志条目。事件 表示来自任何源的单个请求。每个事件包括有关所请求操作的信息、操作的日期和时间和请求参数等。CloudTrail 日志文件不是公用 API 调用的有序堆栈跟踪，因此它们不会按任何特定顺序显示。

下面的示例显示了一个 CloudTrail 日志条目，该条目说明了 `GetResourceMetrics` 操作。

```
{
```

```
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "IAMUser",
         "principalId": "AKIAIOSFODNN7EXAMPLE",
        "arn": "arn:aws:iam::123456789012:user/johndoe",
        "accountId": "123456789012",
        "accessKeyId": "AKIAI44QH8DHBEXAMPLE",
        "userName": "johndoe"
    },
    "eventTime": "2019-12-18T19:28:46Z",
    "eventSource": "pi.amazonaws.com",
    "eventName": "GetResourceMetrics",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "72.21.198.67",
    "userAgent": "aws-cli/1.16.240 Python/3.7.4 Darwin/18.7.0 botocore/1.12.230",
    "requestParameters": {
        "identifier": "db-YTDU5J5V66X7CXSCVDFD2V3SZM",
        "metricQueries": [
            {
                "metric": "os.cpuUtilization.user.avg"
            },
            {
                "metric": "os.cpuUtilization.idle.avg"
            }
        ],
        "startTime": "Dec 18, 2019 5:28:46 PM",
        "periodInSeconds": 60,
        "endTime": "Dec 18, 2019 7:28:46 PM",
        "serviceType": "RDS"
    },
    "responseElements": null,
    "requestID": "9ffbe15c-96b5-4fe6-bed9-9fccff1a0525",
    "eventID": "08908de0-2431-4e2e-ba7b-f5424f908433",
    "eventType": "AwsApiCall",
    "recipientAccountId": "123456789012"
}
```

性能详情 API 和接口 VPC 端点（Amazon PrivateLink）

可以使用 Amazon PrivateLink 在 VPC 和 Amazon RDS 性能详情之间创建私有连接。可以像在 VPC 中一样访问性能详情，而无需使用互联网网关、NAT 设备、VPN 连接或 Amazon Direct Connect 连接。VPC 中的实例不需要公有 IP 地址即可访问性能详情。

您可以通过创建由 Amazon PrivateLink 提供支持的接口端点来建立此私有连接。我们将在您为接口端点启用的每个子网中创建一个端点网络接口。这些是请求者托管的网络接口，用作发往性能详情的流量的入口点。

有关更多信息，请参阅《Amazon PrivateLink 指南》中的通过 Amazon PrivateLink 访问 Amazon Web Services 服务。

性能详情的注意事项

在为性能详情设置接口端点之前，请首先查看《Amazon PrivateLink 指南》中的 Considerations。

性能详情支持通过接口端点调用其所有 API 操作。

默认情况下，支持通过接口端点对性能详情进行完全访问。要控制通过接口端点流向性能详情的流量，请将安全组与端点网络接口关联。

可用性

性能详情 API 目前在支持性能详情的 Amazon Web Services 区域中支持 VPC 端点。有关性能详情可用性的信息，请参阅支持 Amazon RDS 中 Performance Insights 的区域和数据库引擎。

为性能详情创建接口端点

可以使用 Amazon VPC 控制台或 Amazon Command Line Interface（Amazon CLI）为性能详情创建接口端点。有关更多信息，请参阅《Amazon PrivateLink 指南》中的创建接口端点。

使用以下服务名称为性能详情创建接口端点：

如果为接口端点启用私有 DNS，则可使用其默认区域 DNS 名称向性能详情发出 API 请求。例如，`pi.us-east-1.amazonaws.com`。

为性能详情 API 创建 VPC 端点策略

端点策略是一种 IAM 资源，您可以将其附加到接口端点。默认端点策略支持通过接口端点完全访问性能详情。要控制支持从 VPC 访问性能详情的权限，请将自定义端点策略附加到接口端点。

端点策略指定以下信息：

- 可执行操作的主体（Amazon Web Services 账户、IAM 用户和 IAM 角色）。
- 可执行的操作。

- 可对其执行操作的资源。

有关更多信息，请参阅《Amazon PrivateLink 指南》中的使用端点策略控制对服务的访问权限。

示例：性能详情操作的 VPC 端点策略

以下是自定义端点策略的示例。将此策略附加到接口端点时，该策略会向所有主体授予对所有资源执行所列出的性能详情操作的访问权限。

```
{
   "Statement":[
      {
         "Principal":"*",
         "Effect":"Allow",
         "Action":[
            "rds:CreatePerformanceAnalysisReport",
            "rds:DeletePerformanceAnalysisReport",
            "rds:GetPerformanceAnalysisReport"
         ],
         "Resource":"*"
      }
   ]
}
```

示例：拒绝来自指定 Amazon 账户的所有访问的 VPC 终端节点策略

以下 VPC 端点策略会拒绝 Amazon 账户 123456789012 所有使用端点访问资源的权限。此策略允许来自其他账户的所有操作。

```
{
  "Statement": [
    {
      "Action": "*",
      "Effect": "Allow",
      "Resource": "*",
      "Principal": "*"
    },
    {
      "Action": "*",
      "Effect": "Deny",
      "Resource": "*",
      "Principal": { "AWS": [ "123456789012" ] }
```

```
        }
    ]
}
```

性能详情的 IP 寻址

IP 地址使 VPC 中的资源能够相互通信以及与 Internet 上的资源进行通信。性能详情同时支持 IPv4 和 IPv6 寻址协议。默认情况下，性能详情和 Amazon VPC 使用 IPv4 寻址协议。您无法关闭这种行为。创建 VPC 时，请确保指定 IPv4 CIDR 块 (一系列私有 IPv4 地址)。

可以选择将 IPv6 CIDR 块分配给 VPC 和子网，并将来自该块的 IPv6 地址分配给子网中的 RDS 资源。对 IPv6 协议的支持扩展了支持的 IP 地址数量。通过使用 IPv6 协议，您可以确保有足够的可用地址来应对 Internet 的未来发展。新的和现有 RDS 资源可以在 VPC 内使用 IPv4 和 IPv6 地址。在应用程序不同部分使用的两个协议之间配置、保护和转换网络流量可能会产生运营开销。您可以对 Amazon RDS 资源的 IPv6 协议进行标准化，以简化网络配置。有关服务端点和配额的更多信息，请参阅 Amazon Relational Database Service endpoints and quotas。

有关 Amazon RDS IP 寻址的更多信息，请参阅 Amazon RDS IP 寻址。

使用适用于 Amazon RDS 的 Amazon DevOps Guru 分析性能异常

Amazon DevOps Guru 是一项完全托管式运营服务，可帮助开发人员和运营商提高应用程序的性能和可用性。DevOps Guru 卸下了与识别运营问题相关的任务，以便您可以快速实施改进应用程序的建议。有关更多信息，请参阅《Amazon DevOps Guru 用户指南》中的什么是 Amazon DevOps Guru？。

DevOps Guru 可检测、分析所有 Amazon RDS 数据库引擎的现有操作问题并提出建议。DevOps Guru for RDS 通过将机器学习应用于 RDS for PostgreSQL 数据库的性能详情指标，扩展了此功能。这些监控功能使得适用于 RDS 的 DevOps Guru 可以检测和诊断性能瓶颈，并建议具体的纠正措施。DevOps Guru for RDS 还可以在您的 RDS for PostgreSQL 数据库中出现问题状况之前检测出这些状况。

现在，您可以在 RDS 控制台中查看这些建议。有关更多信息，请参阅 来自 Amazon RDS 的建议。

以下视频是关于 DevOps Guru for RDS 的概览。

有关此主题的深入分析，请参阅 Amazon DevOps Guru for RDS 深入剖析。

主题

DevOps Guru for RDS 的优势

如果是您负责 RDS for PostgreSQL 数据库，则您可能不知道正在发生影响该数据库的事件或回归。当您了解这个问题时，您可能不知道为什么会发生这个问题，也不知道该怎么处理它。您可以遵循 DevOps Guru for RDS 的建议，而不是向数据库管理员 (DBA) 寻求帮助或依赖第三方工具。

从 DevOps Guru for RDS 的详细分析中，您可以获得以下优势：

快速诊断

> DevOps Guru for RDS 会持续监控和分析数据库遥测。性能详情、增强监控和 Amazon CloudWatch 会收集数据库实例的遥测数据。DevOps Guru for RDS 会使用统计和机器学习技术来挖掘这些数据并检测异常情况。要了解有关遥测数据的更多信息，请参阅《Amazon RDS 用户指南》中的在 Amazon RDS 上使用性能详情监控数据库负载和使用增强监控来监控操作系统指标。

快速解决方案

每个异常情况都会识别性能问题，并建议调查或纠正措施的途径。例如，DevOps Guru for RDS 可能会建议您调查特定的等待事件。或者，它可能建议您优化应用程序池设置以限制数据库连接的数量。根据这些建议，您可以比手动进行故障排除更快地解决性能问题。

主动见解

DevOps Guru for RDS 使用资源中的指标来检测潜在的问题行为，以免其成为更大的问题。例如，此服务可以检测数据库何时使用越来越多的磁盘临时表，这可能会开始影响性能。然后，DevOps Guru 会提供建议以帮助解决问题，从而防止问题变成更大的问题。

深入了解 Amazon 工程师和机器学习

为了检测性能问题并帮助您解决瓶颈，DevOps Guru for RDS 依赖于机器学习（ML）和高级数据公式。Amazon 数据库工程师为 DevOps Guru for RDS 结果的开发做出了贡献，该服务封装了多年来管理成千上万个数据库的经验。通过利用这些集体知识，适用于 RDS 的 DevOps Guru 可以教您最佳实践。

适用于 RDS 的 DevOps Guru 的工作原理

DevOps Guru for RDS 从 Amazon RDS 性能详情中收集有关 RDS for PostgreSQL 数据库的数据。最重要的指标是 `DBLoad`。适用于 RDS 的 DevOps Guru 使用性能详情指标，使用机器学习对其进行分析，并将洞察发布到控制面板。

洞察是 DevOps Guru 检测到的相关异常的集合。

在 DevOps Guru for RDS 中，异常是一种与 RDS for PostgreSQL 数据库的正常性能有所偏差的模式。

主动见解

主动见解可以让您在问题发生之前了解问题行为。它包含异常情况以及建议和相关指标，可以帮助您解决 RDS for PostgreSQL 数据库中的问题，以免问题变得更严重。这些见解发布在 DevOps Guru 控制面板上。

例如，DevOps Guru 可能会检测到 RDS for PostgreSQL 数据库正在创建许多磁盘上的临时表。如果不加以解决，这种趋势可能会导致性能问题。每项主动见解都包含有关纠正行为的建议以及指向使用 Amazon DevOps Guru 主动见解优化 RDS for PostgreSQL中相关主题的链接。有关更多信息，请参阅《Amazon DevOps Guru 用户指南》中的在 DevOps Guru 中使用见解。

被动见解

被动见解可在异常行为发生时识别此类行为。如果 DevOps Guru for RDS 发现您的 RDS for PostgreSQL 数据库实例中的性能问题，它会在 DevOps Guru 控制面板中发布被动见解。有关更多信息，请参阅《Amazon DevOps Guru 用户指南》中的在 DevOps Guru 中使用见解。

因果异常

因果异常是被动见解内的一项顶级异常。数据库加载（数据库加载）是适用于 RDS 的 DevOps Guru 的因果异常。

异常情况通过分配严重性级别 High（高）、Medium（中）或 Low（低）来衡量性能影响。要了解详情，请参阅《Amazon DevOps Guru 用户指南》中的适用于 RDS 的 DevOps Guru 的关键概念。

如果 DevOps Guru 检测到数据库实例上的当前异常情况，则会在 RDS 控制台的 Databases（数据库）页面中收到提示。控制台还会提示您注意过去 24 小时内发生的异常。要从 RDS 控制台转到异常页面，请选择警报消息中的链接。RDS 控制台还会在 RDS for PostgreSQL 数据库实例的页面中向您发出提示。

上下文异常

上下文异常是数据库负载（DB 负载）内与被动见解相关的一项调查结果。每个上下文异常都描述了需要调查的特定 RDS for PostgreSQL 性能问题。例如，适用于 RDS 的 DevOps Guru 可能会建议您考虑增加 CPU 容量或调查导致数据库负载的等待事件。

> **Important**
>
> 我们建议您在修改生产实例之前在测试实例上测试所有更改。通过这种方式，您可以了解更改的影响。

要了解详情，请参阅《Amazon DevOps Guru 用户指南》中的分析 Amazon RDS 中的异常。

设置适用于 RDS 的 DevOps Guru

要允许 DevOps Guru for Amazon RDS 发布 RDS for PostgreSQL 数据库的见解，请完成以下任务。

主题

配置适用于 RDS 的 DevOps Guru 的 IAM 访问策略

要在 RDS 控制台中查看来自 DevOps Guru 的提示，您的 Amazon Identity and Access Management (IAM) 用户或角色必须具有以下任一策略：

- Amazon 托管策略 `AmazonDevOpsGuruConsoleFullAccess`
- Amazon 托管式策略 `AmazonDevOpsGuruConsoleReadOnlyAccess` 以及以下任一策略：
  - Amazon 托管策略 `AmazonRDSFullAccess`
  - 客户托管策略包括 `pi:GetResourceMetrics` 和 `pi:DescribeDimensionKeys`

有关更多信息，请参阅 为 Performance Insights 配置访问策略。

为您的 RDS for PostgreSQL 数据库实例开启性能详情

适用于 RDS 的 DevOps Guru 依靠性能详情来获取其数据。如果没有性能详情，DevOps Guru 会发布异常情况，但不包括详细的分析和建议。

在创建或修改 RDS for PostgreSQL 数据库实例时，可以开启性能详情。有关更多信息，请参阅 为 Amazon RDS 打开和关闭性能详情。

打开 DevOps Guru 并指定资源覆盖范围

您可以开启 DevOps Guru，以让它通过以下任一方式监控您的 RDS for PostgreSQL 数据库。

主题

在 RDS 控制台中开启 DevOps Guru

您可以在 Amazon RDS 控制台中通过多个途径来开启 DevOps Guru。

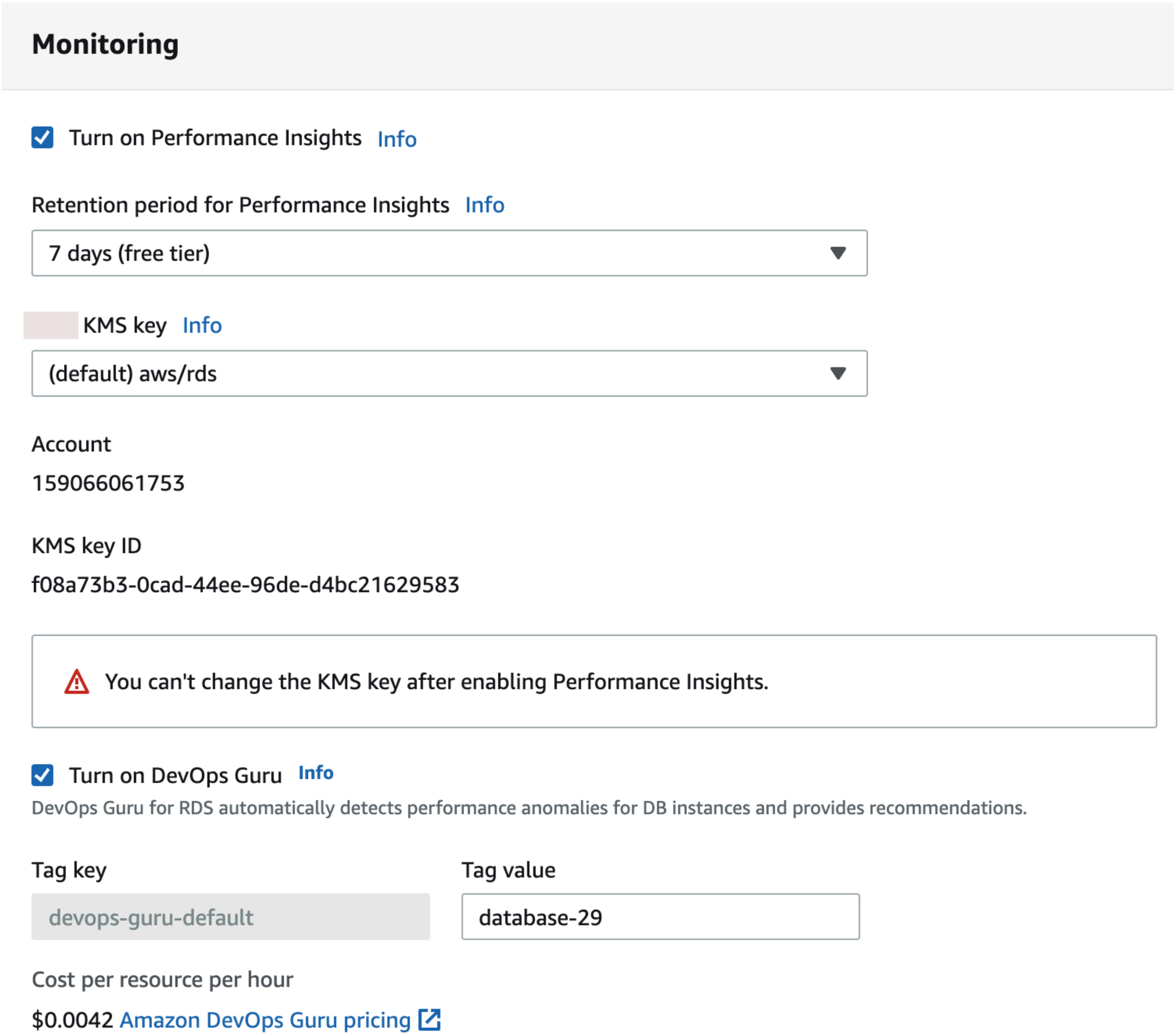
主题

创建 RDS for PostgreSQL 数据库时开启 DevOps Guru

创建工作流包括一个设置，该设置可为数据库开启 DevOps Guru 覆盖范围。原定设置情况下，当您选择 Production（生产）模板时，此设置处于开启状态。

创建 RDS for PostgreSQL 数据库时开启 DevOps Guru

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 按创建数据库实例中的步骤操作，直至（但不包括）您选择监控设置的步骤。
3. 在 Monitoring（监控）中，选择 Turn on Performance Insights（开启性能详情）。为让 DevOps Guru for RDS 提供对性能异常情况的详细分析，必须开启性能详情。
4. 选择 Turn on DevOps Guru（开启 DevOps Guru）。



5. 为数据库创建一个标签，以便 DevOps Guru 可以对其进行监控。执行以下操作：

   - 在 Tag key（标签键）的文本字段中，输入以 **Devops-Guru-** 开头的名称。
   - 在 Tag key（标签键）的文本字段中，输入任意值。例如，如果输入 **rds-database-1** 作为 RDS for PostgreSQL 数据库的名称，还可以输入 **rds-database-1** 作为标签值。

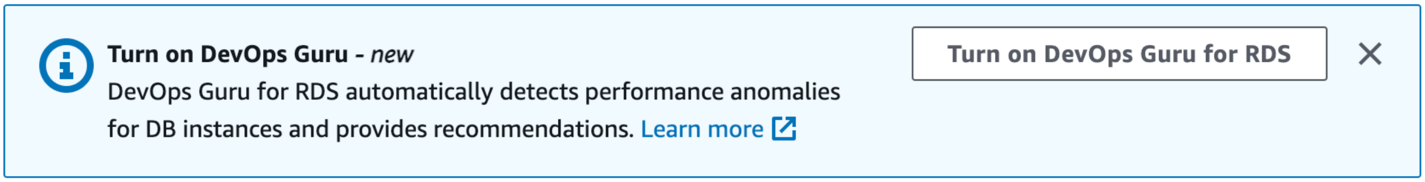
   有关标签的更多信息，请参阅《Amazon DevOps Guru 用户指南》中的“[使用标签识别 DevOps Guru 应用程序中的资源](#)”。

6. 完成[创建数据库实例](#) 中的其余步骤。

从通知横幅开启 DevOps Guru

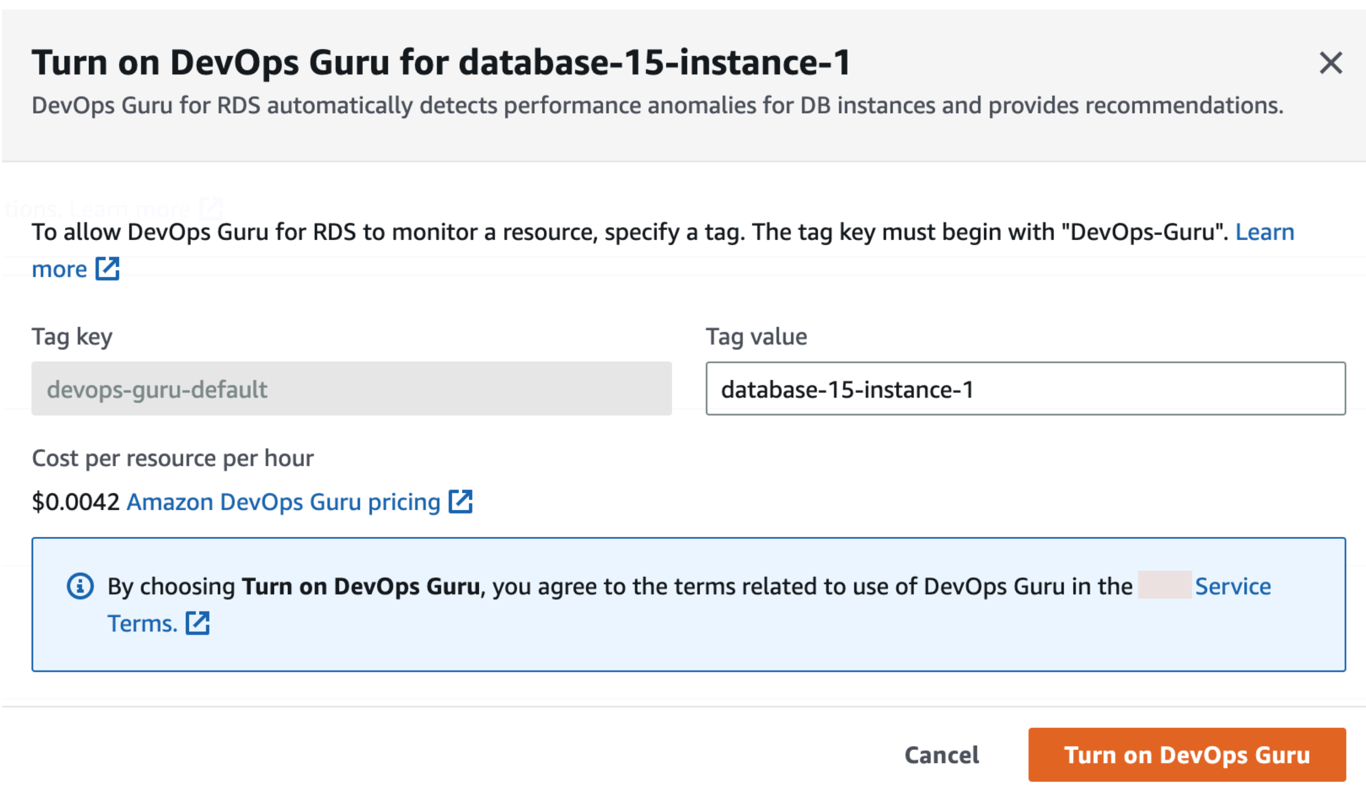
如果 DevOps Guru 未涵盖您的资源，Amazon RDS 会在以下位置通过横幅通知您：

- 数据库集群实例的 Monitoring（监控）选项卡
- 性能详情控制面板



为 RDS for PostgreSQL 数据库开启 DevOps Guru

1. 在横幅中，选择 Turn on DevOps Guru for RDS（开启适用于 RDS 的 DevOps Guru）。
2. 输入标签键名称和值。有关标签的更多信息，请参阅《Amazon DevOps Guru 用户指南》中的[“使用标签识别 DevOps Guru 应用程序中的资源”]。



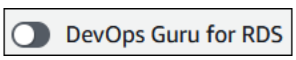
3. 选择 Turn on DevOps Guru（开启 DevOps Guru）。

响应开启 DevOps Guru 时的权限错误

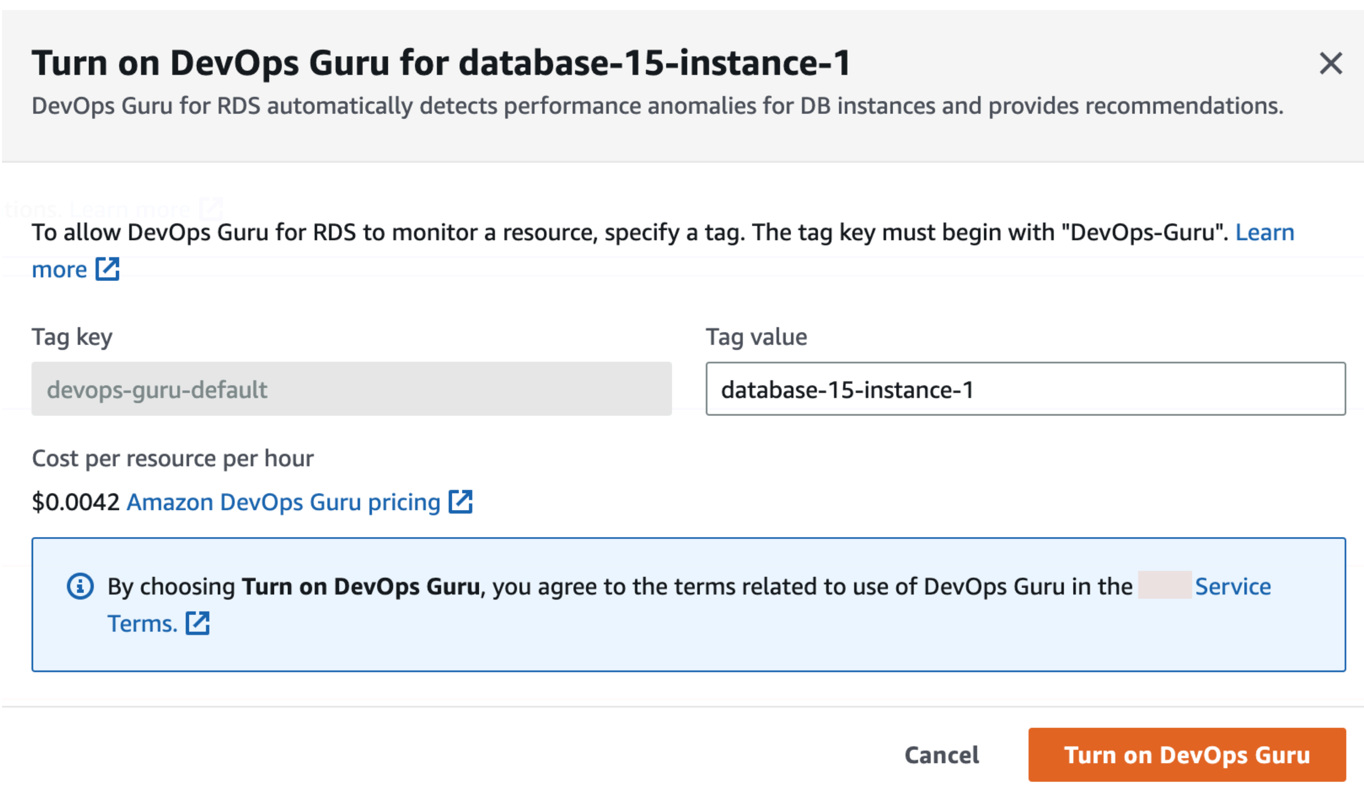
如果您在创建数据库时从 RDS 控制台开启 DevOps Guru，RDS 可能会显示以下关于缺少权限的横幅。

响应权限错误

1. 向您的 IAM 用户或角色授予用户托管式角色 `AmazonDevOpsGuruConsoleFullAccess`。有关更多信息，请参阅 配置适用于 RDS 的 DevOps Guru 的 IAM 访问策略。
2. 打开 RDS 控制台。
3. 在导航窗格中，选择 Performance Insights。
4. 在刚创建的集群中选择一个数据库实例。
5. 选择此开关以开启 DevOps Guru for RDS。



6. 选择标签值。有关更多信息，请参阅《Amazon DevOps Guru 用户指南》中的“使用标签识别 DevOps Guru 应用程序中的资源”。



7. 选择 Turn on DevOps Guru（开启 DevOps Guru）。

在 DevOps Guru 控制台中添加 RDS for PostgreSQL 资源

您可以在 DevOps Guru 控制台上指定 DevOps Guru 资源覆盖范围。按照《Amazon DevOps Guru 用户指南》中的指定 DevOps Guru 资源覆盖范围中描述的步骤操作。编辑所分析的资源时，请选择以下选项之一：

- 选择所有账户资源以分析所有受支持的资源，包括您的 Amazon Web Services 账户和区域中的 RDS for PostgreSQL 数据库。
- 选择 CloudFormation 堆栈来分析您选择的堆栈中的 RDS for PostgreSQL 数据库。有关更多信息，请参阅《Amazon DevOps Guru 用户指南》中的使用 Amazon CloudFormation 堆栈识别 DevOps Guru 应用程序中的资源。
- 选择标签来分析您已标记的 RDS for PostgreSQL 数据库。有关更多信息，请参阅《Amazon DevOps Guru 用户指南》中的使用标签识别 DevOps Guru 应用程序中的资源。

有关更多信息，请参阅《Amazon DevOps Guru 用户指南》中的启用 DevOps Guru。

使用 Amazon CloudFormation 添加 RDS for PostgreSQL 资源

可以使用标签向您的 CloudFormation 模板添加 RDS for PostgreSQL 资源的覆盖范围。以下过程假设您具有同时适用于 RDS for PostgreSQL 数据库实例和 DevOps Guru 堆栈的 CloudFormation 模板。

使用 CloudFormation 标签指定 RDS for PostgreSQL 数据库实例

1. 在数据库实例的 CloudFormation 模板中，使用键/值对定义标签。

   以下示例将 RDS for PostgreSQL 数据库实例的值 `my-db-instance1` 分配给 `Devops-guru-cfn-default`。

   ```
   MyDBInstance1:
     Type: "AWS::RDS::DBInstance"
     Properties:
       DBInstanceIdentifier: my-db-instance1
       Tags:
         - Key: Devops-guru-cfn-default
           Value: devopsguru-my-db-instance1
   ```

2. 在 DevOps Guru 堆栈的 CloudFormation 模板中，在资源收集筛选条件中指定相同的标签。

以下示例将 DevOps Guru 配置为使用标签值 `my-db-instance1` 为资源提供覆盖范围。

```
DevOpsGuruResourceCollection:
  Type: AWS::DevOpsGuru::ResourceCollection
  Properties:
    ResourceCollectionFilter:
      Tags:
        - AppBoundaryKey: "Devops-guru-cfn-default"
          TagValues:
          - "devopsguru-my-db-instance1"
```

以下示例涵盖了应用程序边界 `Devops-guru-cfn-default` 内的所有资源。

```
DevOpsGuruResourceCollection:
  Type: AWS::DevOpsGuru::ResourceCollection
  Properties:
    ResourceCollectionFilter:
      Tags:
        - AppBoundaryKey: "Devops-guru-cfn-default"
          TagValues:
          - "*"
```

有关更多信息，请参阅《Amazon CloudFormation 用户指南》中的 AWS::DevOpsGuru::ResourceCollection 和 AWS::RDS::DBInstance。

使用增强监控来监控操作系统指标

利用增强监测功能，您可以实时监控数据库实例的操作系统。若您想了解不同进程或线程对 CPU 的使用差异，增强监测指标非常有用。

主题

增强监测概述

Amazon RDS 为数据库实例运行的操作系统 (OS) 实时提供指标。您可以在控制台上查看 RDS 数据库实例的所有系统指标和过程信息。您可以管理要为每个实例监控哪些指标，并根据您的要求自定义控制面板。有关增强监控指标的说明，请参阅 增强监控中的操作系统指标。

RDS 将增强监测中的指标传输到您的 Amazon CloudWatch Logs 账户。您可以在 CloudWatch Logs 中创建指标筛选条件，并在 CloudWatch 控制面板上显示图表。此外，您可以在选择的监控系统中通过 Amazon CloudWatch Logs 使用增强监测 JSON 输出。有关更多信息，请参阅 Amazon RDS 常见问题中的增强监测。

主题

增强监测可用性

增强监测对以下数据库引擎可用：

- Db2
- MariaDB
- Microsoft SQL Server

- MySQL
- Oracle
- PostgreSQL

CloudWatch 与增强监控指标的区别

虚管理程序创建并运行虚拟机 (VM)。借助管理程序，实例可以通过虚拟共享内存和 CPU 支持多个来宾虚拟机。CloudWatch 从数据库实例管理程序收集关于 CPU 使用率的指标。而增强监测则从数据库实例上的代理收集这一指标。

您可能会发现 CloudWatch 和增强监测测量值之间存在差异，因为管理程序层执行的工作较少。如果数据库实例使用较小的实例类，差异可能会更大。在这种情况下，单个物理实例上的管理程序层可能会管理更多的虚拟机 (VM)。

有关增强监控指标的说明，请参阅 增强监控中的操作系统指标。有关 Amazon CloudWatch 的更多信息，请参阅 Amazon CloudWatch 用户指南。

保留增强监测指标

默认情况下，增强监测指标可在 CloudWatch Logs 中存储达 30 天。此保留期与典型的 CloudWatch 指标不同。

要修改指标存储在 CloudWatch Logs 中的时间量，请在 CloudWatch 控制台中更改 `RDSOSMetrics` 日志组的保留期。有关更多信息，请参阅 Amazon CloudWatch Logs User Guide 中的更改 CloudWatch Logs 中的日志数据保留期。

增强监测的成本

增强监测指标存储在 CloudWatch Logs 中，而不是存储在 CloudWatch 指标中。增强监测的成本取决于以下因素：

- 您需要为 Amazon CloudWatch Logs 提供的免费套餐之外的增强监测付费。收费是根据 CloudWatch Logs 数据传输和存储费率计算的。
- RDS 实例传输的信息量与为增强型监测功能定义的精细程度成正比。监控间隔越短，操作系统指标报告频率越高，监控成本也就越高。要管理成本，请为账户中的不同实例设置不同的精细度。
- 增强监测的使用成本适用于启用了增强监测的每个数据库实例。监控大量数据库实例的成本要高于监控少量数据库实例。
- 数据库实例支持的工作负载计算越密集，要报告的操作系统进程活动就越多，增强监测的成本也越高。

设置和启用增强监控

要使用增强监控，您必须创建 IAM 角色，然后启用增强监控。

主题

- 为增强监控创建 IAM 角色
- 打开和关闭增强监控
- 防范混淆代理问题

为增强监控创建 IAM 角色

增强监测需要代表您执行操作的权限来向 CloudWatch Logs发送操作系统指标信息。您使用 Amazon Identity and Access Management（IAM）角色授予增强监控权限。您可以在启用增强监控时创建此角色，也可以事先创建该角色。

主题

- 在启用增强监控时创建 IAM 角色
- 在启用增强监控之前创建 IAM 角色

在启用增强监控时创建 IAM 角色

在 RDS 控制台中启用增强监控后，Amazon RDS 可以为您创建所需的 IAM 角色。该角色命名为 `rds-monitoring-role`。RDS 将此角色用于指定的数据库实例、只读副本或多可用区数据库集群。

在启用增强监控时创建 IAM 角色的方法

1. 按打开和关闭增强监控中的步骤操作。
2. 在选择角色的步骤中，将监视角色设置为默认值。

在启用增强监控之前创建 IAM 角色

您可以在启用增强监控之前创建所需的角色。在启用增强监控时，请指定新角色的名称。如果使用 Amazon CLI 或 RDS API 启用增强监测，则必须创建此必需角色。

必须向启用增强监控的用户授予 `PassRole` 权限。有关更多信息，请参阅 IAM 用户指南的授予向 Amazon 服务传递角色的用户权限中的示例 2。

为 Amazon RDS 增强监控创建 IAM 角色

1. 通过以下网址打开 IAM 控制台：https://console.aws.amazon.com。
2. 在导航窗格中，选择 Roles (角色)。
3. 选择创建角色。
4. 选择 Amazon service（Amazon Web Services 服务）选项卡，然后从服务列表中选择 RDS。
5. 选择 RDS - Enhanced Monitoring（RDS - 增强监测），然后选择 Next（下一步）。
6. 确保 Permissions policies（权限策略）显示 AmazonRDSEnhancedMonitoringRole，然后选择 Next（下一步）。
7. 对于角色名称，请为您的角色输入一个名称。例如，输入 **`emaccess`**。

   您角色的可信实体是 Amazon 服务 monitoring.rds.amazonaws.com。
8. 选择创建角色。

打开和关闭增强监控

可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 管理增强监控。可以为每个数据库实例上的指标收集设置不同的粒度。

控制台

您可以在创建数据库实例、多可用区数据库集群或只读副本，或者在修改数据库实例或多可用区数据库集群时打开增强监控。如果修改数据库实例以开启增强监控，则不需要重启数据库实例，更改也会生效。

在 Databases（数据库）页面执行以下某种操作时，您可以在 RDS 控制台中打开增强监控：

- 创建数据库实例或多可用区数据库集群 - 选择 Create database（创建数据库）。
- Create a read replica（创建只读副本）：选择 Actions（操作），然后选择 Create read replica（创建只读副本）。
- 修改数据库实例或多可用区数据库集群：选择修改。

在 RDS 控制台中打开或关闭增强监控

1. 滚动到 Additional configuration（其他配置）。
2. 在监控中，为数据库实例或只读副本选择启用增强监控。取消选择该选项可。

3. 将 Monitoring Role 属性设置为您创建的 IAM 角色以允许 Amazon RDS 代表您与 Amazon CloudWatch Logs 通信，或选择 Default 让 RDS 为您创建一个名为 `rds-monitoring-role` 的角色。
4. 将粒度属性设置成两次为数据库实例或只读副本收集指标之间的间隔，以秒为单位。Granularity 属性可以设置为以下值之一：`1`、`5`、`10`、`15`、`30` 或 `60`。

   RDS 控制台最快每 5 秒刷新一次。如果您在 RDS 控制台中将粒度设置为 1 秒，仍然会看到指标每 5 秒更新一次。使用 CloudWatch Logs 可以获得 1 秒的指标更新。

Amazon CLI

要使用 Amazon CLI 打开增强监控，请在以下命令中将 `--monitoring-interval` 选项设置为 `0` 以外的值，并将 `--monitoring-role-arn` 选项设置为您在 为增强监控创建 IAM 角色 中创建的角色。

- create-db-instance
- create-db-instance-read-replica
- modify-db-instance
- create-db-cluster（多可用区数据库集群）
- modify-db-cluster（多可用区数据库集群）

`--monitoring-interval` 选项指定收集增强监控指标的时间点之间的间隔，以秒为单位。选项的有效值为 `0`、`1`、`5`、`10`、`15`、`30`、和 `60`。

要使用 Amazon CLI 关闭增强监控，请在这些命令中将 `--monitoring-interval` 选项设置为 `0`。

Example

以下示例将打开数据库实例的增强监控：

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier mydbinstance \
    --monitoring-interval 30 \
    --monitoring-role-arn arn:aws:iam::123456789012:role/emaccess
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier mydbinstance ^
    --monitoring-interval 30 ^
    --monitoring-role-arn arn:aws:iam::123456789012:role/emaccess
```

Example

以下示例将打开多可用区数据库集群的增强监控：

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-cluster \
    --db-cluster-identifier mydbcluster \
    --monitoring-interval 30 \
    --monitoring-role-arn arn:aws:iam::123456789012:role/emaccess
```

对于 Windows：

```
aws rds modify-db-cluster ^
    --db-cluster-identifier mydbcluster ^
    --monitoring-interval 30 ^
    --monitoring-role-arn arn:aws:iam::123456789012:role/emaccess
```

RDS API

要使用 RDS API 打开增强监控，请将 `MonitoringInterval` 参数设置为 `0` 以外的值，并将 `MonitoringRoleArn` 参数设置为您在 为增强监控创建 IAM 角色 中创建的角色。在以下操作中设置这些参数：

- CreateDBInstance
- CreateDBInstanceReadReplica
- ModifyDBInstance
- CreateDBCluster（多可用区数据库集群）
- ModifyDBCluster（多可用区数据库集群）

`MonitoringInterval` 参数指定收集增强监控指标的时间点之间的间隔，以秒为单位。有效值为 `0`、`1`、`5`、`10`、`15`、`30` 和 `60`。

要使用 RDS API 关闭增强监控，请将 `MonitoringInterval` 设置为 `0`。

防范混淆代理问题

混淆代理问题是一个安全性问题，即不具有某操作执行权限的实体可能会迫使具有更高权限的实体执行该操作。在 Amazon 中，跨服务模拟可能会导致混淆代理问题。一个服务（呼叫服务）调用另一项服务（所谓的服务）时，可能会发生跨服务模拟。可以操纵调用服务，使用其权限以在其他情况下该服务不应有访问权限的方式对另一个客户的资源进行操作。为防止这种情况，Amazon 提供可帮助您保护所有服务的数据的工具，而这些服务中的服务主体有权限访问账户中的资源。有关更多信息，请参阅混淆代理问题。

要限制 Amazon RDS 授予另一项服务对资源的访问权限，建议在增强监控角色的信任策略中使用 `aws:SourceArn` 和 `aws:SourceAccount` 全局条件上下文键。如果使用两个全局条件上下文键，则这两个键必须使用相同的账户 ID。

防范混淆代理问题最有效的方法是使用 `aws:SourceArn` 全局条件上下文键和资源的完整 ARN。对于 Amazon RDS，请将 `aws:SourceArn` 设置为 `arn:aws:rds:`*`Region`*`:`*`my-account-id`*`:db:`*`dbname`*。

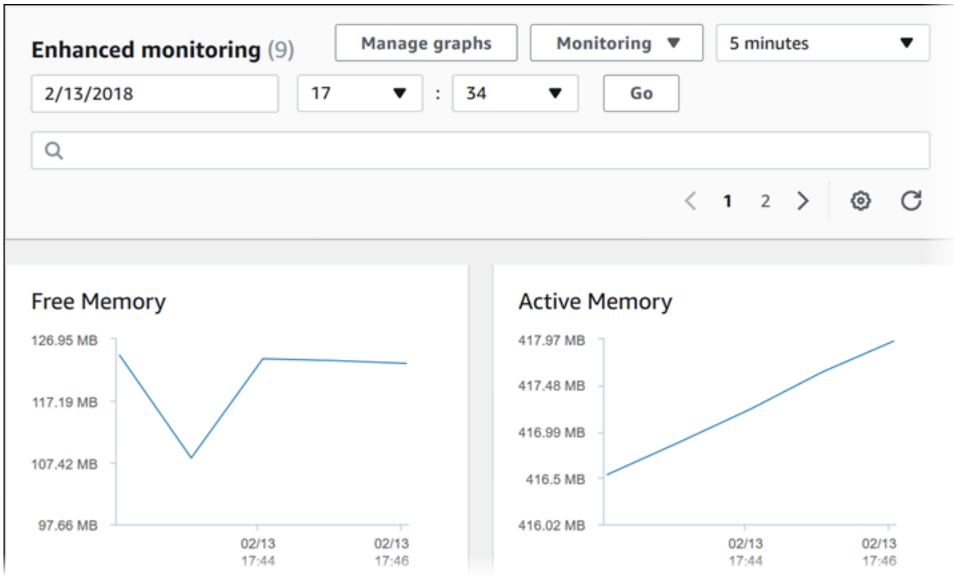
以下示例在信任策略中使用 `aws:SourceArn` 和 `aws:SourceAccount` 全局条件上下文键来防范混淆代理问题。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "monitoring.rds.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringLike": {
          "aws:SourceArn": "arn:aws:rds:Region:my-account-id:db:dbname"
        },
        "StringEquals": {
          "aws:SourceAccount": "my-account-id"
        }
      }
    }
  ]
}
```

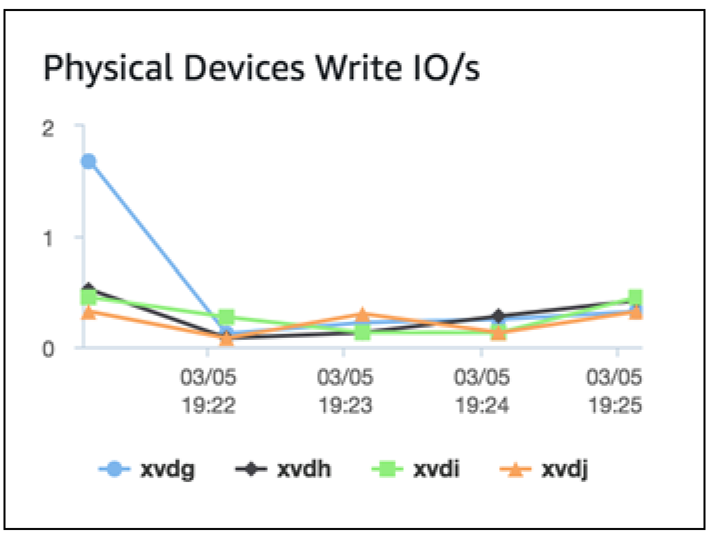
在 RDS 控制台中查看操作系统指标

您可以在 RDS 控制台中查看增强监测报告的操作系统指标，方法是为监测选择 Enhanced monitoring (增强监测)。

以下示例显示“Enhanced Monitoring（增强监控）”页面。有关增强监控指标的说明，请参阅 增强监控中的操作系统指标。

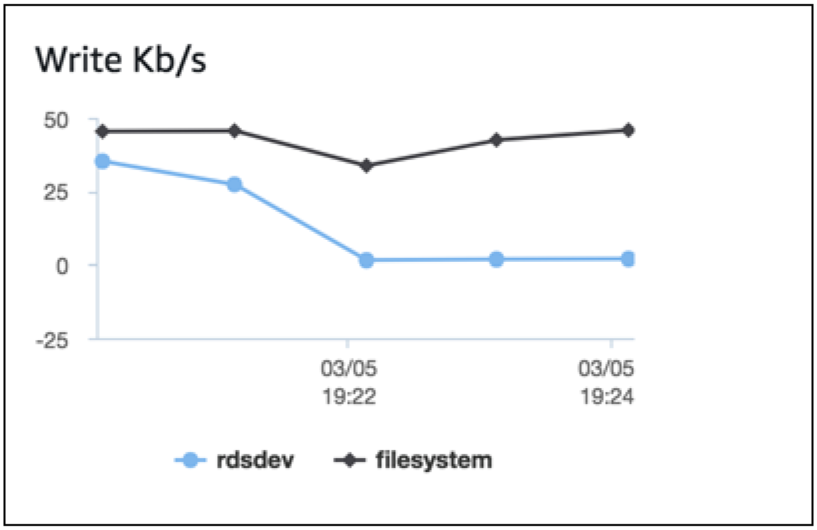


一些数据库实例为数据库实例的数据存储卷使用多个磁盘。在这些数据库实例上，Physical Devices (物理设备) 图表显示每一个磁盘的指标。例如，下图显示四个磁盘的指标。

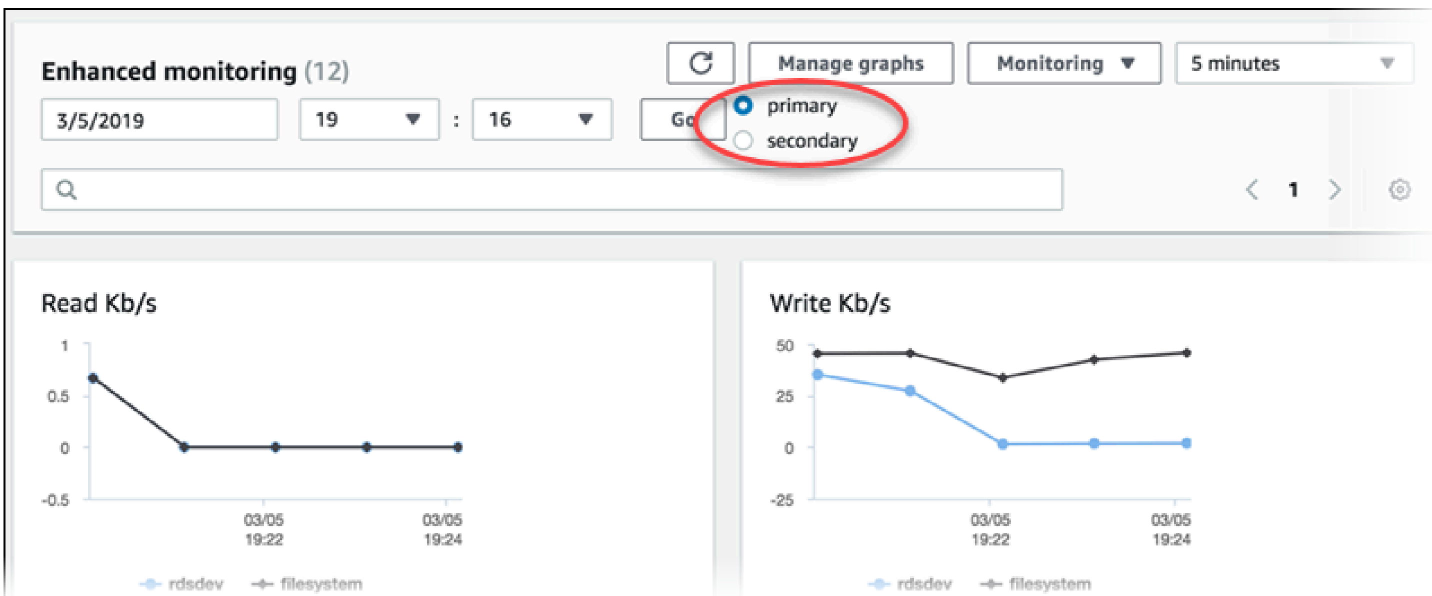


> **Note**
>
> 目前，Physical Devices (物理设备) 图表不可用于 Microsoft SQL Server 数据库实例。

在查看聚合的 Disk I/O（磁盘输入/输出）和 File system（文件系统）图表时，rdsdev 设备与 / `rdsdbdata` 文件系统相关联，后者存储会所有数据库文件和日志。filesystem 设备与 / 文件系统（又称为根）相关联，后者存储了与操作系统相关的文件。



如果数据库实例是多可用区部署，则您可以查看主数据库实例及其多可用区备用副本的操作系统指标。在 Enhanced monitoring (增强监测) 视图中，选择 primary (主) 来查看主数据库实例的操作系统指标，或选择 secondary (辅助) 来查看备用副本的操作系统指标。

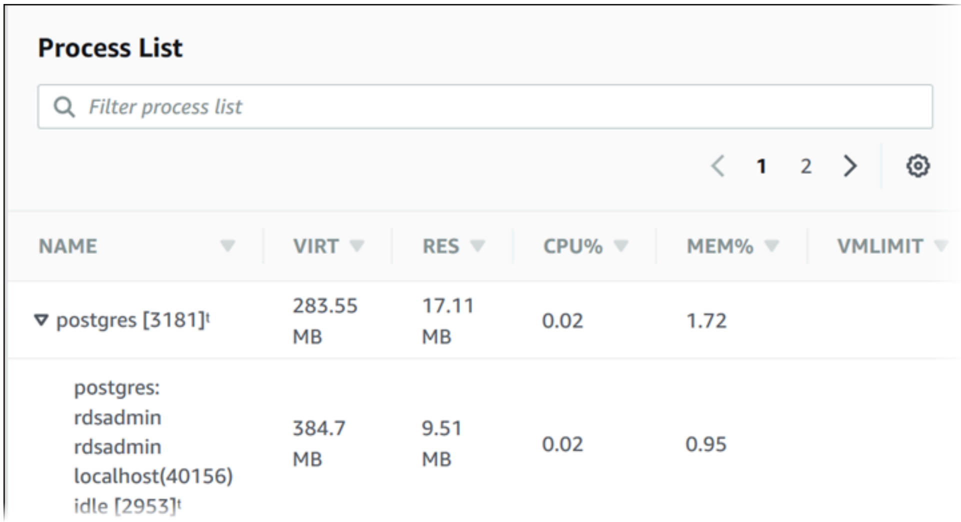


有关多可用区部署的更多信息，请参阅[配置和管理 Amazon RDS 的多可用区部署]。

> **Note**
>
> 目前，对于 MariaDB 数据库实例，不支持查看多可用区备用副本的操作系统指标。

如果要查看运行在数据库实例上的进程的详细信息，请为监测选择 OS 进程列表。

Process List (进程列表) 视图如下所示。



Process list (进程列表) 视图中所示的增强监测指标按以下方式组织：

- RDS child processes (RDS 子进程) – 显示支持数据库实例的 RDS 进程的摘要，例如 和 `mysqld`（对于 MySQL 数据库实例）。进程线程在父进程下嵌套显示。进程线程仅显示 CPU 使用率，因为进程所有线程的其他指标都相同。控制台最多显示 100 个进程和线程。因此，显示结果将是占用最多 CPU 和内存的进程和线程。如果有超过 50 个进程和超过 50 个线程，则控制台只会显示每种类别的前 50 个。这种显示方式有助于您了解哪些进程对性能影响最大。
- RDS processes (RDS 进程) – 摘要显示 RDS 管理代理所用的资源、诊断监控进程以及支持 RDS 数据库实例所需的其他Amazon进程。
- OS processes – 显示内核和系统进程摘要，这些进程通常对性能影响最小。

对每个进程列出的项目有：

- VIRT – 显示进程的虚拟大小。
- RES – 显示进程正在使用的实际物理内存。
- CPU% – 显示进程正在使用的总 CPU 带宽的百分比。
- MEM% – 显示进程使用的总内存的百分比。

RDS 控制台中显示的监控数据是从 Amazon CloudWatch Logs 中检索的。您还可以从 CloudWatch Logs 中检索数据库实例的指标作为日志流。有关更多信息，请参阅“使用 CloudWatch Logs 查看操作系统指标”。

以下情况下不会返回增强监测指标：

- 数据库实例发生了故障转移。
- 更改了数据库实例的实例类 (扩展计算)。

增强监测指标在重启数据库实例期间返回，因为只会重启数据库引擎。仍会报告操作系统的指标。

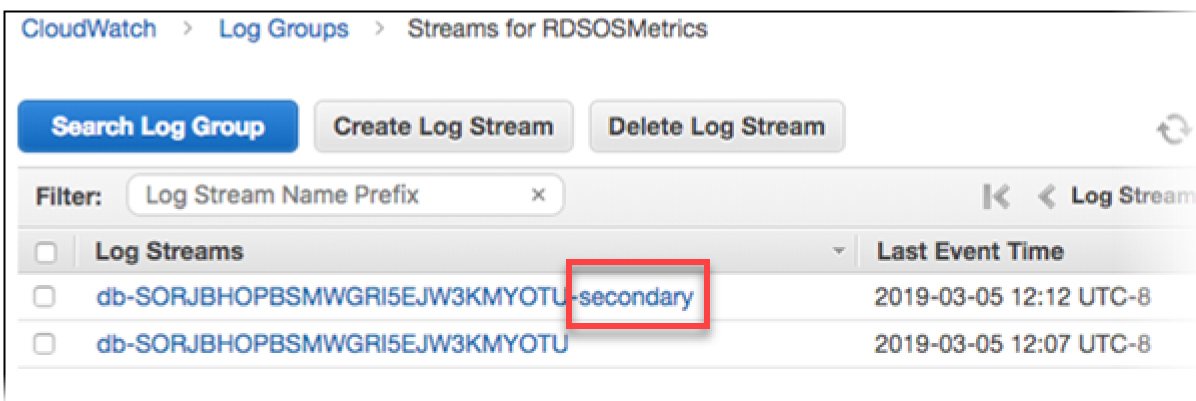
使用 CloudWatch Logs 查看操作系统指标

为数据库实例或多可用区数据库集群启用增强监控后，您可以使用 CloudWatch Logs 查看其指标，每个日志流表示一个被监控的数据库实例或数据库集群。日志流标识符是数据库实例或数据库集群的资源标识符 (`DbiResourceId`)。

查看增强监控日志数据

1. 访问 https://console.aws.amazon.com/cloudwatch/ 打开 CloudWatch 控制台。
2. 如有必要，请选择数据库实例或多可用区数据库集群所在的 Amazon Web Services 区域。有关更多信息，请参阅 Amazon Web Services 一般参考 中的区域和终端节点。
3. 在导航窗格中，选择 Logs (日志)。
4. 从日志组列表中选择 RDSOSMetrics。

   在多可用区数据库实例部署中，名称中附加了 `-secondary` 的日志文件针对多可用区备用副本。

5. 从日志流列表中选择要查看的日志流。

Amazon RDS 的指标参考

在此参考中，您可以找到 Amazon CloudWatch、Performance Insights 和增强监控的 Amazon RDS 指标的说明。

主题

Amazon RDS 的 Amazon CloudWatch 指标

Amazon CloudWatch 指标可让您深入了解 Amazon RDS 实例和集群的性能和运行状况，并使您能够监控系统行为并做出以数据为依据的决策。这些指标有助于跟踪资源利用率、数据库活动和操作效率，从而可以深入了解实例的性能。

本参考资料概述了可用于 Amazon RDS 的特定指标，并说明了如何解释和使用这些指标来优化数据库性能、排查问题和确保高可用性。

Amazon RDS 将指标发布到 Amazon CloudWatch 中的 `AWS/RDS` 和 `AWS/Usage` 命名空间。

主题

Amazon RDS 的 Amazon CloudWatch 实例级指标

Amazon CloudWatch 中的 `AWS/RDS` 命名空间包括以下实例级指标。

> **Note**
>
> Amazon RDS 控制台可能会以与发送到 Amazon CloudWatch 的单位不同的单位显示指标。例如，Amazon RDS 控制台可能会以兆字节 (MB) 为单位显示一个指标，同时以字节为单位将该指标发送到 Amazon CloudWatch。

指标	描述	适用于	单位
`BinLogDiskUsage`	二进制日志所占的磁盘空间大小。如果为 MySQL 和 MariaDB 实例（包括只读副本）启用了自动备份，则会创建二进制日志。	MariaDB MySQL	字节
`BurstBalance`	可用的通用型 SSD (GP2) 突增存储桶 I/O 点数的百分比。	全部	百分比
`CheckpointLag`	自最近一次检查点以来的时间。		秒
`ConnectionAttempts`	尝试连接实例的次数，无论成功与否。	MySQL	计数
`CPUUtilization`	CPU 使用百分率。	全部	百分比
`CPUCreditUsage`	实例为保持 CPU 使用率而花费的 CPU 积分数。一个 CPU 积分等于一个 vCPU 以 100% 的使用率运行一分钟或等同的 vCPU、使用率与时间的组合。例如，您可以有一个 vCPU 按 50% 使用率运行两分钟，或者两个 vCPU 按 25% 使用率运行两分钟。 此指标仅适用于 `db.t2`、`db.t3` 和 `db.t4g` 实例。		积分 (vCPU 分钟)

指标	描述	适用于	单位
	Note 建议仅将 T 数据库实例类用于开发和测试服务器，或其他非生产服务器。有关 T 实例类的更多详细信息，请参阅[数据库实例类类型](#) CPU 积分指标仅每 5 分钟提供一次。如果您指定一个大于五分钟的时间段，请使用`Sum` 统计数据，而非 `Average` 统计数据。		

指标	描述	适用于	单位
CPUCredit Balance	实例自启动后已累积获得的 CPU 积分数。对于 T2 标准，CPUCredit Balance 还包含已累积的启动积分数。 在获得积分后，积分将在积分余额中累积；在花费积分后，将从积分余额中扣除积分。积分余额具有最大值限制，这是由实例大小决定的。在达到限制后，将丢弃获得的任何新积分。对于 T2 标准，启动积分不计入限制。 实例可以花费 CPUCreditBalance 中的积分，以便突增到基准 CPU 使用率以上。 在实例运行过程中，CPUCredit Balance 中的积分不会过期。在实例停止时，CPUCreditBalance 不会保留，并且所有累积的积分都将丢失。 CPU 信用指标仅每 5 分钟提供一次。此指标仅适用于 db.t2、db.t3 和 db.t4g 实例。 **Note** 建议仅将 T 数据库实例类用于开发和测试服务器，或其他非生产服务器。有关 T 实例类的更多详细信息，请参阅数据库实例类类型 启动积分在 Amazon RDS 中的作用方式与在 Amazon EC2 中的作用方式相同。		积分（vCPU 分钟）

指标	描述	适用于	单位
	有关更多信息，请参阅[适用于 Linux 实例的 Amazon Elastic Compute Cloud 用户指南]()中的启动积分。		
`CPUSurplusCreditBalance`	在 `CPUCreditBalance` 值为零时，无限实例花费的超额积分数。 `CPUSurplusCreditBalance` 值由获得的 CPU 积分支付。如果超额积分数超出实例可在 24 小时周期内获得的最大积分数，则超出最大积分数的已花费超额积分将产生额外费用。 CPU 信用指标仅每 5 分钟提供一次。	全部	积分（vCPU 分钟）
`CPUSurplusCreditsCharged`	未由获得的 CPU 积分支付并且会产生额外费用的已花费超额积分数。 在出现以下任一情况时，将对花费的超额积分收费： • 花费的超额积分超出实例可在 24 小时周期内获得的最大积分数。对于超出最大积分数的所花费超额积分，将在该小时结束时向您收费。 • 实例已停止或终止。 • 实例从 `unlimited` 切换为 `standard`。 CPU 信用指标仅每 5 分钟提供一次。	全部	积分（vCPU 分钟）

指标	描述	适用于	单位
DatabaseConnections	连接至数据库实例的客户端网络连接数。 数据库会话数可能高于指标值，因为指标值不包括以下内容： • 不再具有网络连接但数据库尚未清理的会话 • 数据库引擎出于自身目的创建的会话 • 由数据库引擎的并行执行功能创建的会话 • 由数据库引擎任务计划程序创建的会话 • Amazon RDS 连接	全部	计数
DiskQueueDepth	等待访问磁盘的未完成 I/O（读取/写入请求）的数量。	全部	计数
DiskQueueDepthLogVolume	等待访问日志卷磁盘的未完成 I/O（读取/写入请求）的数量。	启用了专用日志卷的数据库实例	计数
EBSByteBalance%	RDS 数据库突增存储桶中剩余的吞吐量积分的百分比 此指标仅对基本监控可用。 该指标值基于包括根卷在内的所有卷的吞吐量，而不是仅基于那些包含数据库文件的卷。 要查找支持此指标的实例大小，请参阅《Amazon EC2 用户指南》中默认优化的 EBS 中带星号（*）的实例大小。Sum 统计数据不适用于该指标。	全部	百分比

指标	描述	适用于	单位
`EBSIOBalance%`	RDS 数据库突增存储桶中剩余的 I/O 积分的百分比 此指标仅对基本监控可用。 该指标值基于包括根卷在内的所有卷的 IOPS，而不是仅基于那些包含数据库文件的卷。 要查找支持此指标的实例大小，请参阅《Amazon EC2 用户指南》中的 Amazon EBS 优化的实例类型。Sum 统计数据不适用于该指标。 这个指标不同于 `BurstBalance` 。要了解如何使用此指标，请参阅利用 Amazon EBS 优化型实例突发功能提升应用程序性能并降低成本。	全部	百分比
`FailedSQLServerAgentJobsCount`	过去 1 分钟内失败的 Microsoft SQL Server Agent 作业的数量。	Microsoft SQL Server	每分钟计数
`FreeableMemory`	随机存取内存的可用大小。 对于 MariaDB、MySQL、Oracle 和 PostgreSQL 数据库实例，此指标报告 `MemAvailable` 的 `/proc/meminfo` 字段的值。	全部	字节

指标	描述	适用于	单位
FreeLocalStorage	可用本地存储空间的大小。 此指标仅适用于具有 NVMe SSD 实例存储卷的数据库实例类。有关具有 NVMe SSD 实例存储卷的 Amazon EC2 实例的信息，请参阅实例存储卷。等效的 RDS 数据库实例类具有相同的实例存储卷。例如，db.m6gd 和 db.r6gd 数据库实例类具有 NVMe SSD 实例存储卷。		字节
FreeLocalStoragePercent	可用本地存储空间的百分比。 此指标仅适用于具有 NVMe SSD 实例存储卷的数据库实例类。有关具有 NVMe SSD 实例存储卷的 Amazon EC2 实例的信息，请参阅实例存储卷。等效的 RDS 数据库实例类具有相同的实例存储卷。例如，db.m6gd 和 db.r6gd 数据库实例类具有 NVMe SSD 实例存储卷。		百分比
FreeStorageSpace	可用存储空间的大小。	全部	字节
FreeStorageSpaceLogVolume	日志卷上可用存储空间的大小。	启用了专用日志卷的数据库实例	字节
IamDbAuthConnectionRequests	使用 IAM 身份验证来访问数据库实例的连接请求数。	全部	计数
MaximumUsedTransactionIDs	已使用的最大事务 ID。	PostgreSQL	计数

指标	描述	适用于	单位
NetworkReceiveThroughput	数据库实例的传入（接收）网络流量，包括用于监控和复制的客户数据库流量和 Amazon RDS 流量。	全部	每秒字节数
NetworkTransmitThroughput	数据库实例的传出（传输）网络流量，包括用于监控和复制的客户数据库流量和 Amazon RDS 流量。	全部	每秒字节数
OldestLogicalReplicationSlotLag	Amazon RDS 在源数据库上提交事务与 RDS 在副本数据库上应用该事务的时间之间的滞后大小。	PostgreSQL	字节
OldestReplicationSlotLag	在接收预写日志 (WAL) 数据方面最滞后的副本的滞后大小。	PostgreSQL	字节
ReadIOPS	每秒平均磁盘读取 I/O 操作数。	全部	每秒计数
ReadIOPSLocalStorage	每秒至本地存储的平均磁盘读取输入/输出操作数。 此指标仅适用于具有 NVMe SSD 实例存储卷的数据库实例类。有关具有 NVMe SSD 实例存储卷的 Amazon EC2 实例的信息，请参阅实例存储卷。等效的 RDS 数据库实例类具有相同的实例存储卷。例如，db.m6gd 和 db.r6gd 数据库实例类具有 NVMe SSD 实例存储卷。		每秒计数
ReadIOPSLogVolume	每秒磁盘至日志卷平均读取 I/O 操作数。	启用了专用日志卷的数据库实例	每秒计数
ReadLatency	每个磁盘 I/O 操作所需的平均时间。	全部	秒

指标	描述	适用于	单位
ReadLatencyLocalStorage	每个磁盘对本地存储输入/输出操作所需的平均时间。 此指标仅适用于具有 NVMe SSD 实例存储卷的数据库实例类。有关具有 NVMe SSD 实例存储卷的 Amazon EC2 实例的信息，请参阅[实例存储卷]。等效的 RDS 数据库实例类具有相同的实例存储卷。例如，db.m6gd 和 db.r6gd 数据库实例类具有 NVMe SSD 实例存储卷。		秒
ReadLatencyLogVolume	每个磁盘至日志卷 I/O 操作所需的平均时间。	启用了[专用日志卷]的数据库实例	秒
ReadThroughput	每秒从磁盘读取的平均字节数。	全部	每秒字节数
ReadThroughputLocalStorage	每秒从磁盘至本地存储读取的平均字节数。 此指标仅适用于具有 NVMe SSD 实例存储卷的数据库实例类。有关具有 NVMe SSD 实例存储卷的 Amazon EC2 实例的信息，请参阅[实例存储卷]。等效的 RDS 数据库实例类具有相同的实例存储卷。例如，db.m6gd 和 db.r6gd 数据库实例类具有 NVMe SSD 实例存储卷。		每秒字节数
ReadThroughputLogVolume	每秒从磁盘向日志卷读取的平均字节数。	启用了[专用日志卷]的数据库实例	每秒字节数

指标	描述	适用于	单位
`ReplicaLag`	对于只读副本配置，只读副本数据库实例滞后于源数据库实例的时间量。适用于 MariaDB、Microsoft SQL Server、MySQL、Oracle 和 PostgreSQL 只读副本。 对于多可用区数据库集群，写入器数据库实例上的最新事务与读取器数据库实例上的最新应用事务之间的时间差异。		秒
`ReplicationChannelLag`	对于多源副本配置，多源副本上的特定通道滞后于源数据库实例的时间量。有关更多信息，请参阅the section called “监控多源复制通道”。	MySQL	秒
`ReplicationSlotDiskUsage`	副本槽文件使用的磁盘空间。	PostgreSQL	字节
`SwapUsage`	数据库实例上使用的交换空间的大小。	MariaDB MySQL Oracle PostgreSQL	字节
`TempDbAvailableDataSpace`	tempdb 和 tempdb 所在卷上的可用数据空间量。 使用此指标可监控 tempdb 数据空间可用性并相应地规划容量。较低的值可能表示需要增加存储空间或优化大量使用 tempdb 的查询。	SQL Server	字节

指标	描述	适用于	单位
TempDbAvailableLogSpace	tempdb 和 tempdb 所在卷上的可用日志空间量。 使用此指标可监控 tempdb 日志空间可用性并防止出现事务日志已满情况。对于大型事务或并发性高的工作负载至关重要，这些工作负载会生成大量日志活动。	SQL Server	字节
TempDbDataFileUsage	tempdb 上使用的数据文件的百分比。该指标未考虑潜在的文件增长。 使用此指标可监控 tempdb 数据文件利用率并识别潜在的性能瓶颈。较高的值可能表示需要优化创建大型临时对象的查询或增加 tempdb 大小。	SQL Server	百分比
TempDbLogFileUsage	tempdb 上使用的日志文件的百分比。该指标未考虑潜在的文件增长。 使用此指标可监控 tempdb 日志文件利用率并防止出现性能问题。较高的值可能表示长时间运行的事务或过多的日志记录活动可能会影响数据库的整体性能。	SQL Server	百分比
TransactionLogsDiskUsage	事务日志使用的磁盘空间。	PostgreSQL	字节
TransactionLogsGeneration	每秒生成的事务日志的大小。	PostgreSQL	每秒字节数
WriteIOPS	每秒平均磁盘写入 I/O 操作数。	全部	每秒计数

指标	描述	适用于	单位
WriteIOPSLocalStorage	本地存储上的每秒平均磁盘写入 I/O 操作数。 此指标仅适用于具有 NVMe SSD 实例存储卷的数据库实例类。有关具有 NVMe SSD 实例存储卷的 Amazon EC2 实例的信息，请参阅实例存储卷。等效的 RDS 数据库实例类具有相同的实例存储卷。例如，db.m6gd 和 db.r6gd 数据库实例类具有 NVMe SSD 实例存储卷。		每秒计数
WriteIOPSLogVolume	每秒磁盘至日志卷平均写入 I/O 操作数。	启用了专用日志卷的数据库实例	每秒计数
WriteLatency	每个磁盘 I/O 操作所需的平均时间。	全部	秒
WriteLatencyLocalStorage	本地存储上每个磁盘 I/O 操作所需的平均时间。 此指标仅适用于具有 NVMe SSD 实例存储卷的数据库实例类。有关具有 NVMe SSD 实例存储卷的 Amazon EC2 实例的信息，请参阅实例存储卷。等效的 RDS 数据库实例类具有相同的实例存储卷。例如，db.m6gd 和 db.r6gd 数据库实例类具有 NVMe SSD 实例存储卷。		秒
WriteLatencyLogVolume	每个磁盘至日志卷 I/O 操作所需的平均时间。	启用了专用日志卷的数据库实例	秒
WriteThroughput	每秒写入磁盘的平均字节数。	全部	每秒字节数

指标	描述	适用于	单位
WriteThroughputLogVolume	每秒日志卷写入磁盘的平均字节数。	启用了专用日志卷的数据库实例	每秒字节数
WriteThroughputLocalStorage	本地存储每秒写入磁盘的平均字节数。 此指标仅适用于具有 NVMe SSD 实例存储卷的数据库实例类。有关具有 NVMe SSD 实例存储卷的 Amazon EC2 实例的信息，请参阅实例存储卷。等效的 RDS 数据库实例类具有相同的实例存储卷。例如，db.m6gd 和 db.r6gd 数据库实例类具有 NVMe SSD 实例存储卷。		每秒字节数

Amazon RDS 的 Amazon CloudWatch 用量指标

Amazon CloudWatch 中的 AWS/Usage 命名空间包括 Amazon RDS 服务配额的账户级用量指标。CloudWatch 自动收集所有 Amazon Web Services 区域的使用量指标。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 CloudWatch 用量指标。有关配额的更多信息，请参阅《Service Quotas 用户指南》中 Amazon RDS 的配额和限制和请求增加配额。

指标	描述	单位*
AllocatedStorage	所有数据库实例的总存储空间。总和不包括临时迁移实例。	千兆字节
AuthorizationsPerDBSecurityGroup	您的 Amazon Web Services 账户中每个数据库安全组的入口规则数。使用的值是账户中数据库安全组中入口规则的最大数。账户中的其他数据库安全组的入口规则数可能较少。	计数
CustomEndpointsPerDBCluster	您的 Amazon Web Services 账户中每个数据库集群的自定义端点数。使用的值是账户中数据库集群内自定义端点的最大数。账户中的其他数据库集群可能具有更少的自定义端点数。	计数

指标	描述	单位*
CustomEngineVersions	您的 Amazon Web Services 账户中 Amazon RDS Custom 的自定义引擎版本（CEV）数量。	计数
DBClusterParameterGroups	您的 Amazon Web Services 账户中数据库集群参数组的数量。该计数不包括默认参数组。	计数
DBClusterRoles	您的 Amazon Web Services 账户中每个数据库集群的关联 Amazon Identity and Access Management（IAM）角色数。使用的值是账户中数据库集群关联的 IAM 角色的最大数量。账户中的其他数据库集群可能具有更少的关联 IAM 角色数。	计数
DBClusters	您的 Amazon Web Services 账户 中的 Amazon Aurora 数据库集群数量。	计数
DBInstanceRoles	您的 Amazon Web Services 账户中每个数据库实例的关联 Amazon Identity and Access Management（IAM）角色数。使用的值是账户中数据库实例关联的 IAM 角色的最大数量。账户中的其他数据库实例可能具有更少的关联 IAM 角色数。	计数
DBInstances	您的 Amazon Web Services 账户 中的数据库实例数量。	计数
DBParameterGroups	您的 Amazon Web Services 账户 中的数据库参数组数量。该计数不包括默认数据库参数组。	计数
DBSecurityGroups	您的 Amazon Web Services 账户 中的安全组数量。该计数不包括默认安全组和默认 VPC 安全组。	计数
DBSubnetGroups	您的 Amazon Web Services 账户 中的数据库子网组数量。该计数不包括默认子网组。	计数
EventSubscriptions	您的 Amazon Web Services 账户中的事件通知订阅数量。	计数
Integrations	您的 Amazon Web Services 账户中与 Amazon Redshift 进行的零 ETL 集成数量。	计数

指标	描述	单位*
ManualClusterSnapshots	您的 Amazon Web Services 账户 中手动创建的数据库集群快照数量。该计数不包括无效快照。	计数
ManualSnapshots	您的 Amazon Web Services 账户 中手动创建的数据库快照数量。该计数不包括无效快照。	计数
OptionGroups	您的 Amazon Web Services 账户 中的选项组数量。该计数不包括默认选项组。	计数
Proxies	您的 Amazon 账户中 RDS 代理的数量。	计数
ReadReplicasPerMaster	您的账户中每个数据库实例的只读副本数。使用的值是账户中数据库实例的只读副本的最大数。账户中的其他数据库实例可能具有更少的只读副本数。	计数
ReservedDBInstances	您的 Amazon Web Services 账户 中的预留数据库实例数量。该计数不包括停用或被拒绝的实例。	计数
SubnetsPerDBSubnetGroup	您的 Amazon Web Services 账户中每个数据库子网组的子网数。账户中数据库子网组中子网的最大数。账户中的其他数据库子网组可能具有更少的子网数量。	计数

> **Note**
> Amazon RDS 不会向 CloudWatch 发布用量指标的单位。这些单位仅出现在文档中。

Amazon RDS 的 Amazon CloudWatch 维度

您可以使用下表中的任何维度筛选 Amazon RDS 指标数据。

维度	筛选为 . . 请求的数据
DBInstanceIdentifier	特定的数据库实例。

维度	筛选为 . . 请求的数据
`DatabaseClass`	数据库类中的所有实例。例如，您可以聚合属于数据库类 `db.r5.large` 的所有实例的指标。
`EngineName`	仅标识的引擎名称。例如，您可以聚合具有引擎名称 `postgres` 的所有实例的指标。
`SourceRegion`	仅指定区域。例如，您可以聚合 `us-east-1` 区域中的所有数据库实例的指标。

Amazon RDS 性能详情的 Amazon CloudWatch 指标

性能详情会自动将一些指标发布到 Amazon CloudWatch。可以从 Performance Insights 查询相同的数据，但具有 CloudWatch 中的指标可以轻松地添加 CloudWatch 警报。还可以轻松地将指标添加到现有 CloudWatch 控制面板中。

指标	描述
DBLoad	数据库的活动会话的数量。通常，您需要活动会话的平均数量数据。在 Performance Insights 中，作为 `db.load.avg` 查询此数据。
DBLoadCPU	等待事件类型为 CPU 的活动会话的数量。在 Performance Insights 中，作为 `db.load.avg` 查询此数据，按等待事件类型 CPU 进行筛选。
DBLoadNonCPU	等待事件类型不为 CPU 的活动会话的数量。
DBLoadRelativeToNumVCPUs	数据库负载与数据库的虚拟 CPU 数量之比。

> **Note**
>
> 仅当数据库实例上有负载时，这些指标才会发布到 CloudWatch。

可以使用 CloudWatch 控制台、Amazon CLI 或 CloudWatch API 来查看这些指标。您还可以使用特殊的指标数学函数检查其他性能详情计数器指标。有关更多信息，请参阅 在 CloudWatch 中查询其他性能详情计数器指标。

例如，可以通过运行 get-metric-statistics 命令来获取 DBLoad 指标的统计数据。

```
aws cloudwatch get-metric-statistics \
    --region us-west-2 \
    --namespace AWS/RDS \
    --metric-name DBLoad  \
    --period 60 \
    --statistics Average \
    --start-time 1532035185 \
    --end-time 1532036185 \
    --dimensions Name=DBInstanceIdentifier,Value=db-loadtest-0
```

该示例将生成与下类似的输出。

```
{
  "Datapoints": [
  {
  "Timestamp": "2021-07-19T21:30:00Z",
  "Unit": "None",
  "Average": 2.1
  },
  {
  "Timestamp": "2021-07-19T21:34:00Z",
  "Unit": "None",
  "Average": 1.7
  },
  {
  "Timestamp": "2021-07-19T21:35:00Z",
  "Unit": "None",
  "Average": 2.8
  },
  {
  "Timestamp": "2021-07-19T21:31:00Z",
  "Unit": "None",
  "Average": 1.5
  },
  {
  "Timestamp": "2021-07-19T21:32:00Z",
  "Unit": "None",
```

```
"Average": 1.8
},
{
"Timestamp": "2021-07-19T21:29:00Z",
"Unit": "None",
"Average": 3.0
},
{
"Timestamp": "2021-07-19T21:33:00Z",
"Unit": "None",
"Average": 2.4
}
],
"Label": "DBLoad"
}
```


有关 CloudWatch 的更多信息，请参阅 Amazon CloudWatch 用户指南中的什么是 Amazon CloudWatch？。

在 CloudWatch 中查询其他性能详情计数器指标

> **Note**
>
> 如果您启用数据库洞察的高级模式，Amazon RDS 会向 Amazon CloudWatch 发布性能详情计数器指标。使用数据库洞察，不需要使用 `DB_PERF_INSIGHTS` 指标数学函数。您可以使用 CloudWatch 数据库洞察控制面板来搜索、查询性能详情计数器指标并为其设置警报。

您可以对 CloudWatch 中的 RDS 性能详情指标进行查询、创建警报和绘制图表。您可以使用 CloudWatch 的 `DB_PERF_INSIGHTS` 指标数学函数，来访问有关数据库实例的信息。借助此功能，您可以使用未直接报告给 CloudWatch 的性能详情指标来创建新的时间序列。

您可以在 CloudWatch 控制台的选择指标屏幕中单击添加数学下拉菜单，来使用新的指标数学函数。您可以使用它来创建有关性能详情指标，或者 CloudWatch 和性能详情指标组合的警报和图表，包括亚分钟指标的高分辨率警报。您也可以通过在 `get-metric-data` 请求中包含公制数学表达式来以编程方式使用该函数。有关更多信息，请参阅指标数学语法和函数，以及针对 Amazon 数据库中的性能详情计数器指标创建警报。

性能详情计数器指标

计数器指标是性能详情控制面板中的操作系统和数据库性能指标。为帮助确定和分析性能问题，您可将计数器指标与数据库负载相关联。您必须向指标追加一个统计函数来获取指标值。例如，`os.memory.active` 指标支持的函数为 `.avg`、`.min`、`.max`、`.sum` 和 `.sample_count`。

每分钟收集一次计数器指标。操作系统指标收集取决于增强监控是开启还是关闭。如果关闭增强监控，则每分钟收集一次操作系统指标。如果开启增强监控，则会收集所选时间段内的操作系统指标。有关开启或关闭增强监控的更多信息，请参阅打开和关闭增强监控。

主题

- 性能详情操作系统计数器
- 适用于 Amazon RDS for MariaDB 和 MySQL 的性能详情计数器
- 适用于 Amazon RDS for Microsoft SQL Server 的性能详情计数器
- Amazon RDS for Oracle 的性能详情计数器
- 适用于 Amazon RDS for PostgreSQL 的性能详情计数器

性能详情操作系统计数器

以下操作系统计数器（前缀为 os）可用于除 RDS for SQL Server 之外的所有 RDS 引擎的性能详情。

您可以使用 `ListAvailableResourceMetrics` API 获取数据库实例的可用计数器指标列表。有关更多信息，请参阅《Amazon RDS 性能详情 API 参考指南》中的 ListAvailableResourceMetrics。

计数器	类型	单位	指标	描述
Active	内存	千字节	os.memory.active	已分配的内存量 (以 KB 为单位)。
Buffers	内存	千字节	os.memory.buffers	在写入存储设备前用于缓存 I/O 请求的内存量 (以 KB 为单位)。
已缓存	内存	千字节	os.memory.cached	用于缓存基于文件系统的 I/O 的

计数器	类型	单位	指标	描述
				内存量，以 KB 为单位。
数据库缓存	内存	字节	os.memory .db.cache	包括 tmpfs (shmem) 在内的数据库进程用于页面缓存的内存量，以字节为单位。
数据库驻留集大小	内存	字节	os.memory .db.resid entSetSize	不包括 tmpfs (shmem) 的数据库进程用于匿名缓存和交换缓存的内存量，以字节为单位。
数据库交换	内存	字节	os.memory .db.swap	数据库进程用于交换的内存量，以字节为单位。
脏	内存	千字节	os.memory.dirty	RAM 中已修改但未写入存储中的相关数据块的内存页面大小 (以 KB 为单位)。
Free	内存	千字节	os.memory.free	未分配的内存量 (以 KB 为单位)。
Huge Pages Free	内存	页面	os.memory .hugePagesFree	可用大页数。大页是 Linux 内核的一项功能。
Huge Pages Rsvd	内存	页面	os.memory .hugePagesRsvd	已提交大页数。

计数器	类型	单位	指标	描述
Huge Pages Size	内存	千字节	os.memory .hugePagesSize	每个大页单位的大小 (以 KB 为单位)。
Huge Pages Surp	内存	页面	os.memory .hugePagesSurp	剩余可用大页总数。
Huge Pages Total	内存	页面	os.memory .hugePagesTotal	大页总数。
Inactive	内存	千字节	os.memory .inactive	最不常用内存页面大小 (以 KB 为单位)。
已映射	内存	千字节	os.memory .mapped	在进程地址空间中有内存映射的文件系统内容的总大小 (以 KB 为单位)。
内存不足终止计数	内存	终止次数	os.memory .outOfMem oryKillCount	在上次收集间隔内发生的 OOM 终止次数。
Page Tables	内存	千字节	os.memory .pageTables	页表使用的内存量 (以 KB 为单位)。
碎片	内存	千字节	os.memory.slab	可重用内核数据结构大小 (以 KB 为单位)。
Total	内存	千字节	os.memory.total	内存总量 (以 KB 为单位)。

计数器	类型	单位	指标	描述
回写	内存	千字节	os.memory .writeback	RAM 中仍在写入备份存储的脏页大小 (以 KB 为单位)。
来宾	Cpu 利用率	百分比	os.cpuUtilization. guest	来宾程序使用的 CPU 百分比。
Idle	Cpu 利用率	百分比	os.cpuUtilization. idle	CPU 空闲百分比。
Irq	Cpu 利用率	百分比	os.cpuUtilization. irq	软件中断使用的 CPU 百分比。
良好	Cpu 利用率	百分比	os.cpuUtilization. nice	以最低优先级运行的程序使用的 CPU 百分比。
被盗用	Cpu 利用率	百分比	os.cpuUtilization. steal	其他虚拟机使用的 CPU 百分比。
System	Cpu 利用率	百分比	os.cpuUtilization. system	内核使用的 CPU 百分比。
Total	Cpu 利用率	百分比	os.cpuUtilization. total	使用中的 CPU 百分比总计。此值包含 nice（良好）值。
User	Cpu 利用率	百分比	os.cpuUtilization. user	用户程序使用的 CPU 百分比。
Wait	Cpu 利用率	百分比	os.cpuUtilization. wait	等待 I/O 访问时的未使用 CPU 百分比。

计数器	类型	单位	指标	描述
读取 IO PS	磁盘 IO	每秒请求数	os.diskIO.<devicename>.readIOsPS	每秒的读取操作数。
写入 IO PS	磁盘 IO	每秒请求数	os.diskIO.<devicename>.writeIOsPS	每秒的写入操作数。
平均队列长度	磁盘 IO	请求	os.diskIO.<devicename>.avgQueueLen	在 I/O 设备队列中等待的请求数。
平均请求大小	磁盘 IO	请求	os.diskIO.<devicename>.avgReqSz	在 I/O 设备队列中等待的请求数。
等待	磁盘 IO	毫秒	os.diskIO.<devicename>.await	响应请求所需的毫秒数，包括排队时间和服务时间。
读取 IO PS	磁盘 IO	请求	os.diskIO.<devicename>.readIOsPS	每秒的读取操作数。
读取 KB	磁盘 IO	千字节	os.diskIO.<devicename>.readKb	读取的总 KB 数。

计数器	类型	单位	指标	描述
Read KB PS	磁盘 IO	千字节/秒	os.diskIO.<devicename>.readKbPS	每秒读取的 KB 数。
Rrqm PS	磁盘 IO	每秒请求数	os.diskIO.<devicename>.rrqmPS	每秒排队的合并读取请求数。
TPS	磁盘 IO	每秒事务数	os.diskIO.<devicename>.tps	每秒的 I/O 事务数。
Util	磁盘 IO	百分比	os.diskIO.<devicename>.util	发出请求所经历的 CPU 时间的百分比。
Write KB	磁盘 IO	千字节	os.diskIO.<devicename>.writeKb	写入的总 KB 数。
写入 KB PS	磁盘 IO	千字节/秒	os.diskIO.<devicename>.writeKbPS	每秒写入的 KB 数。
Wrqm PS	磁盘 IO	每秒请求数	os.diskIO.<devicename>.wrqmPS	每秒排队的合并写入请求数。
阻止	任务	任务	os.tasks.blocked	已阻止的任务的数量。

计数器	类型	单位	指标	描述
Running	任务	任务	os.tasks.running	正在运行的任务的数量。
Sleeping	任务	任务	os.tasks.sleeping	正在睡眠的任务的数量。
Stopped（已停止）	任务	任务	os.tasks.stopped	已停止的任务的数量。
Total	任务	任务	os.tasks.total	任务总数。
Zombie	任务	任务	os.tasks.zombie	有活动父任务的不活动子任务的数量。
一	加载平均分钟数	进程	os.loadAverageMinute.one	过去 1 分钟内请求 CPU 时间的进程数。
十五	加载平均分钟数	进程	os.loadAverageMinute.fifteen	过去 15 分钟内请求 CPU 时间的进程数。
五	加载平均分钟数	进程	os.loadAverageMinute.five	过去 5 分钟内请求 CPU 时间的进程数。
已缓存	Swap (交换)	千字节	os.swap.cached	用作缓存内存的交换内存量 (以 KB 为单位)。
免费	Swap (交换)	千字节	os.swap.free	空闲交换内存量（以 KB 为单位）。

计数器	类型	单位	指标	描述
In	Swap (交换)	千字节	os.swap.in	从磁盘换入的内存量（以 KB 为单位）。
Out	Swap (交换)	千字节	os.swap.out	换出到磁盘的内存量（以 KB 为单位）。
Total	Swap (交换)	千字节	os.swap.total	可用的总交换内存量（以 KB 为单位）。
最大文件数	文件系统	文件	os.fileSys.maxFiles	可为文件系统创建的文件的最大数量。
已用文件	文件系统	文件	os.fileSys.usedFiles	文件系统中的文件数。
已用文件百分比	文件系统	文件	os.fileSys.usedFilePercent	使用中的可用文件百分比。
已用百分比	文件系统	百分比	os.fileSys.usedPercent	使用中的文件系统磁盘空间百分比。
已用	文件系统	千字节	os.fileSys.used	文件系统中的文件所用的磁盘空间量 (以 KB 为单位)。
Total	文件系统	千字节	os.fileSys.total	文件系统的可用磁盘空间总量 (以 KB 为单位)。

计数器	类型	单位	指标	描述
Rx	网络	每秒字节数	os.network.rx	每秒接收的字节数。
Tx	网络	每秒字节数	os.network.tx	每秒上传的字节数。
Acu 利用率	常规	百分比	os.general.acuUtilization	当前容量占最大配置容量的百分比。
最大配置的 Acu	常规	ACU	os.general.maxConfiguredAcu	用户配置的最大容量，以 Aurora 容量单位（ACU）为单位。
最小配置的 Acu	常规	ACU	os.general.minConfiguredAcu	用户配置的最小容量，以 ACU 为单位。
VCPU 数量	常规	vCPU	os.general.numVCPUs	数据库实例的虚拟 CPU（vCPU）数量。
无服务器数据库容量	常规	ACU	os.general.serverlessDatabaseCapacity	实例以 ACU 表示的当前容量。

适用于 Amazon RDS for MariaDB 和 MySQL 的性能详情计数器

以下数据库计数器可用于适用于 Amazon RDS for MariaDB 和 MySQL 的性能详情。

主题

- RDS for MariaDB 和 RDS for MySQL 的本机计数器

- 适用于 Amazon RDS for MariaDB 和 RDS for MySQL 的非本机计数器

RDS for MariaDB 和 RDS for MySQL 的本机计数器

本机指标由数据库引擎定义，而不是由 Amazon RDS 定义。有关这些本机指标的定义，请参阅 MySQL 文档中的 Server Status Variables（适用于 8.0）和 Server Status Variables（适用于 8.4）。

计数器	类型	单位	指标
Com_analyze	SQL	每秒查询数	db.SQL.Com_analyze
Com_optimize	SQL	每秒查询数	db.SQL.Com_optimize
Com_select	SQL	每秒查询数	db.SQL.Com_select
连接	SQL	每分钟尝试连接到 MySQL 服务器的次数（无论成功与否）	db.Users.Connections
Innodb_rows_deleted	SQL	每秒行数	db.SQL.Innodb_rows_deleted
Innodb_rows_inserted	SQL	每秒行数	db.SQL.Innodb_rows_inserted
Innodb_rows_read	SQL	每秒行数	db.SQL.Innodb_rows_read
Innodb_rows_updated	SQL	每秒行数	db.SQL.Innodb_rows_updated
Select_full_join	SQL	每秒查询数	db.SQL.Select_full_join
Select_full_range_join	SQL	每秒查询数	db.SQL.Select_full_range_join
Select_range	SQL	每秒查询数	db.SQL.Select_range
Select_range_check	SQL	每秒查询数	db.SQL.Select_range_check
Select_scan	SQL	每秒查询数	db.SQL.Select_scan

计数器	类型	单位	指标
Slow_queries	SQL	每秒查询数	db.SQL.Slow_queries
Sort_merge_passes	SQL	每秒查询数	db.SQL.Sort_merge_passes
Sort_range	SQL	每秒查询数	db.SQL.Sort_range
Sort_rows	SQL	每秒查询数	db.SQL.Sort_rows
Sort_scan	SQL	每秒查询数	db.SQL.Sort_scan
问题	SQL	每秒查询数	db.SQL.Questions
Innodb_row_lock_time	锁	毫秒数（平均值）	db.Locks.Innodb_row_lock_time
Table_locks_immediate	锁	每秒请求数	db.Locks.Table_locks_immediate
Table_locks_waited	锁	每秒请求数	db.Locks.Table_locks_waited
Aborted_clients	用户	连接	db.Users.Aborted_clients
Aborted_connects	用户	连接	db.Users.Aborted_connects
max_connections	Users	连接	db.User.max_connections
Threads_created	用户	连接	db.Users.Threads_created
Threads_running	用户	连接	db.Users.Threads_running
Innodb_data_writes	I/O	每秒操作数	db.IO.Innodb_data_writes
Innodb_dblwr_writes	I/O	每秒操作数	db.IO.Innodb_dblwr_writes
Innodb_log_write_requests	I/O	每秒操作数	db.IO.Innodb_log_write_requests
Innodb_log_writes	I/O	每秒操作数	db.IO.Innodb_log_writes
Innodb_pages_written	I/O	每秒页数	db.IO.Innodb_pages_written

计数器	类型	单位	指标
Created_tmp_disk_tables	临时文件	每秒表数	db.Temp.Created_tmp_disk_tables
Created_tmp_tables	临时文件	每秒表数	db.Temp.Created_tmp_tables
Innodb_buffer_pool_pages_data	缓存	页面	db.Cache.Innodb_buffer_pool_pages_data
Innodb_buffer_pool_pages_total	缓存	页面	db.Cache.Innodb_buffer_pool_pages_total
Innodb_buffer_pool_read_requests	缓存	每秒页数	db.Cache.Innodb_buffer_pool_read_requests
Innodb_buffer_pool_reads	缓存	每秒页数	db.Cache.Innodb_buffer_pool_reads
Opened_tables	缓存	表	db.Cache.Opened_tables
Opened_table_definitions	缓存	表	db.Cache.Opened_table_definitions
Qcache_hits	缓存	查询	db.Cache.Qcache_hits

适用于 Amazon RDS for MariaDB 和 RDS for MySQL 的非本机计数器

非本机计数器指标是 Amazon RDS 定义的计数器。非本机指标可以是您使用特定查询获取的指标。非本机指标还可以是派生指标，使用两个或更多个本机计数器来计算比率、命中率或延迟。

计数器	类型	单位	指标	描述	定义
innodb_buffer_pool_hits	缓存	写入	db.Cache.innoDB_buffer_pool_hits	InnoDB 可满足的缓冲池中的读取数。	`innodb_buffer_pool_read_requests - innodb_bu`

计数器	类型	单位	指标	描述	定义
					ffer_pool _reads
innodb_buffer_pool _hit_rate	缓存	百分比	db.Cache. innoDB_bu ffer_pool_hit_rate	InnoDB 可满足的缓冲池中的读取百分比。	`100 * innodb_bu ffer_pool _read_req uests / (innodb_b uffer_poo l_read_re quests + innodb_ buffer_po ol_reads)`

计数器	类型	单位	指标	描述	定义
innodb_buffer_pool_usage	缓存	百分比	db.Cache.innoDB_buffer_pool_usage	包含数据（页面）的 InnoDB 缓冲池的百分比。 Note 在使用压缩表时，此值可改变。有关更多信息，请参阅 MySQL 文档的 Server	`Innodb_buffer_pool_pages_data / Innodb_buffer_pool_pages_total * 100.0`

计数器	类型	单位	指标	描述	定义
				Status Variab（适用于 8.0）和 Server Status Variab（适用于 8.4）中有关 Innodffer__page ta 和 Innodffer__page tal 的信息。	

计数器	类型	单位	指标	描述	定义
query_cache_hit_rate	缓存	百分比	db.Cache.query_cache_hit_rate	MySQL 结果集缓存（查询缓存）命中率。	`Qcache_hits / (QCache_hits + Com_select) * 100`
innodb_datafile_writes_to_disk	I/O	读取	db.IO.innoDB_datafile_writes_to_disk	InnoDB 数据文件写入磁盘的操作数，不包括两次写和重做日志记录写入操作。	`Innodb_data_writes - Innodb_log_writes - Innodb_dblwr_writes`
innodb_rows_changed	SQL	行	db.SQL.innodb_rows_changed	总 InnoDB 行操作数。	`db.SQL.Innodb_rows_inserted + db.SQL.Innodb_rows_deleted + db.SQL.Innodb_rows_updated`
active_transactions	事务	事务	db.Transactions.active_transactions	总活动事务数。	`SELECT COUNT(1) AS active_transactions FROM INFORMATION_SCHEMA.INNODB_TRX`

计数器	类型	单位	指标	描述	定义
trx_rseg_history_len	事务	无	db.Transactions.trx_rseg_history_len	InnoDB 事务系统维护的已提交事务的撤消日志页面列表，用于实现多版本并发控制。有关撤消日志记录详细信息的更多信息，请参阅 MySQL 文档中的 InnoDB Multi-Versioning（适用于 8.0）和 InnoDB Multi-Versioning（适用于 8.4）。	`SELECT COUNT AS trx_rseg_history_len FROM INFORMATION_SCHEMA.INNODB_METRICS WHERE NAME='trx_rseg_history_len'`

计数器	类型	单位	指标	描述	定义
innodb_deadlocks	锁	锁	db.Locks.innodb_deadlocks	死锁总数。	`SELECT COUNT AS innodb_deadlocks FROM INFORMATION_SCHEMA.INNODB_METRICS WHERE NAME='lock_deadlocks'`
innodb_lock_timeouts	锁	锁	db.Locks.innodb_lock_timeouts	超时的锁定的总数。	`SELECT COUNT AS innodb_lock_timeouts FROM INFORMATION_SCHEMA.INNODB_METRICS WHERE NAME='lock_timeouts'`
innodb_row_lock_waits	锁	锁	db.Locks.innodb_row_lock_waits	导致等待的行锁总数。	`SELECT COUNT AS innodb_row_lock_waits FROM INFORMATION_SCHEMA.INNODB_METRICS WHERE NAME='lock_row_lock_waits'`

适用于 Amazon RDS for Microsoft SQL Server 的性能详情计数器

以下数据库计数器可用于 RDS for Microsoft SQL Server 的性能详情。

适用于 RDS for Microsoft SQL Server 的本机计数器

本机指标由数据库引擎定义，而不是由 Amazon RDS 定义。您可以在 Microsoft SQL Server 文档的使用 SQL Server 对象中找到这些本机指标的定义。

计数器	类型	单位	指标
转发的记录数	访问方法	每秒记录数	db.Access Methods.Forwarded Records
分页数	访问方法	每秒拆分次数	db.Access Methods.Page Splits
缓冲区缓存命中率	缓冲区管理器	Ratio	db.Buffer Manager.Buffer cache hit ratio
页生命周期	缓冲区管理器	寿命（以秒为单位）	db.Buffer Manager.Page life expectancy
页查找次数	缓冲区管理器	每秒查找次数	db.Buffer Manager.Page lookups
页读取次数	缓冲区管理器	每秒读取次数	db.Buffer Manager.Page reads
页写入次数	缓冲区管理器	每秒写入次数	db.Buffer Manager.Page writes
活动事务数	数据库	事务	db.Databases.Active Transactions (_Total)
刷新的日志字节数	数据库	每秒刷新的日志字节数	db.Databases.Log Bytes Flushed (_Total)
日志刷新等待次数	数据库	每秒等待次数	db.Databases.Log Flush Waits (_Total)
日志刷新次数	数据库	每秒刷新次数	db.Databases.Log Flushes (_Total)

计数器	类型	单位	指标
写入事务数	数据库	每秒事务数	db.Databases.Write Transactions (_Total)
阻止的进程数	常规统计数据	阻止的进程数	db.General Statistic s.Processes blocked
用户连接数	常规统计数据	连接	db.General Statistic s.User Connections
闩锁等待次数	闩锁	每秒等待次数	db.Latches.Latch Waits
死锁数	锁	每秒死锁数	db.Locks.Number of Deadlocks (_Total)
内存授予挂起	内存管理器	内存授予	db.Memory Manager.Memory Grants Pending
批处理请求数	SQL 统计数据	每秒请求数	db.SQL Statistic s.Batch Requests
SQL 编译	SQL 统计数据	每秒编译数	db.SQL Statistics.SQL Compilations
SQL 重新编译	SQL 统计数据	每秒重新编译数	db.SQL Statistics.SQL Re-Compilations

Amazon RDS for Oracle 的性能详情计数器

以下数据库计数器可用于 RDS for Oracle 的性能详情。

适用于 RDS for Oracle 的本机计数器

本机指标由数据库引擎定义，而不是由 Amazon RDS 定义。您可以在 Oracle 文档中的统计数据描述中找到这些本机指标的定义。

Note

对于 CPU used by this session 计数器指标，单位已从本机厘秒转换为活动会话以使值更易于使用。例如，“DB Load (数据库负载)”图表中的“CPU send (CPU 发送)”表示 CPU 需求。计数器指标 CPU used by this session 表示 Oracle 会话使用的 CPU 量。您可以将“CPU 发送”与 CPU used by this session 计数器指标进行比较。当 CPU 需求高于使用的 CPU 时，会话将等待 CPU 时间。

计数器	类型	单位	指标
此会话使用的 CPU	User	活动的会话	db.User.CPU used by this session
SQL*Net 与客户端之间的往返	User	每秒往返	db.User.SQL*Net roundtrips to/from client
通过 SQL*Net 从客户端收到的字节数	User	每秒字节数	db.User.bytes received via SQL*Net from client
用户提交	User	每秒提交数	db.User.user commits
累积登录数	User	每秒登录数	db.User.logons cumulative
用户调用	User	每秒调用数	db.User.user calls
通过 SQL*Net 发送到客户端的字节数	User	每秒字节数	db.User.bytes sent via SQL*Net to client
用户回滚	User	每秒回滚数	db.User.user rollbacks
重做大小	Redo	每秒字节数	db.Redo.redo size
分析计数（总计）	SQL	每秒分析数	db.SQL.parse count (total)

计数器	类型	单位	指标
分析计数（硬）	SQL	每秒分析数	db.SQL.parse count (hard)
获得的表扫描行数	SQL	每秒行数	db.SQL.table scan rows gotten
分类（内存）	SQL	每秒分类数	db.SQL.sorts (memory)
分类（磁盘）	SQL	每秒分类数	db.SQL.sorts (disk)
分类（行）	SQL	每秒分类数	db.SQL.sorts (rows)
物理读取字节数	缓存	每秒字节数	db.Cache.physical read bytes
数据库块获取数	缓存	每秒块数	db.Cache.db block gets
DBWR 检查点	缓存	每分钟检查点数	db.Cache.DBWR checkpoints
物理读取	缓存	每秒读取次数	db.Cache.physical reads
从缓存进行的一致性获取	缓存	每秒获取数	db.Cache.consistent gets from cache
从缓存进行的数据库块获取数	缓存	每秒获取数	db.Cache.db block gets from cache
一致性获取	缓存	每秒获取数	db.Cache.consistent gets

适用于 Amazon RDS for PostgreSQL 的性能详情计数器

以下数据库计数器可用于 Amazon RDS for PostgreSQL 的性能详情。

主题

- [Amazon RDS for PostgreSQL 的本机计数器]
- [Amazon RDS for PostgreSQL 的非本机计数器]

Amazon RDS for PostgreSQL 的本机计数器

本机指标由数据库引擎定义，而不是由 Amazon RDS 定义。您可以在 PostgreSQL 文档中的查看统计数据中找到这些本机指标的定义。

计数器	类型	单位	指标
blks_hit	缓存	每秒块数	db.Cache.blks_hit
buffers_alloc	缓存	每秒块数	db.Cache.buffers_alloc
buffers_checkpoint	检查点	每秒块数	db.Checkpoint.buffers_checkpoint
checkpoint_sync_time	检查点	每个检查点的毫秒数	db.Checkpoint.checkpoint_sync_time
checkpoint_write_time	检查点	每个检查点的毫秒数	db.Checkpoint.checkpoint_write_time
checkpoints_req	检查点	每分钟检查点数	db.Checkpoint.checkpoints_req
checkpoints_timed	检查点	每分钟检查点数	db.Checkpoint.checkpoints_timed
maxwritten_clean	检查点	每分钟 Bgwriter 清理停止数	db.Checkpoint.maxwritten_clean
deadlocks	并发	每分钟死锁数	db.Concurrency.deadlocks
blk_read_time	I/O	毫秒	db.IO.blk_read_time
blks_read	I/O	每秒块数	db.IO.blks_read
buffers_backend	I/O	每秒块数	db.IO.buffers_backend
buffers_backend_fsync	I/O	每秒块数	db.IO.buffers_backend_fsync

计数器	类型	单位	指标
buffers_clean	I/O	每秒块数	db.IO.buffers_clean
tup_deleted	SQL	每秒元组数	db.SQL.tup_deleted
tup_fetched	SQL	每秒元组数	db.SQL.tup_fetched
tup_inserted	SQL	每秒元组数	db.SQL.tup_inserted
tup_returned	SQL	每秒元组数	db.SQL.tup_returned
tup_updated	SQL	每秒元组数	db.SQL.tup_updated
temp_bytes	临时文件	每秒字节数	db.Temp.temp_bytes
temp_files	临时文件	每分钟文件数	db.Temp.temp_files
xact_commit	事务	每秒提交数	db.Transactions.xact_commit
xact_rollback	事务	每秒回滚数	db.Transactions.xact_rollback
numbackends	User	连接	db.User.numbackends
archived_count	预写日志 (WAL)	每分钟文件数	db.WAL.archived_count

Amazon RDS for PostgreSQL 的非本机计数器

非本机计数器指标是 Amazon RDS 定义的计数器。非本机指标可以是您使用特定查询获取的指标。非本机指标还可以是派生指标，使用两个或更多个本机计数器来计算比率、命中率或延迟。

计数器	类型	单位	指标	描述	定义
checkpoint_sync_latency	检查点	db.Checkpoint.checkpoint_sync_latency		在文件同步到磁盘时，检查点处理部分已花费的总时间量。	`checkpoint_sync_time / (checkpoints_timed`

计数器	类型	单位	指标	描述	定义
					+ checkpoints_req)
checkpoint_write_latency	检查点	db.Checkpoint.checkpoint_write_latency	在文件写入磁盘时，检查点处理部分已花费的总时间量。	checkpoint_write_time / (checkpoints_timed + checkpoints_req)	
read_latency	I/O	db.IO.read_latency	此实例中的后端读取数据文件块所花费的时间。	blk_read_time / blks_read	
idle_in_transaction_aborted_count	状态	会话	db.state.idle_in_transaction_aborted_count	处于 idle in transaction (aborted) 状态的会话数量。	不适用
idle_in_transaction_count	状态	会话	db.state.idle_in_transaction_count	处于 idle in transaction 状态的会话数量。	不适用
idle_in_transaction_max_time	状态	秒	db.state.idle_in_transaction_max_time	idle in transaction 状态下运行时间最长的事务的持续时间（以秒为单位）。	不适用
active_transactions	事务	事务	db.Transactions.active_transactions	活动事务的数量。	不适用
blocked_transactions	事务	事务	db.Transactions.blocked_transactions	阻止的事务的数量。	不适用

计数器	类型	单位	指标	描述	定义
oldest_active_logical_replication_slot_xid_age	事务	db.Tractions.dest_ave_logl_replition_sl_xid_a	活动逻辑复制插槽中最早事务的期限。 有关更多信息，请参阅 逻辑复制槽。	–	
oldest_inactive_logical_replication_slot_xid_age	事务	db.Tractions.dest_intive_local_repcation_ot_xid_e	非活动逻辑复制插槽中最早事务的期限。 有关更多信息，请参阅 逻辑复制槽。	–	
oldest_prepared_transaction_xid_age	事务	db.Tractions.dest_pared_tsactionid_age	最早的准备好的事务的期限。 有关更多信息，请参阅 准备好的事务。	–	
oldest_running_transaction_xid_age	事务	db.Tractions.dest_ruing_traaction_d_age	最早的运行事务的期限。 有关更多信息，请参阅活动语句以了解最早的运行活动事务，以及参阅事务中空闲以了解最早的运行事务中的空闲。	–	

计数器	类型	单位	指标	描述	定义
oldest_hot_standby_feedback_xid_age	事务	db.Tractions.dest_hstandbeedbaid_age	启用了 `hot_standby_feedback` 的只读副本上最早的运行事务的期限。 有关更多信息，请参阅 [只读副本](#)。	–	
max_used_xact_ids	事务	事务	db.Transactions.max_used_xact_ids	尚未清空的事务数量。	不适用
max_connections	Users	连接	db.User.max_connections	`max_connections` 参数中配置的数据库实例允许的最大连接数。	不适用
archive_failed_count	WAL	每分钟文件数	db.WAL.archive_failed_count	尝试存档 WAL 文件的失败次数（以每分钟文件数为单位）。	不适用

Performance Insights 的 SQL 统计数据

SQL 统计数据是由 Performance Insights 收集的关于 SQL 查询的性能相关指标。Performance Insights 收集查询运行的每一秒的统计数据，以及每次 SQL 调用的统计数据。SQL 统计数据是所选时间范围内的平均值。

SQL 摘要 是具有给定模式但不一定具有相同文本值的所有查询的组合。摘要用问号替换文本值。例如，`SELECT * FROM emp WHERE lname= ?`。此摘要可能由以下子查询组成：

```
SELECT * FROM emp WHERE lname = 'Sanchez'
SELECT * FROM emp WHERE lname = 'Olagappan'
SELECT * FROM emp WHERE lname = 'Wu'
```

所有引擎都支持摘要查询的 SQL 统计数据。

有关此功能的区域、数据库引擎和实例类支持信息，请参阅支持性能详情功能的 Amazon RDS 数据库引擎、区域和实例类

主题

- MariaDB 和 MySQL 的 SQL 统计数据
- Amazon RDS for Oracle 的 SQL 统计数据
- Amazon RDS for SQL Server 的 SQL 统计数据
- RDS PostgreSQL 的 SQL 统计数据

MariaDB 和 MySQL 的 SQL 统计数据

MariaDB 和 MySQL 仅在摘要级别收集 SQL 统计信息。语句级别没有显示任何统计数据。

主题

- MariaDB 和 MySQL 的摘要统计数据
- MariaDB 和 MySQL 的每秒统计数据
- MariaDB 和 MySQL 的每次调用统计数据
- MariaDB 和 MySQL 的主要统计数据

MariaDB 和 MySQL 的摘要统计数据

性能详情从 `events_statements_summary_by_digest` 表中收集 SQL 摘要统计数据。`events_statements_summary_by_digest` 表由您的数据库管理。

此摘要表不提供移出策略。当表已满时，Amazon Web Services Management Console 将显示以下消息：

```
Performance Insights is unable to collect SQL Digest statistics on new queries because
 the table events_statements_summary_by_digest is full.
Please truncate events_statements_summary_by_digest table to clear the issue. Check the
 User Guide for more details.
```

在这种情况下，MariaDB 和 MySQL 不会 跟踪 SQL 查询。为了解决此问题，性能详情会在满足以下两个条件时自动截取摘要表：

- 表已满。

- 性能详情会自动管理 Performance Schema。

  对于自动管理，`performance_schema` 参数必须设置为 `0`，Source (源) 不应设置为 `user`。如果性能详情没有自动管理 Performance Schema，请参阅 Amazon RDS for MariaDB 或 Amazon RDS for MySQL 上性能详情的性能架构概述。

在 Amazon CLI 中，检查参数值的来源，方法是运行 describe-db-parameters 命令。

MariaDB 和 MySQL 的每秒统计数据

以下 SQL 统计数据适用于 MariaDB 和 MySQL 数据库实例。

指标	Unit
db.sql_tokenized.stats.count_star_per_sec	每秒调用数
db.sql_tokenized.stats.sum_timer_wait_per_sec	每秒平均延迟（以毫秒为单位）
db.sql_tokenized.stats.sum_select_full_join_per_sec	选择每秒完全联接数
db.sql_tokenized.stats.sum_select_range_check_per_sec	选择每秒范围检查数
db.sql_tokenized.stats.sum_select_scan_per_sec	选择每秒扫描数
db.sql_tokenized.stats.sum_sort_merge_passes_per_sec	对每秒合并传递数进行排序
db.sql_tokenized.stats.sum_sort_scan_per_sec	对每秒对扫描数进行排序
db.sql_tokenized.stats.sum_sort_range_per_sec	对每秒范围数进行排序
db.sql_tokenized.stats.sum_sort_rows_per_sec	对每秒行数进行排序
db.sql_tokenized.stats.sum_rows_affected_per_sec	每秒受影响的行数

指标	Unit
db.sql_tokenized.stats.sum_rows_examined_per_sec	每秒检查的行数
db.sql_tokenized.stats.sum_rows_sent_per_sec	每秒发送的行数
db.sql_tokenized.stats.sum_created_tmp_disk_tables_per_sec	每秒创建的临时磁盘表数
db.sql_tokenized.stats.sum_created_tmp_tables_per_sec	每秒创建的临时表数
db.sql_tokenized.stats.sum_lock_time_per_sec	每秒锁定时间（以毫秒为单位）

MariaDB 和 MySQL 的每次调用统计数据

以下指标提供了 SQL 语句的每次调用统计数据。

指标	单位
db.sql_tokenized.stats.sum_timer_wait_per_call	每个调用的平均延迟（以毫秒为单位）
db.sql_tokenized.stats.sum_select_full_join_per_call	选择每次调用完全联接数
db.sql_tokenized.stats.sum_select_range_check_per_call	选择每次调用范围检查数
db.sql_tokenized.stats.sum_select_scan_per_call	选择每次调用扫描数
db.sql_tokenized.stats.sum_sort_merge_passes_per_call	对每次调用合并传递数进行排序
db.sql_tokenized.stats.sum_sort_scan_per_call	对每次调用扫描数进行排序
db.sql_tokenized.stats.sum_sort_range_per_call	对每次调用范围数进行排序

指标	单位
db.sql_tokenized.stats.sum_sort_rows_per_call	对每次调用行数进行排序
db.sql_tokenized.stats.sum_rows_affected_per_call	每次调用受影响的行数
db.sql_tokenized.stats.sum_rows_examined_per_call	每次调用检查的行数
db.sql_tokenized.stats.sum_rows_sent_per_call	每次调用发送的行数
db.sql_tokenized.stats.sum_created_tmp_disk_tables_per_call	每次调用创建的临时磁盘表数
db.sql_tokenized.stats.sum_created_tmp_tables_per_call	每次调用创建的临时表数
db.sql_tokenized.stats.sum_lock_time_per_call	每个调用的锁定时间（以毫秒为单位）

MariaDB 和 MySQL 的主要统计数据

以下 SQL 统计数据适用于 MariaDB 和 MySQL 数据库实例。

指标	单位
db.sql_tokenized.stats.count_star	Calls
db.sql_tokenized.stats.sum_timer_wait	等待时间（以毫秒为单位）
db.sql_tokenized.stats.sum_select_full_join	选择完全联接
db.sql_tokenized.stats.sum_select_range_check	选择范围检查
db.sql_tokenized.stats.sum_select_scan	选择扫描
db.sql_tokenized.stats.sum_sort_merge_passes	对合并传递进行排序
db.sql_tokenized.stats.sum_sort_scan	对扫描进行排序

指标	单位
db.sql_tokenized.stats.sum_sort_range	对范围进行排序
db.sql_tokenized.stats.sum_sort_rows	排序行
db.sql_tokenized.stats.sum_rows_affected	受影响的行数
db.sql_tokenized.stats.sum_rows_examined	检查的行数
db.sql_tokenized.stats.sum_rows_sent	发送的行数
db.sql_tokenized.stats.sum_created_tmp_disk_tables	创建的临时磁盘表数
db.sql_tokenized.stats.sum_created_tmp_tables	创建的临时表数
db.sql_tokenized.stats.sum_lock_time	锁定时间（以毫秒为单位）

Amazon RDS for Oracle 的 SQL 统计数据

Amazon RDS for Oracle 收集语句和摘要级别的 SQL 统计信息。在语句级别，ID 列表示 `V$SQL.SQL_ID` 的值。在摘要级别，ID 列显示 `V$SQL.FORCE_MATCHING_SIGNATURE` 的值。

如果在摘要级别，ID 为 `0`，则 Oracle 数据库已确定此语句不适合重复使用。在这种情况下，子 SQL 语句可能属于不同的摘要。不过，这些语句都将分组在第一个收集到的 SQL 语句的 `digest_text` 项下。

主题

- Oracle 的每秒统计数据
- Oracle 的每次调用统计数据
- Oracle 的主要统计数据

Oracle 的每秒统计数据

以下指标提供了 Oracle SQL 查询的每秒统计数据。

指标	单位
db.sql.stats.executions_per_sec	每秒执行次数
db.sql.stats.elapsed_time_per_sec	平均活动执行次数 (AAE)
db.sql.stats.rows_processed_per_sec	每秒处理的行数
db.sql.stats.buffer_gets_per_sec	每秒缓冲区获取数
db.sql.stats.physical_read_requests_per_sec	每秒物理读取次数
db.sql.stats.physical_write_requests_per_sec	每秒物理写入次数
db.sql.stats.total_sharable_mem_per_sec	每秒总可共享内存（以字节为单位）
db.sql.stats.cpu_time_per_sec	每秒 CPU 时间（以毫秒为单位）

以下指标提供了 Oracle SQL 摘要查询的每秒统计数据。

指标	单位
db.sql_tokenized.stats.executions_per_sec	每秒执行次数
db.sql_tokenized.stats.elapsed_time_per_sec	平均活动执行次数 (AAE)
db.sql_tokenized.stats.rows_processed_per_sec	每秒处理的行数
db.sql_tokenized.stats.buffer_gets_per_sec	每秒缓冲区获取数
db.sql_tokenized.stats.physical_read_requests_per_sec	每秒物理读取次数
db.sql_tokenized.stats.physical_write_requests_per_sec	每秒物理写入次数
db.sql_tokenized.stats.total_sharable_mem_per_sec	每秒总可共享内存（以字节为单位）

指标	单位
db.sql_tokenized.stats.cpu_time_per_sec	每秒 CPU 时间（以毫秒为单位）

Oracle 的每次调用统计数据

以下指标提供了 Oracle SQL 语句的每次调用统计数据。

指标	单位
db.sql.stats.elapsed_time_per_exec	每次执行所用的时间（以毫秒为单位）
db.sql.stats.rows_processed_per_exec	每次执行处理的行数
db.sql.stats.buffer_gets_per_exec	每次执行的缓冲区获取数
db.sql.stats.physical_read_requests_per_exec	每次执行的物理读取次数
db.sql.stats.physical_write_requests_per_exec	每次执行的物理写入次数
db.sql.stats.total_sharable_mem_per_exec	每次执行的总可共享内存（以字节为单位）
db.sql.stats.cpu_time_per_exec	每次执行的 CPU 时间（以毫秒为单位）

以下指标提供了 Oracle SQL 摘要查询的每次调用统计数据。

指标	单位
db.sql_tokenized.stats.elapsed_time_per_exec	每次执行所用的时间（以毫秒为单位）
db.sql_tokenized.stats.rows_processed_per_exec	每次执行处理的行数
db.sql_tokenized.stats.buffer_gets_per_exec	每次执行的缓冲区获取数
db.sql_tokenized.stats.physical_read_requests_per_exec	每次执行的物理读取次数

指标	单位
db.sql_tokenized.stats.physical_write_requests_per_exec	每次执行的物理写入次数
db.sql_tokenized.stats.total_sharable_mem_per_exec	每次执行的总可共享内存（以字节为单位）
db.sql_tokenized.stats.cpu_time_per_exec	每次执行的 CPU 时间（以毫秒为单位）

Oracle 的主要统计数据

以下指标提供了 Oracle SQL 查询的主要统计数据。

指标	单位
db.sql.stats.executions	执行次数
db.sql.stats.elapsed_time	经过的时间（以毫秒为单位）
db.sql.stats.rows_processed	处理的行数
db.sql.stats.buffer_gets	缓冲区获得次数
db.sql.stats.physical_read_requests	物理读取
db.sql.stats.physical_write_requests	物理写入次数
db.sql.stats.total_sharable_mem	总可共享内存（以字节为单位）
db.sql.stats.cpu_time	CPU 时间（以毫秒为单位）

以下指标提供了 Oracle SQL 摘要查询的主要统计数据。

指标	单位
db.sql_tokenized.stats.executions	执行次数
db.sql_tokenized.stats.elapsed_time	经过的时间（以毫秒为单位）

指标	单位
db.sql_tokenized.stats.rows_processed	处理的行数
db.sql_tokenized.stats.buffer_gets	缓冲区获得次数
db.sql_tokenized.stats.physical_read_requests	物理读取
db.sql_tokenized.stats.physical_write_requests	物理写入次数
db.sql_tokenized.stats.total_sharable_mem	总可共享内存（以字节为单位）
db.sql_tokenized.stats.cpu_time	CPU 时间（以毫秒为单位）

Amazon RDS for SQL Server 的 SQL 统计数据

Amazon RDS for SQL Server 收集语句和摘要级别的 SQL 统计信息。在语句级别，ID 列表示 `sql_handle` 的值。在摘要级别，ID 列显示 `query_hash` 的值。

对于一些语句，SQL Server 会为 `query_hash` 返回 NULL 值。例如，ALTER INDEX、CHECKPOINT、UPDATE STATISTICS、COMMIT TRANSACTION、FETCH NEXT FROM Cursor 和一些 INSERT 语句、SELECT @<variable>、条件语句和可执行存储过程。在这种情况下，`sql_handle` 值显示为该语句的摘要级别的 ID。

主题

- SQL Server 的每秒统计数据
- SQL Server 的每次调用统计数据
- SQL Server 的主要统计数据

SQL Server 的每秒统计数据

以下指标提供了 SQL Server SQL 查询的每秒统计数据。

指标	单位
db.sql.stats.execution_count_per_sec	每秒执行次数
db.sql.stats.total_elapsed_time_per_sec	每秒总用时

指标	单位
db.sql.stats.total_rows_per_sec	每秒处理的总行数
db.sql.stats.total_logical_reads_per_sec	每秒逻辑读取总数
db.sql.stats.total_logical_writes_per_sec	每秒逻辑写入总数
db.sql.stats.total_physical_reads_per_sec	每秒物理读取总数
db.sql.stats.total_worker_time_per_sec	CPU 总时间（以毫秒为单位）

以下指标提供了 SQL Server SQL 摘要查询的每秒统计数据。

指标	单位
db.sql_tokenized.stats.execution_count_per_sec	每秒执行次数
db.sql_tokenized.stats.total_elapsed_time_per_sec	每秒总用时
db.sql_tokenized.stats.total_rows_per_sec	每秒处理的总行数
db.sql_tokenized.stats.total_logical_reads_per_sec	每秒逻辑读取总数
db.sql_tokenized.stats.total_logical_writes_per_sec	每秒逻辑写入总数
db.sql_tokenized.stats.total_physical_reads_per_sec	每秒物理读取总数
db.sql_tokenized.stats.total_worker_time_per_sec	CPU 总时间（以毫秒为单位）

SQL Server 的每次调用统计数据

以下指标提供了 SQL Server SQL 语句的每次调用统计数据。

指标	单位
db.sql.stats.total_elapsed_time_per_call	每次执行的总用时（以毫秒为单位）
db.sql.stats.total_rows_per_call	每次执行处理的总行数
db.sql.stats.total_logical_reads_per_call	每次执行的逻辑读取总数
db.sql.stats.total_logical_writes_per_call	每次执行的逻辑写入总数
db.sql.stats.total_physical_reads_per_call	每次执行的物理读取总数
db.sql.stats.total_worker_time_per_call	每次执行的总 CPU 时间（以毫秒为单位）

以下指标提供了 SQL Server SQL 摘要查询的每次调用统计数据。

指标	单位
db.sql_tokenized.stats.total_elapsed_time_per_call	每次执行的总用时
db.sql_tokenized.stats.total_rows_per_call	每次执行处理的总行数
db.sql_tokenized.stats.total_logical_reads_per_call	每次执行的逻辑读取总数
db.sql_tokenized.stats.total_logical_writes_per_call	每次执行的逻辑写入总数
db.sql_tokenized.stats.total_physical_reads_per_call	每次执行的物理读取总数
db.sql_tokenized.stats.total_worker_time_per_call	每次执行的总 CPU 时间（以毫秒为单位）

SQL Server 的主要统计数据

以下指标提供了 SQL Server SQL 查询的主要统计数据。

指标	单位
db.sql.stats.execution_count	执行次数
db.sql.stats.total_elapsed_time	经过的总时间（以毫秒为单位）
db.sql.stats.total_rows	处理的总行数
db.sql.stats.total_logical_reads	逻辑读取总次数
db.sql.stats.total_logical_writes	逻辑写入总次数
db.sql.stats.total_physical_reads	物理读取总次数
db.sql.stats.total_worker_time	CPU 总时间（以毫秒为单位）

以下指标提供了 SQL Server SQL 摘要查询的主要统计数据。

指标	单位
db.sql_tokenized.stats.execution_count	执行次数
db.sql_tokenized.stats.total_elapsed_time	经过的总时间（以毫秒为单位）
db.sql_tokenized.stats.total_rows	处理的总行数
db.sql_tokenized.stats.total_logical_reads	逻辑读取总次数
db.sql_tokenized.stats.total_logical_writes	逻辑写入总次数
db.sql_tokenized.stats.total_physical_reads	物理读取总次数
db.sql_tokenized.stats.total_worker_time	CPU 总时间（以毫秒为单位）

RDS PostgreSQL 的 SQL 统计数据

对于每个 SQL 调用和查询运行的每一秒，性能详情都会收集 SQL 统计数据。RDS for PostgreSQL 仅在摘要级别收集 SQL 统计数据。语句级别不显示任何统计数据。

接下来，您可以了解有关 RDS for PostgreSQL 的摘要级别统计数据的信息。

主题

- RDS PostgreSQL 的摘要统计数据
- RDS PostgreSQL 的每秒摘要统计数据
- RDS PostgreSQL 的每次调用摘要统计数据
- RDS PostgreSQL 的主要统计数据

RDS PostgreSQL 的摘要统计数据

要查看 SQL 摘要统计数据，RDS PostgreSQL 必须加载 `pg_stat_statements` 库。对于与 PostgreSQL 11 或更高版本兼容的 PostgreSQL 数据库实例，数据库默认加载此库。对于与 PostgreSQL 10 或更低版本兼容的 PostgreSQL 数据库实例，请手动启用此库。要手动启用该库，请在与数据库实例关联的数据库参数组中的 `pg_stat_statements` 中添加 `shared_preload_libraries`。然后重启数据库实例。有关更多信息，请参阅"Amazon RDS 的参数组"。

> **Note**
>
> 性能详情只能收集 `pg_stat_activity` 中未被截断的查询的统计数据。默认情况下，PostgreSQL 数据库会截断长度超过 1,024 字节的查询。要增加查询大小，请更改与数据库实例关联的数据库参数组中的 `track_activity_query_size` 参数。更改此参数后，需要重新启动数据库实例。

RDS PostgreSQL 的每秒摘要统计数据

以下 SQL 摘要统计数据可用于 PostgreSQL 数据库实例。

指标	单位
db.sql_tokenized.stats.calls_per_sec	每秒调用数
db.sql_tokenized.stats.rows_per_sec	每秒行数
db.sql_tokenized.stats.total_time_per_sec	每秒平均活动执行次数 (AAE)
db.sql_tokenized.stats.shared_blks_hit_per_sec	每秒块命中次数

指标	单位
db.sql_tokenized.stats.shared_blks_read_per_sec	每秒块读取次数
db.sql_tokenized.stats.shared_blks_dirtied_per_sec	每秒损坏的块数
db.sql_tokenized.stats.shared_blks_written_per_sec	每秒块写入次数
db.sql_tokenized.stats.local_blks_hit_per_sec	每秒本地块命中次数
db.sql_tokenized.stats.local_blks_read_per_sec	每秒本地块读取次数
db.sql_tokenized.stats.local_blks_dirtied_per_sec	每秒弄脏的本地块数
db.sql_tokenized.stats.local_blks_written_per_sec	每秒本地块写入次数
db.sql_tokenized.stats.temp_blks_written_per_sec	每秒临时写入次数
db.sql_tokenized.stats.temp_blks_read_per_sec	每秒临时读取次数
db.sql_tokenized.stats.blk_read_time_per_sec	每秒平均并发读取次数
db.sql_tokenized.stats.blk_write_time_per_sec	每秒平均并发写入次数

RDS PostgreSQL 的每次调用摘要统计数据

以下指标提供了 SQL 语句的每次调用统计数据。

指标	单位
db.sql_tokenized.stats.rows_per_call	每个调用的行数
db.sql_tokenized.stats.avg_latency_per_call	每个调用的平均延迟（以毫秒为单位）

指标	单位
db.sql_tokenized.stats.shared_blks_hit_per_call	每个调用的块命中次数
db.sql_tokenized.stats.shared_blks_read_per_call	每个调用的块读取次数
db.sql_tokenized.stats.shared_blks_written_per_call	每个调用的块写入次数
db.sql_tokenized.stats.shared_blks_dirtied_per_call	每个调用损坏的块数
db.sql_tokenized.stats.local_blks_hit_per_call	每个调用的本地块命中次数
db.sql_tokenized.stats.local_blks_read_per_call	每个调用的本地块读取次数
db.sql_tokenized.stats.local_blks_dirtied_per_call	每个调用弄脏的本地块数
db.sql_tokenized.stats.local_blks_written_per_call	每个调用的本地块写入次数
db.sql_tokenized.stats.temp_blks_written_per_call	每个调用的临时块写入次数
db.sql_tokenized.stats.temp_blks_read_per_call	每个调用的临时块读取次数
db.sql_tokenized.stats.blk_read_time_per_call	每个调用的读取时间（以毫秒为单位）
db.sql_tokenized.stats.blk_write_time_per_call	每个调用的写入时间（以毫秒为单位）

RDS PostgreSQL 的主要统计数据

以下 SQL 统计数据可用于 PostgreSQL 数据库实例。

指标	单位
db.sql_tokenized.stats.calls	Calls

指标	单位
db.sql_tokenized.stats.rows	行
db.sql_tokenized.stats.total_time	总时间（以毫秒为单位）
db.sql_tokenized.stats.shared_blks_hit	块命中次数
db.sql_tokenized.stats.shared_blks_read	块读取次数
db.sql_tokenized.stats.shared_blks_dirtied	弄脏的块数
db.sql_tokenized.stats.shared_blks_written	块写入次数
db.sql_tokenized.stats.local_blks_hit	本地块命中次数
db.sql_tokenized.stats.local_blks_read	本地块读取次数
db.sql_tokenized.stats.local_blks_dirtied	弄脏的本地块数
db.sql_tokenized.stats.local_blks_written	本地块写入次数
db.sql_tokenized.stats.temp_blks_written	临时写入次数
db.sql_tokenized.stats.temp_blks_read	临时读取次数
db.sql_tokenized.stats.blk_read_time	平均并发读取次数（以毫秒为单位）
db.sql_tokenized.stats.blk_write_time	平均并发写入次数（以毫秒为单位）

有关这些指标的更多信息，请参阅 PostgreSQL 文档中的 pg_stat_statements。

增强监控中的操作系统指标

Amazon RDS 为数据库实例运行于的操作系统 (OS) 实时提供指标。RDS 将增强监控中的指标传输到您的 Amazon CloudWatch Logs 账户。下列各表列出了使用 Amazon CloudWatch Logs 可获得的操作系统指标。

主题

- Microsoft SQL Server 的操作系统指标

Db2、MariaDB、MySQL、Oracle 和 PostgreSQL 的操作系统指标

组	指标	控制台名称	描述
General	engine	不适用	数据库实例的数据库引擎。
	instanceID	不适用	数据库实例标识符。
	instanceResourceID	不适用	数据库实例的不可变标识符，此标识符对于 Amazon 区域是唯一的，也用作日志流标识符。
	numVCPUs	不适用	数据库实例的虚拟 CPU 数量。
	timestamp	不适用	提取指标的时间。
	uptime	不适用	数据库实例处于活动状态的时间量。
	version	不适用	操作系统指标的流 JSON 格式版本。
cpuUtilization	guest	CPU Guest (CPU 访客)	来宾程序使用的 CPU 百分比。
	idle	CPU Idle (CPU 空闲)	CPU 空闲百分比。
	irq	CPU IRQ	软件中断使用的 CPU 百分比。
	nice	CPU Nice (CPU 良好)	以最低优先级运行的程序使用的 CPU 百分比。
	steal	CPU Steal (CPU 被盗用)	其他虚拟机使用的 CPU 百分比。

组	指标	控制台名称	描述
	system	CPU System (CPU 系统)	内核使用的 CPU 百分比。
	total	CPU Total (CPU 总计)	使用中的 CPU 百分比总计。此值包含 nice 值。
	user	CPU User (CPU 用户)	用户程序使用的 CPU 百分比。
	wait	CPU Wait (CPU 等待)	等待 I/O 访问时的未使用 CPU 百分比。
diskIO	avgQueueLen	Avg Queue Size (平均队列大小)	在 I/O 设备队列中等待的请求数。
	avgReqSz	Ave Request Size (平均请求大小)	平均请求大小 (以 KB 为单位)。
	await	Disk I/O Await (磁盘 I/O 等待)	响应请求所需的毫秒数，包括排队时间和服务时间。
	device	不适用	使用中的磁盘设备的标识符。
	readIOsPS	Read IO/s (读取 IO/秒)	每秒的读取操作数。
	readKb	Read Total (读取总计)	读取的总 KB 数。

组	指标	控制台名称	描述
	`readKbPS`	Read Kb/s (读取 Kb/秒)	每秒读取的 KB 数。
	`readLatency`	Read Latency (读取延迟)	从提交读取 I/O 请求到完成该请求所经过的时间（以毫秒为单位）。 此指标仅对 Amazon Aurora 可用。
	`readThroughput`	Read Throughput (读取吞吐量)	向数据库集群发出的请求要使用的网络吞吐量（以字节/秒为单位）。 此指标仅对 Amazon Aurora 可用。
	`rrqmPS`	Rrqms	每秒排队的合并读取请求数。
	`tps`	TPS	每秒的 I/O 事务数。
	`util`	Disk I/O Util (磁盘 I/O 利用率)	发出请求所经历的 CPU 时间的百分比。
	`writeIOsPS`	Write IO/s (写入 IO/秒)	每秒的写入操作数。
	`writeKb`	Write Total (写入总计)	写入的总 KB 数。
	`writeKbPS`	Write Kb/s (写入 Kb/秒)	每秒写入的 KB 数。
	`writeLatency`	Write Latency (写入延迟)	从提交写入 I/O 请求到完成该请求所经历的平均时间（以毫秒为单位）。 此指标仅对 Amazon Aurora 可用。

组	指标	控制台名称	描述
	writeThroughput	Write Throughput (写入吞吐量)	来自数据库集群的响应所使用的网络吞吐量（以字节/秒为单位）。 此指标仅对 Amazon Aurora 可用。
	wrqmPS	Wrqms	每秒排队的合并写入请求数。
physicalDeviceIO	avgQueueLen	Physical Devices Avg Queue Size (物理设备平均队列大小)	在 I/O 设备队列中等待的请求数。
	avgReqSz	Physical Devices Ave Request Size (物理设备平均请求大小)	平均请求大小 (以 KB 为单位)。
	await	Physical Devices Disk I/O Await (物理设备磁盘 I/O 等待)	响应请求所需的毫秒数，包括排队时间和服务时间。
	device	不适用	使用中的磁盘设备的标识符。
	readIOsPS	Physical Devices Read IO/s (物理设备读取 IO/秒)	每秒的读取操作数。

组	指标	控制台名称	描述
	readKb	Physical Devices Read Total (物理设备读取总数)	读取的总 KB 数。
	readKbPS	Physical Devices Read Kb/s (物理设备读取 Kb/秒)	每秒读取的 KB 数。
	rrqmPS	Physical Devices Rrqms (物理设备 Rrqms)	每秒排队的合并读取请求数。
	tps	Physical Devices TPS (物理设备 TPS)	每秒的 I/O 事务数。
	util	Physical Devices Disk I/O Util (物理设备磁盘 I/O 利用率)	发出请求所经历的 CPU 时间的百分比。
	writeIOsPS	Physical Devices Write IO/s (物理设备写入 IO/秒)	每秒的写入操作数。

组	指标	控制台名称	描述
	`writeKb`	Physical Devices Write Total (物理设备写入总数)	写入的总 KB 数。
	`writeKbPS`	Physical Devices Write Kb/s (物理设备写入 Kb/秒)	每秒写入的 KB 数。
	`wrqmPS`	Physical Devices Wrqms (物理设备 Wrqms)	每秒排队的合并写入请求数。
`fileSys`	`maxFiles`	Max Inodes (最大 Inode 数)	可为文件系统创建的文件的最大数量。
	`mountPoint`	不适用	文件系统的路径。
	`name`	不适用	文件系统的名称。
	`total`	Total Filesystem (文件系统总计)	文件系统的可用磁盘空间总量 (以 KB 为单位)。

组	指标	控制台名称	描述
	used	Used Filesystem (已使用的文件系统)	文件系统中的文件所用的磁盘空间量 (以 KB 为单位)。
	usedFilePercent	Used Inodes (已使用 Inode)	使用中的可用文件百分比。
	usedFiles	已使用百分比	文件系统中的文件数。
	usedPercent	Used Filesystem (已使用的文件系统)	使用中的文件系统磁盘空间百分比。
loadAverageMinute	fifteen	Load Avg 15 min (负载平均 15 分钟)	过去 15 分钟内请求 CPU 时间的进程数。
	five	Load Avg 5 min (负载平均 5 分钟)	过去 5 分钟内请求 CPU 时间的进程数。
	one	Load Avg 1 min (负载平均 1 分钟)	过去 1 分钟内请求 CPU 时间的进程数。
memory	active	Active Memory (活动内存)	已分配的内存量 (以 KB 为单位)。

组	指标	控制台名称	描述
	`buffers`	Buffered Memory (缓冲内存)	在写入存储设备前用于缓存 I/O 请求的内存量 (以 KB 为单位)。
	`cached`	Cached Memory (缓存内存)	用于缓存基于文件系统的输入/输出的内存量。
	`dirty`	Dirty Memory (脏内存)	RAM 中已修改但未写入存储中的相关数据块的内存页面大小 (以 KB 为单位)。
	`free`	Free Memory (空闲内存)	未分配的内存量 (以 KB 为单位)。
	`hugePagesFree`	Huge Pages Free (可用大页)	可用大页数。大页是 Linux 内核的一项功能。
	`hugePagesRsvd`	Huge Pages Rsvd (预留大页)	已提交大页数。
	`hugePagesSize`	Huge Pages Size (大页大小)	每个大页单位的大小 (以 KB 为单位)。
	`hugePagesSurp`	Huge Pages Surp (剩余大页)	剩余可用大页总数。

组	指标	控制台名称	描述
	hugePages Total	Huge Pages Total (大页总计)	大页总数。
	inactive	Inactive Memory (非活动内存)	最不常用内存页面大小 (以 KB 为单位)。
	mapped	Mapped Memory (映射的内存)	在进程地址空间中有内存映射的文件系统内容的总大小 (以 KB 为单位)。
	pageTables	Page Tables (页表)	页表使用的内存量 (以 KB 为单位)。
	slab	Slab Memory (Slab 内存)	可重用内核数据结构大小 (以 KB 为单位)。
	total	总内存	内存总量 (以 KB 为单位)。
	writeback	Writeback Memory (回写内存)	RAM 中仍在写入备份存储的脏页大小 (以 KB 为单位)。
network	interface	不适用	数据库实例现用网络接口的标识符。
	rx	RX	每秒接收的字节数。
	tx	TX	每秒上传的字节数。

组	指标	控制台名称	描述
processList	cpuUsedPc	CPU % (CPU 百分比)	进程使用的 CPU 百分比。
	id	不适用	进程的标识符。
	memoryUsedPc	MEM%	进程使用的内存百分比。
	name	不适用	进程的名称。
	parentID	不适用	进程的父进程的进程标识符。
	rss	RES	分配给进程的 RAM 量 (以 KB 为单位)。
	tgid	不适用	线程组标识符，是表示线程所属进程的 ID 的数字。此标识符用于将同一进程内的线程分入一组。
	vss	VIRT	分配给进程的虚拟内存量 (以 KB 为单位)。
swap	total	Swap (交换)	可用的交换内存量（以 KB 为单位）。
	in	Swaps in (换入)	从磁盘换入的内存量（以 KB 为单位）。
	out	Swaps out (换出)	换出到磁盘的内存量（以 KB 为单位）。
	free	Free Swap (可用交换)	空闲交换内存量（以 KB 为单位）。
	cached	Committed Swap (已提交的交换)	用作缓存内存的交换内存量 (以 KB 为单位)。

组	指标	控制台名称	描述
tasks	blocked	Tasks Blocked (阻止的任务)	已阻止的任务的数量。
	running	Tasks Running (正在运行的任务)	正在运行的任务的数量。
	sleeping	Tasks Sleeping (正在休眠的任务)	正在睡眠的任务的数量。
	stopped	Tasks Stopped (已停止的任务)	已停止的任务的数量。
	total	Tasks Total (任务总计)	任务总数。
	zombie	Tasks Zombie (停滞的任务)	有活动父任务的不活动子任务的数量。

Microsoft SQL Server 的操作系统指标

组	指标	控制台名称	描述
General	engine	不适用	数据库实例的数据库引擎。

组	指标	控制台名称	描述
	instanceID	不适用	数据库实例标识符。
	instanceResourceID	不适用	数据库实例的不可变标识符，此标识符对于 Amazon 区域是唯一的，也用作日志流标识符。
	numVCPUs	不适用	数据库实例的虚拟 CPU 数量。
	timestamp	不适用	提取指标的时间。
	uptime	不适用	数据库实例处于活动状态的时间量。
	version	不适用	操作系统指标的流 JSON 格式版本。
cpuUtilization	idle	CPU Idle (CPU 空闲)	CPU 空闲百分比。
	kern	CPU Kernel (CPU 内核)	内核使用的 CPU 百分比。
	user	CPU User (CPU 用户)	用户程序使用的 CPU 百分比。
disks	name	不适用	磁盘的标识符。
	totalKb	Total Disk Space (磁盘空间总计)	磁盘的总空间 (KB)。
	usedKb	Used Disk Space (已用磁盘空间)	磁盘上使用的空间量 (KB)。
	usedPc	Used Disk Space % (已用磁盘空间百分比)	磁盘上使用的空间百分比。

组	指标	控制台名称	描述
	availKb	Available Disk Space (可用磁盘空间)	磁盘上的可用空间 (KB)。
	availPc	Available Disk Space % (可用磁盘空间百分比)	磁盘上可用空间的百分比。
	rdCountPS	Reads/s (读取数/秒)	每秒的读取操作数
	rdBytesPS	Read Kb/s (读取 Kb/秒)	每秒读取的字节数。
	wrCountPS	Write IO/s (写入 IO/秒)	每秒的写入操作数。
	wrBytesPS	Write Kb/s (写入 Kb/秒)	每秒写入的字节数。
memory	commitTotKb	Commit Total (提交总计)	正在使用的页面文件备份的虚拟地址空间量，即当前内存使用。此值包含主内存 (RAM) 和磁盘 (页面文件)。
	commitLimitKb	Maximum Commit (最大提交量)	commitTotKb 指标的最大可取值。此值为当前页面文件大小加上可用于可分页内容的物理内存之和，不包含已分配给不可分页区域的 RAM。
	commitPeakKb	Commit Peak (提交峰值)	自上次启动操作系统以来 commitTotKb 指标的最大值。
	kernTotKb	Total Kernel Memory (内核内存总量)	分页和未分页内核池中的内存之和 (KB)。

组	指标	控制台名称	描述
	kernPagedKb	Paged Kernel Memory (分页内核内存)	分页内核池中的内存量 (KB)。
	kernNonpagedKb	Nonpaged Kerenel Memory (未分页内核内存)	未分页内核池中的内存量 (KB)。
	pageSize	页面大小	页面的大小 (字节)。
	physTotKb	总内存	物理内存量 (KB)。
	physAvailKb	可用内存	可用物理内存量 (KB)。
	sqlServerTotKb	SQL Server Total Memory (SQL Server 内存总量)	提交给 SQL Server 的内存量 (KB)。
	sysCacheKb	System Cache (系统缓存)	系统缓存容量 (KB)。
network	interface	不适用	数据库实例现用网络接口的标识符。
	rdBytesPS	Network Read Kb/s (网络读取 Kb/秒)	每秒接收的字节数。
	wrBytesPS	Network Write Kb/s (网络写入 Kb/秒)	每秒发送的字节数。

组	指标	控制台名称	描述
processList	cpuUsedPc	已使用百分比	进程使用的 CPU 百分比。
	memUsedPc	MEM%	进程使用的总内存百分比。
	name	不适用	进程的名称。
	pid	不适用	进程的标识符。Amazon RDS 拥有的进程不存在此值。
	ppid	不适用	此进程的父进程的进程标识符。仅子进程存在此值。
	tid	不适用	线程标识符。仅线程存在此值。拥有的进程可使用 pid 值标识。
	workingSetKb	不适用	私有工作集中的内存量加上进程正在使用且可与其他进程共享的内存量 (KB)。
	workingSetPrivKb	不适用	进程正在使用但无法与其他进程共享的内存量 (KB)。
	workingSetShareableKb	不适用	进程正在使用且可与其他进程共享的内存量 (KB)。
	virtKb	不适用	进程正在使用的虚拟地址空间量 (KB)。虚拟地址空间的使用不一定意味着磁盘或主内存页的相应使用。
system	handles	Handles (句柄)	系统使用的句柄数量。
	processes	进程	系统上运行的进程数量。
	threads	Threads (线程)	系统上运行的线程数量。

监控 Amazon RDS 数据库实例中的事件、日志和流

监控您的 Amazon RDS 数据库和其他 Amazon 解决方案时，您的目标是维持以下各项：

- 可靠性
- 可用性
- 性能
- 安全性

监控 Amazon RDS 实例中的指标 说明如何使用指标监控实例。完整的解决方案还必须监控数据库事件、日志文件和活动流。Amazon 为您提供以下监控工具：

- Amazon EventBridge 是一种无服务器事件总线服务，可以轻松地将应用程序与来自各种来源的数据相连接。EventBridge 可以从您自己的应用程序、软件即服务（SaaS）应用程序和 Amazon 服务传输实时数据流。EventBridge 将该数据路由到诸如 Amazon Lambda 之类的目标。这样，您就可以监控服务中发生的事件，并构建事件驱动的架构。有关更多信息，请参阅 Amazon EventBridge 用户指南。
- Amazon CloudWatch Logs 提供了一种方法，让您可以监控、存储和访问来自 Amazon RDS 实例、Amazon CloudTrail 和其他来源的日志文件。Amazon CloudWatch Logs 可以监控日志文件中的信息，并在达到特定阈值时通知您。您还可以在高持久性存储中检索您的日志数据。有关更多信息，请参阅 Amazon CloudWatch Logs 用户指南。
- Amazon CloudTrail 捕获由某个 Amazon Web Services 账户 发出或代表该账户发出的 API 调用和相关事件。CloudTrail 将日志文件传送到您指定的 Amazon S3 桶。您可以标识哪些用户和账户调用了 Amazon、发出调用的源 IP 地址以及调用的发生时间。有关更多信息，请参阅 Amazon CloudTrail《用户指南》。
- 数据库活动流是一项 Amazon RDS 功能，它提供数据库实例中近乎实时的活动流。Amazon RDS 会将活动推送到 Amazon Kinesis 数据流。系统将自动创建 Kinesis 流。在 Kinesis 中，您可以配置 Amazon 服务（如 Amazon Data Firehose）和 Amazon Lambda 来使用 Kinesis 流并存储数据。

主题

- 使用数据库活动流监控 Amazon RDS
- 使用 Amazon GuardDuty RDS 保护功能监控威胁

在 Amazon RDS 控制台中查看日志、事件和流

Amazon RDS 与 Amazon Web Services 服务 集成，以在 RDS 控制台中显示关于日志、事件和数据库活动流的信息。

RDS 数据库实例的 Logs & events（日志和事件）选项卡显示以下信息：

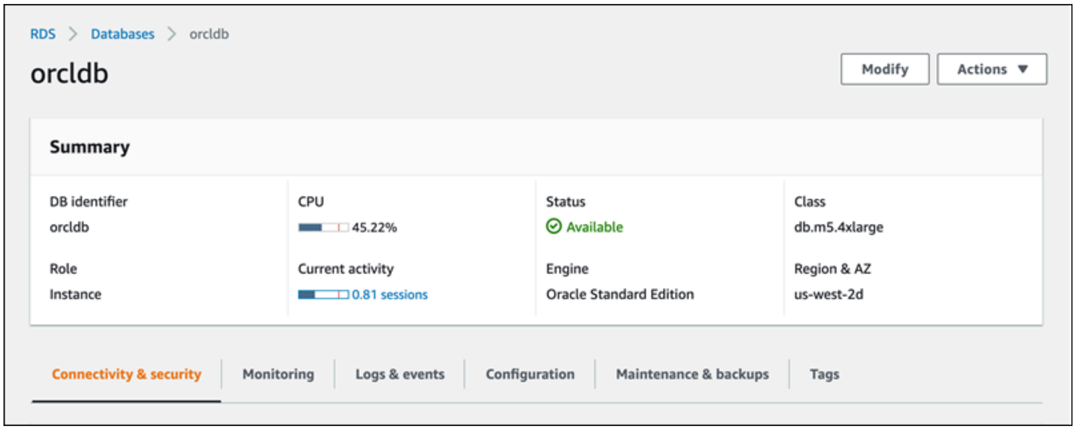
- Amazon CloudWatch 告警 - 显示为 的数据库实例配置的任何指标告警。如果尚未配置告警，您可以在 RDS 控制台中创建告警。有关更多信息，请参阅使用 Amazon CloudWatch 监控 Amazon RDS 指标。
- Recent events（近期事件）- 显示 RDS 数据库实例事件（环境变更）的摘要。有关更多信息，请参阅 查看 Amazon RDS 事件。
- Logs（日志）- 显示中数据库实例生成的数据库日志文件。有关更多信息，请参阅 监控 Amazon RDS 日志文件。

Configuration（配置）选项卡显示关于数据库活动流的信息。

在 RDS 控制台中查看 数据库实例的日志、事件和流

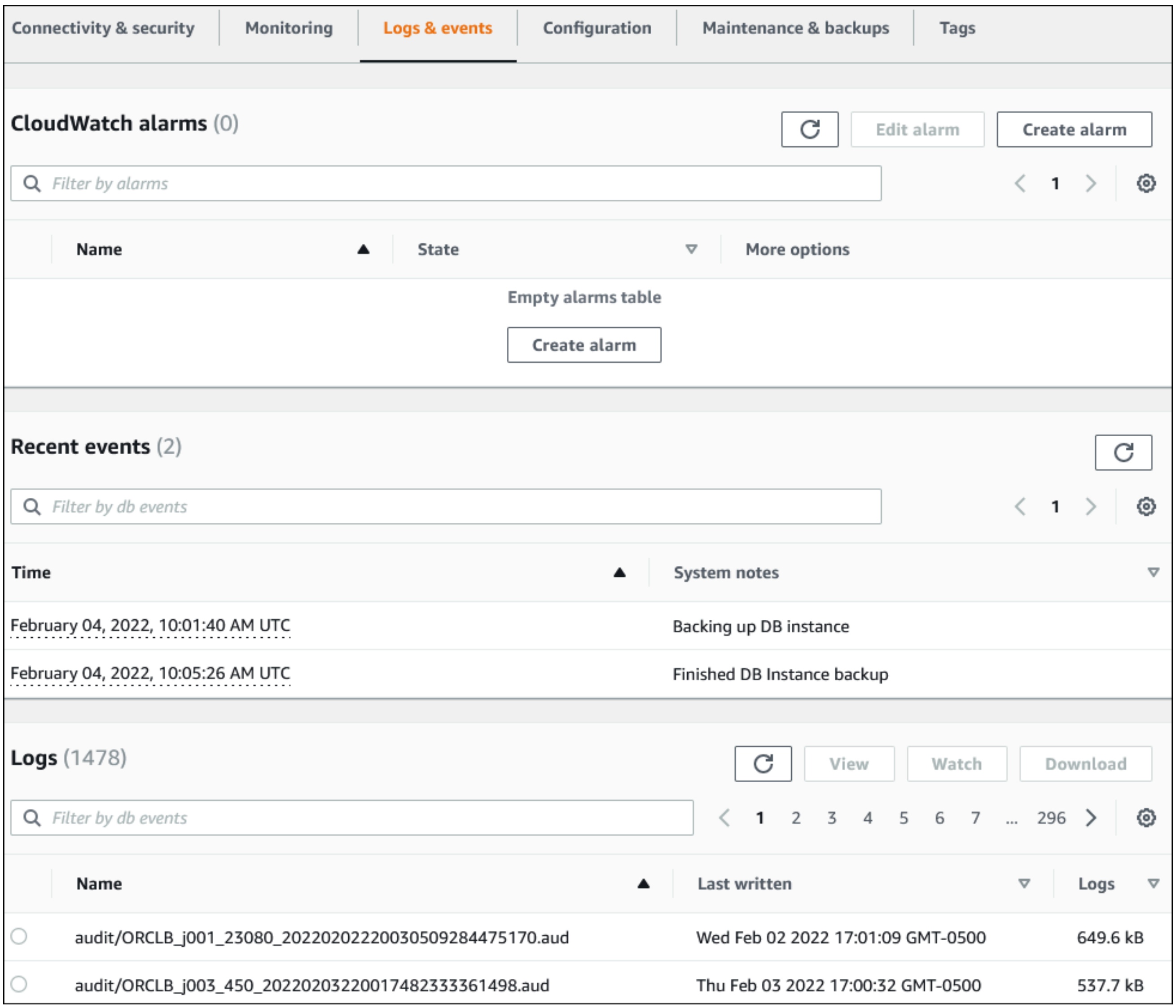
1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases（数据库）。
3. 选择要监控的 数据库实例的名称。

   随后会显示数据库页面。以下示例显示名为 `orclb` 的 Oracle 数据库。



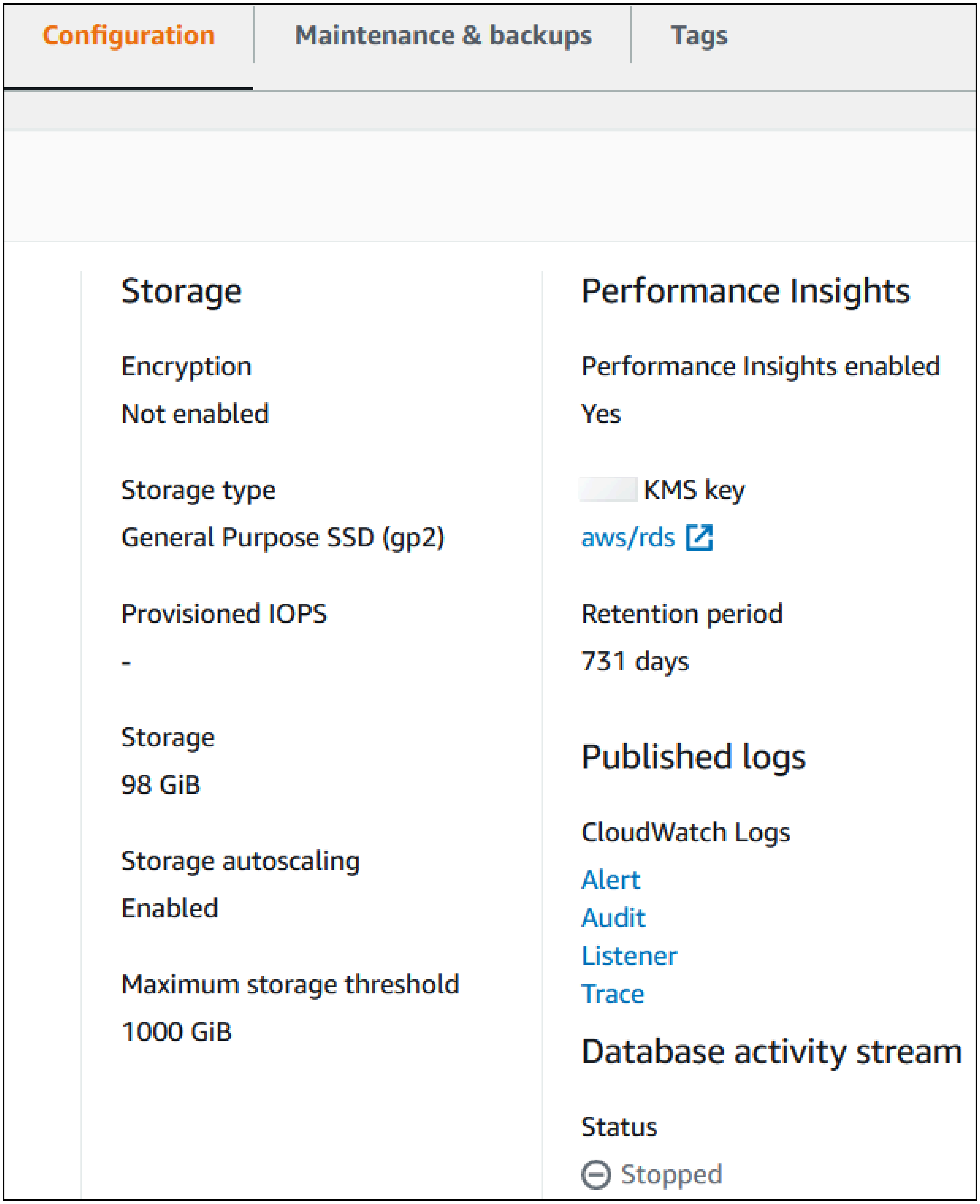
4. 选择日志和事件。

此时将显示 Logs & events（日志和事件）部分。



5. 选择配置。

   以下示例显示数据库实例的数据库活动流的状态。



监控 Amazon RDS 事件

事件表示环境中的更改。这可以是 Amazon 环境、SaaS 合作伙伴服务或应用程序，或自定义应用程序或服务。有关 RDS 事件的说明，请参阅 Amazon RDS 事件类别和事件消息。

主题

Amazon RDS 事件概述

RDS 事件表示 Amazon RDS 环境中的更改。例如，当数据库实例的状态从挂起变为正在运行时，Amazon RDS 将生成事件。Amazon RDS 将事件近乎实时地传输到 EventBridge。

> **Note**
>
> Amazon RDS 尽最大努力发出事件。我们建议您避免编写取决于通知事件的顺序或存在的程序，因为它们可能是乱序或缺失的。

Amazon RDS 记录与以下资源相关的事件：

- 数据库实例

  有关数据库实例事件的列表，请参阅 数据库实例事件。
- 数据库参数组

  有关数据库参数组事件的列表，请参阅 数据库参数组事件。
- 数据库安全组

  有关数据库安全组事件的列表，请参阅 数据库安全组事件。
- 数据库快照

  有关数据库快照事件的列表，请参阅 数据库快照事件。

- RDS Proxy 事件

  有关 RDS Proxy 事件的列表，请参阅 RDS Proxy 事件。
- 蓝绿部署事件

  有关蓝绿部署事件的列表，请参阅 蓝绿部署事件。

这些信息包含：

- 事件的日期和时间
- 事件的源名称和源类型
- 与事件关联的消息
- 事件通知包括发送消息时的标签，可能不反映事件发生时的标签。

查看 Amazon RDS 事件

您可以检索 Amazon RDS 资源的以下事件信息：

- 资源名称
- 资源类型
- 事件的时间
- 事件的消息摘要

您可以访问 Amazon Web Services Management Console的以下各部分中的事件：

- 事件选项卡，显示过去 24 小时的事件。
- 数据库选项卡中日志和事件部分的近期事件表，可以显示多达过去 2 周的事件。

您还可以通过使用 describe-events Amazon CLI 命令或 DescribeEvents RDS API 操作来检索事件。如果您使用 Amazon CLI 或 RDS API 查看事件，则可检索过去长达 14 天内的事件。

> **Note**
>
> 如果需要将事件存储更长的时间段，则可以将 Amazon RDS 事件发送到 EventBridge。有关更多信息，请参阅 创建对 Amazon RDS 事件触发的规则

有关 Amazon RDS 事件的说明，请参阅 Amazon RDS 事件类别和事件消息。

要访问有关使用 Amazon CloudTrail 的事件的详细信息，包括请求参数，请参阅 CloudTrail 事件。

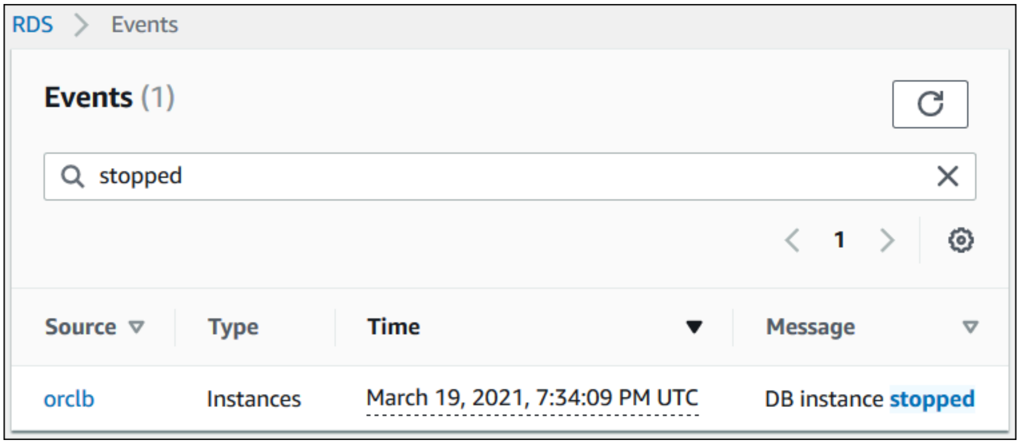
控制台

查看过去 24 小时的所有 Amazon RDS 事件

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Events (事件)。

   列表中显示可用的事件。
3. （可选）输入搜索词以筛选结果。

以下示例显示按字符 **stopped** 筛选的事件列表。



Amazon CLI

要查看过去一小时内生成的所有事件，请调用 describe-events，不带参数。

```
aws rds describe-events
```

以下示例输出显示数据库实例已停止。

```
{
    "Events": [
        {
            "EventCategories": [
                "notification"
            ],
            "SourceType": "db-instance",
            "SourceArn": "arn:aws:rds:us-east-1:123456789012:db:testinst",
            "Date": "2022-04-22T21:31:00.681Z",
            "Message": "DB instance stopped",
            "SourceIdentifier": "testinst"
        }
    ]
}
```

要查看过去 10080 分钟（7 天）的所有 Amazon RDS 事件，请调用 describe-events Amazon CLI 命令并将 `--duration` 参数设置为 `10080`。

```
aws rds describe-events --duration 10080
```

以下示例显示数据库实例 *test-instance* 在指定时间范围内的事件。

```
aws rds describe-events \
    --source-identifier test-instance \
    --source-type db-instance \
    --start-time 2022-03-13T22:00Z \
    --end-time 2022-03-13T23:59Z
```

以下示例输出显示备份的状态。


```
{
    "Events": [
        {
            "SourceType": "db-instance",
            "SourceIdentifier": "test-instance",
            "EventCategories": [
                "backup"
            ],
            "Message": "Backing up DB instance",
            "Date": "2022-03-13T23:09:23.983Z",
            "SourceArn": "arn:aws:rds:us-east-1:123456789012:db:test-instance"
        },
        {
            "SourceType": "db-instance",
            "SourceIdentifier": "test-instance",
            "EventCategories": [
                "backup"
            ],
            "Message": "Finished DB Instance backup",
            "Date": "2022-03-13T23:15:13.049Z",
            "SourceArn": "arn:aws:rds:us-east-1:123456789012:db:test-instance"
        }
    ]
}
```


API

通过调用 DescribeEvents RDS API 操作并将 `Duration` 参数设置为 `20160`，您可查看过去 14 天内的所有 Amazon RDS 实例事件。

使用 Amazon RDS 事件通知

Amazon RDS 使用 Amazon Simple Notification Service (Amazon SNS) 在发生 Amazon RDS 事件时提供通知。这些通知可以采用 Amazon 区域 Amazon SNS 支持的任何通知形式，例如电子邮件、文本消息或对 HTTP 终端节点的调用。

主题

Amazon RDS 事件通知概述

Amazon RDS 将事件分组为您可以订阅的类型，以便您在出现该类事件时收取通知。

主题

符合事件订阅条件的 RDS 资源

您可以订阅以下资源的事件类别：

- 数据库实例

- 数据库快照
- 数据库参数组
- 数据库安全组
- RDS 代理
- 自定义引擎版本

例如，如果您订阅给定数据库实例的备份类别，那么无论何时出现影响该数据库实例的备份相关事件，您都将收到通知。如果您订阅对于数据库实例的配置更改类别，则会在数据库实例出现更改时收到通知。您还将在事件通知订阅更改时收到通知。

您可能需要创建多个不同的订阅。例如，您可能创建一个订阅以接收所有数据库实例的所有事件通知，并创建另一个订阅，其中仅包括数据库实例子集的严重事件。对于第二个订阅，请在筛选条件中指定一个或多个数据库实例。

订阅 Amazon RDS 事件通知的基本流程

订阅 Amazon RDS 事件通知的流程如下：

1. 您使用 Amazon RDS 控制台、Amazon CLI 或者 API 创建 Amazon RDS 事件通知订阅。

   Amazon RDS 使用 Amazon SNS 主题的 ARN 标识每个订阅。Amazon RDS 控制台在您创建订阅时为您创建 ARN。使用 Amazon SNS 控制台、Amazon CLI 或 Amazon SNS API 创建 ARN。
2. Amazon RDS 发送批准电子邮件或者 SMS 消息给您在订阅时提交的地址。
3. 您可以通过选择所收到的通知中的链接来确认您的订阅。
4. Amazon RDS 控制台更新 My Event Subscriptions（我的事件订阅）部分，其中包含您的订阅状态。
5. Amazon RDS 开始将通知发送到您在创建订阅时提供的地址。

要了解使用 Amazon SNS 时的 Identity of Access Management，请查看 Amazon Simple Notification Service 开发人员指南中的 Amazon SNS 中的 Identity of Access Management。

您可以使用处理 Amazon Lambda 来自数据库实例的事件通知。有关更多信息，请参阅 Amazon Lambda 开发人员指南中的将 Amazon Lambda 与 Amazon RDS 结合使用。

发送 RDS 事件通知

Amazon RDS 会将通知发送到您在创建订阅时提供的地址。通知可以包含消息属性，以提供有关消息的结构化元数据。有关消息属性的更多信息，请参阅Amazon RDS 事件类别和事件消息。

事件通知的传递可能需要长达五分钟。

> **Important**
>
> Amazon RDS 不保证在事件流中发送的事件的顺序。事件顺序可能会发生变化。

当 Amazon SNS 向已经订阅的 HTTP 或 HTTPS 端点发送一条通知时，发至端点的 POST 消息具有包含 JSON 格式文档的消息正文。有关更多信息，请参阅 Amazon Simple Notification Service 开发人员指南 中的 Amazon SNS 消息和 JSON 格式。

您可以将 SNS 配置为通过短信通知。有关更多信息，请参阅 Amazon Simple Notification Service 开发人员指南中的移动文本消息 (SMS)。

要在不删除订阅的情况下关闭通知，请在 Amazon RDS 控制台中为已启用选择否。或者，可以使用 Amazon CLI 或 Amazon RDS API 将 `Enabled` 参数设置为 `false`。

Amazon RDS 事件通知的计费

Amazon RDS 事件通知的计费是通过 Amazon SNS 执行的。Amazon SNS 费用在使用事件通知时适用。有关 Amazon SNS 计费的更多信息，请参阅 Amazon Simple Notification Service 定价。

使用 Amazon EventBridge 的 Amazon RDS 事件的示例

以下示例说明了 JSON 格式的不同类型的 Amazon RDS 事件。有关向您说明如何以 JSON 格式捕获和查看事件的教程，请参阅教程：使用 Amazon EventBridge 记录数据库实例的状态更改。

主题

- 数据库实例事件的示例
- 数据库参数组事件的示例
- 数据库快照事件的示例

数据库实例事件的示例

以下是采用 JSON 格式的数据库实例事件的示例。该事件显示，RDS 为名为 `my-db-instance` 的实例执行了多可用区故障转移。事件 ID 为 RDS-EVENT-0049。

```
{
  "version": "0",
```

```
  "id": "68f6e973-1a0c-d37b-f2f2-94a7f62ffd4e",
  "detail-type": "RDS DB Instance Event",
  "source": "aws.rds",
  "account": "123456789012",
  "time": "2018-09-27T22:36:43Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:rds:us-east-1:123456789012:db:my-db-instance"
  ],
  "detail": {
    "EventCategories": [
      "failover"
    ],
    "SourceType": "DB_INSTANCE",
    "SourceArn": "arn:aws:rds:us-east-1:123456789012:db:my-db-instance",
    "Date": "2018-09-27T22:36:43.292Z",
    "Message": "A Multi-AZ failover has completed.",
    "SourceIdentifier": "my-db-instance",
    "EventID": "RDS-EVENT-0049"
  }
}
```


数据库参数组事件的示例

以下是采用 JSON 格式的数据库参数组事件的示例。该事件显示参数 `time_zone` 已在参数组 `my-db-param-group` 中更新。事件 ID 为 RDS-EVENT-0037。


```
{
  "version": "0",
  "id": "844e2571-85d4-695f-b930-0153b71dcb42",
  "detail-type": "RDS DB Parameter Group Event",
  "source": "aws.rds",
  "account": "123456789012",
  "time": "2018-10-06T12:26:13Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:rds:us-east-1:123456789012:pg:my-db-param-group"
  ],
  "detail": {
    "EventCategories": [
      "configuration change"
    ],
    "SourceType": "DB_PARAM",
```

```
    "SourceArn": "arn:aws:rds:us-east-1:123456789012:pg:my-db-param-group",
    "Date": "2018-10-06T12:26:13.882Z",
    "Message": "Updated parameter time_zone to UTC with apply method immediate",
    "SourceIdentifier": "my-db-param-group",
    "EventID": "RDS-EVENT-0037"
  }
}
```

数据库快照事件的示例

以下是采用 JSON 格式的数据库快照事件的示例。该事件显示删除名 `my-db-snapshot` 为的快照。事件 ID 为 RDS-EVENT-0041。

```
{
  "version": "0",
  "id": "844e2571-85d4-695f-b930-0153b71dcb42",
  "detail-type": "RDS DB Snapshot Event",
  "source": "aws.rds",
  "account": "123456789012",
  "time": "2018-10-06T12:26:13Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:rds:us-east-1:123456789012:snapshot:rds:my-db-snapshot"
  ],
  "detail": {
    "EventCategories": [
      "deletion"
    ],
    "SourceType": "SNAPSHOT",
    "SourceArn": "arn:aws:rds:us-east-1:123456789012:snapshot:rds:my-db-snapshot",
    "Date": "2018-10-06T12:26:13.882Z",
    "Message": "Deleted manual snapshot",
    "SourceIdentifier": "my-db-snapshot",
    "EventID": "RDS-EVENT-0041"
  }
}
```

授予向 Amazon SNS 主题发布通知的权限。

要授予 Amazon RDS 将通知发布到 Amazon Simple Notification Service (Amazon SNS) 主题的权限，请将 Amazon Identity and Access Management (IAM) 策略附加到目标主题。有关权限的更多信息，请参阅《Amazon Simple Notification Service 开发人员指南》中的 Amazon Simple Notification Service 访问控制示例案例。

原定设置情况下，Amazon SNS 主题具有一项策略，它允许同一账户中的所有 Amazon RDS 资源向其发布通知。您可以附加自定义策略以允许跨账户通知，或限制对某些资源的访问。

以下是您附加到目标 Amazon SNS 主题的 IAM 策略示例。它将主题限制为名称与指定前缀匹配的数据库实例。要使用此策略，请指定以下值：

- `Resource` – Amazon SNS 主题的 Amazon 资源名称 (ARN)
- `SourceARN` – 您的 RDS 资源 ARN
- `SourceAccount` – 您的 Amazon Web Services 账户 ID

有关资源类型及其 ARN 的列表，请参阅《服务授权参考》中的 Amazon RDS 定义的资源。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "events.rds.amazonaws.com"
      },
      "Action": [
        "sns:Publish"
      ],
      "Resource": "arn:aws:sns:us-east-1:123456789012:topic_name",
      "Condition": {
        "ArnLike": {
          "aws:SourceArn": "arn:aws:rds:us-east-1:123456789012:db:prefix-*"
        },
        "StringEquals": {
          "aws:SourceAccount": "123456789012"
        }
      }
    }
  ]
```

```
}
```

订阅 Amazon RDS 事件通知

最简单的订阅创建方法是使用 RDS 控制台。如果您选择使用 CLI 或 API 创建事件通知，则必须创建 Amazon Simple Notification Service 主题并订阅有关 Amazon SNS 控制台或 Amazon SNS API 的主题。您还必须保留该主题的 Amazon Resource Name (ARN)，因为在提交 CLI 命令或者 API 操作时会用到。有关创建和订阅 SNS 主题的信息，请参阅 Amazon Simple Notification Service 开发人员指南中的 Amazon SNS 入门。

您可以指定希望收取其通知的源类型以及触发该事件的 Amazon RDS 源：

Source type（源类型）

源类型。例如，Source type（源类型）可能是 Instances（实例）。必须选择一种源类型。

`Resources` to include（要包含的资源）

生成事件的 Amazon RDS 资源。例如，您可以选择 Select specific instances（选择特定实例），然后选择 myDBInstance1。

下表说明了指定或不指定 **`Resources`** to include（要包含的资源）时的结果。

要包含的资源	描述	示例
指定	RDS 仅向您通知指定资源的所有事件。	如果 Source type（源类型）为 Instances（实例），资源为 myDBInstance1，则 RDS 仅向您通知 myDBInstance1 的所有事件。
未指定	RDS 会向您通知所有 Amazon RDS 资源的指定源类型的事件。	如果 Source type（源类型）是 Instances（实例），RDS 会向您通知您的账户中所有与实例相关的事件。

原定设置情况下，Amazon SNS 主题订阅用户会收到发布到该主题的每条消息。要仅接收一部分消息，订阅用户必须将筛选策略分配给主题订阅。有关更多信息，请参阅《Amazon Simple Notification Service 开发人员指南》中的 Amazon SNS 消息筛选。

控制台

订阅 RDS 事件通知

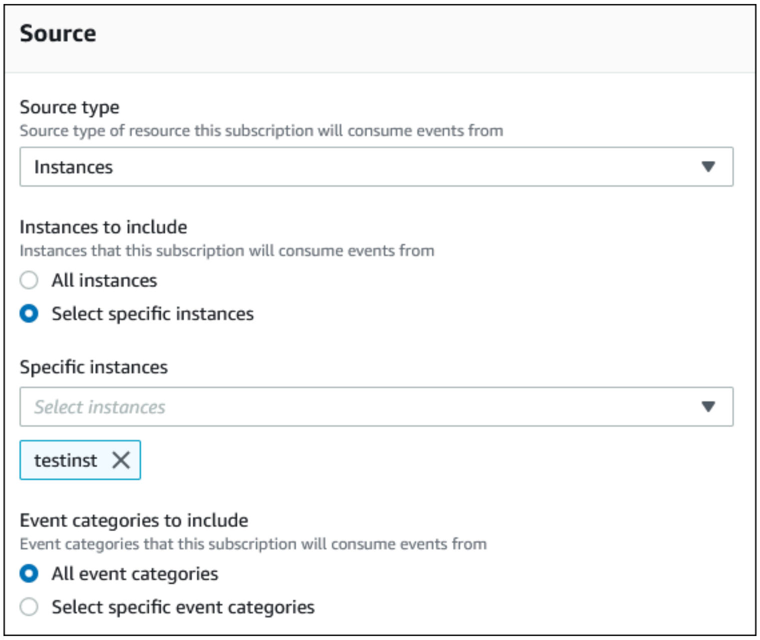
1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择事件订阅。
3. 在事件订阅窗格中，选择创建事件订阅。
4. 输入您的订阅详情，如下所示：

   a. 对于名称，输入事件通知订阅的名称。
   b. 对于 Send notifications to（发送通知到），执行以下操作之一：

      • 选择 New email topic（新建电子邮件主题）。输入电子邮件主题的名称和收件人列表。我们建议您将事件订阅配置为与主要账户联系人相同的电子邮件地址。建议、服务事件和个人健康信息是通过不同的渠道发送的。订阅同一个电子邮件地址可确保将所有邮件合并到一个位置。
      • 选择 Amazon Resource Name (ARN) [Amazon 资源名称（ARN）]。然后，为 Amazon SNS 主题选择现有的 Amazon SNS ARN。

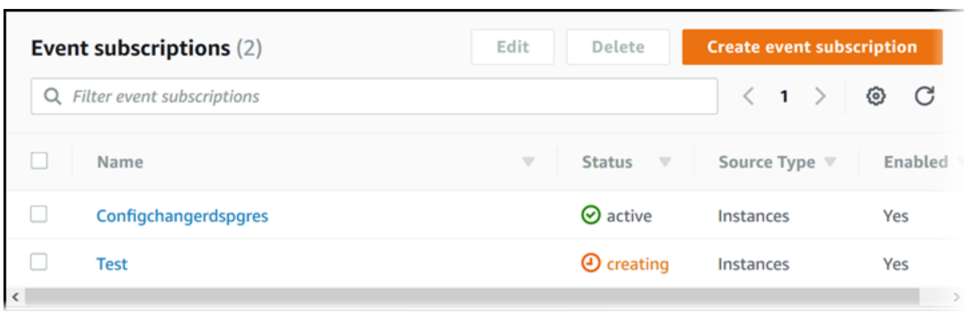
        如果您想使用已启用服务器端加密 (SSE) 的主题，请向 Amazon RDS 授予访问 Amazon KMS key 的必需权限。有关更多信息，请参阅《Amazon Simple Notification Service 开发人员指南》中的实现 Amazon 服务中的事件源与加密主题之间的兼容性。
   c. 对于源类型，请选择一种源类型。例如，选择 Instances（实例）或 Parameter groups（参数组）。
   d. 选择要接收事件通知的事件类别和资源。

      以下示例为名为 `testinst` 的数据库实例配置事件通知。



e. 选择创建。

Amazon RDS 控制台会表明正在创建订阅。



Amazon CLI

要订阅 RDS 事件通知，请使用 Amazon CLI `create-event-subscription` 命令。包括以下必需参数：

- `--subscription-name`
- `--sns-topic-arn`

Example

对于 Linux、macOS 或 Unix：

```
aws rds create-event-subscription \
```

```
    --subscription-name myeventsubscription \
    --sns-topic-arn arn:aws:sns:us-east-1:123456789012:myawsuser-RDS \
    --enabled
```

对于 Windows：

```
aws rds create-event-subscription ^
    --subscription-name myeventsubscription ^
    --sns-topic-arn arn:aws:sns:us-east-1:123456789012:myawsuser-RDS ^
    --enabled
```

API

要订阅 Amazon RDS 事件通知，请调用 Amazon RDS API 函数 `CreateEventSubscription`。包括以下必需参数：

- `SubscriptionName`
- `SnsTopicArn`

Amazon RDS 事件通知标签和属性

当 Amazon RDS 向 Amazon Simple Notification Service（SNS）或 Amazon EventBridge 发送事件通知时，通知包含消息属性和事件标签。RDS 将消息属性与消息一起单独发送，而事件标签位于消息正文中。使用消息属性和 Amazon RDS 标签将元数据添加到您的资源中。您可以使用您自己的关于数据库实例的表示法来修改这些标签。有关标记 Amazon RDS 资源的更多信息，请参阅为 Amazon RDS 资源添加标签。

默认情况下，Amazon SNS 和 Amazon EventBridge 会接收发送给它们的每条消息。SNS 和 EventBridge 可以筛选消息，并将通知发送到首选的通信模式，例如电子邮件、短信或对 HTTP 端点的调用。

> **Note**
> 通过电子邮件或短信发送的通知将没有事件标签。

下表显示了发送给主题订阅用户的 RDS 事件的消息属性。

Amazon RDS 事件属性	描述
EventID	RDS 事件消息的标识符，例如 RDS-EVENT-0006。
资源	发出事件的资源的 ARN 标识符，例如 `arn:aws:rds:ap-southeast-2:123456789012:db:database-1` 。

RDS 标签提供有关受服务事件影响的资源的数据。当通知发送到 SNS 或 EventBridge 时，RDS 会在消息正文中添加标签的当前状态。

有关筛选 SNS 的消息属性的更多信息，请参阅《Amazon Simple Notification Service 开发人员指南》中的 Amazon SNS 消息筛选。

有关筛选 EventBridge 的事件标签的更多信息，请参阅《Amazon EventBridge 用户指南》中的 Comparison operators for use in event patterns in Amazon EventBridge。

有关筛选 SNS 的基于有效载荷的标签的更多信息，请参阅 Introducing payload-based message filtering for Amazon SNS

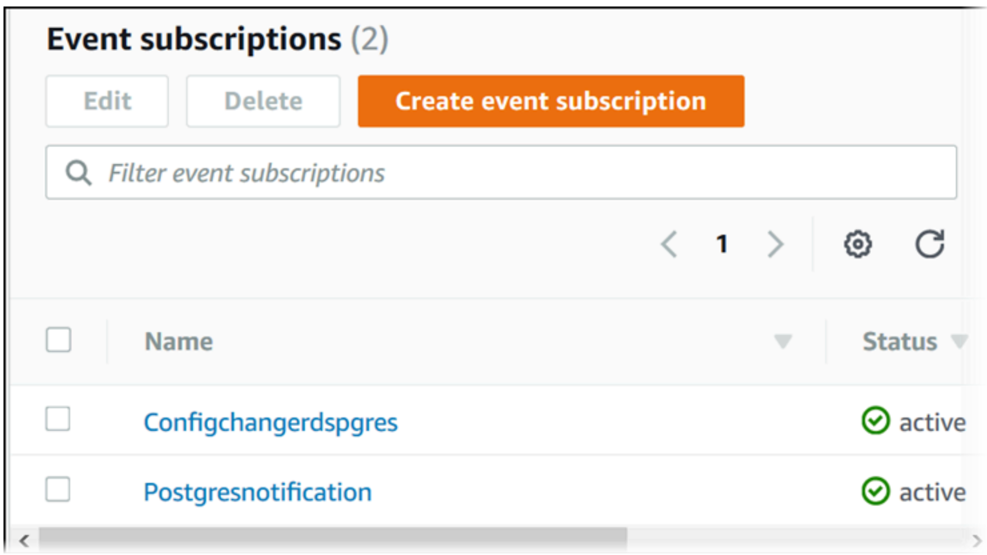
列出 Amazon RDS 事件通知订阅

您可以列出当前的 Amazon RDS 事件通知订阅。

控制台

列出当前的 Amazon RDS 事件通知订阅

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择事件订阅。事件订阅窗格中会显示您的所有事件通知订阅。



Amazon CLI

要列出当前的 Amazon RDS 事件通知订阅，请使用 Amazon CLI `describe-event-subscriptions` 命令。

Example

以下示例描述所有事件订阅。

```
aws rds describe-event-subscriptions
```

以下示例描述 `myfirsteventsubscription`。

```
aws rds describe-event-subscriptions --subscription-name myfirsteventsubscription
```

API

要列出当前的 Amazon RDS 事件通知订阅，请调用 Amazon RDS API `DescribeEventSubscriptions` 操作。

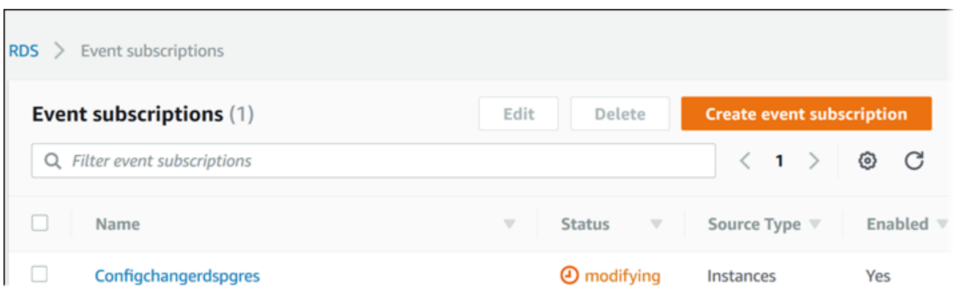
修改 Amazon RDS 事件通知订阅

创建订阅后，您可以更改订阅名称、源标识符、类别或主题 ARN。

控制台

修改 Amazon RDS 事件通知订阅

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择事件订阅。
3. 在事件订阅窗格中，选择您要修改的订阅，然后选择编辑。
4. 在目标或来源部分中对订阅进行更改。
5. 选择编辑。Amazon RDS 控制台会表明正在修改订阅。



Amazon CLI

要修改 Amazon RDS 事件通知订阅，请使用 Amazon CLI `modify-event-subscription` 命令。包括以下必需参数：

- `--subscription-name`

Example

以下代码实现 `myeventsubscription`。

对于 Linux、macOS 或 Unix：

```
aws rds modify-event-subscription \
    --subscription-name myeventsubscription \
    --enabled
```

对于 Windows：

```
aws rds modify-event-subscription ^
    --subscription-name myeventsubscription ^
    --enabled
```

API

要修改 Amazon RDS 事件，请调用 Amazon RDS API 操作 ModifyEventSubscription。包括以下必需参数：

- `SubscriptionName`

在 Amazon RDS 事件通知订阅中添加源标识符

您可以给现有的订阅添加源标识符 (生成事件的 Amazon RDS 源)。

控制台

您可以使用 Amazon RDS 控制台在修改订阅时通过选择或者取消选择操作轻松地添加或删除源标识符。有关更多信息，请参阅“修改 Amazon RDS 事件通知订阅”。

Amazon CLI

要将源标识符添加到 Amazon RDS 事件通知订阅，请使用 Amazon CLI `add-source-identifier-to-subscription` 命令。包括以下必需参数：

- `--subscription-name`
- `--source-identifier`

Example

以下示例将源标识符 `mysqldb` 添加到 `myrdseventsubscription` 订阅。

对于 Linux、macOS 或 Unix：

```
aws rds add-source-identifier-to-subscription \
    --subscription-name myrdseventsubscription \
    --source-identifier mysqldb
```

对于 Windows：

```
aws rds add-source-identifier-to-subscription ^
    --subscription-name myrdseventsubscription ^
    --source-identifier mysqldb
```

API

要将源标识符添加到 Amazon RDS 事件通知订阅，请调用 Amazon RDS API `AddSourceIdentifierToSubscription`。包括以下必需参数：

- `SubscriptionName`
- `SourceIdentifier`

从 Amazon RDS 事件通知订阅中删除源标识符

如果您不再希望收到源的事件通知，可以从订阅中删除源标识符 (生成事件的 Amazon RDS 源)。

控制台

您可以使用 Amazon RDS 控制台在修改订阅时通过选择或者取消选择操作轻松地添加或删除源标识符。有关更多信息，请参阅“修改 Amazon RDS 事件通知订阅”。

Amazon CLI

要从 Amazon RDS 事件通知订阅删除源标识符，请使用 Amazon CLI `remove-source-identifier-from-subscription` 命令。包括以下必需参数：

- `--subscription-name`
- `--source-identifier`

Example

以下示例将从 mysqldb 订阅中删除源标识符 myrdseventsubscription。

对于 Linux、macOS 或 Unix：

```
aws rds remove-source-identifier-from-subscription \
    --subscription-name myrdseventsubscription \
    --source-identifier mysqldb
```

对于 Windows：

```
aws rds remove-source-identifier-from-subscription ^
    --subscription-name myrdseventsubscription ^
    --source-identifier mysqldb
```

API

要从 Amazon RDS 事件通知订阅删除源标识符，请使用 Amazon RDS API `RemoveSourceIdentifierFromSubscription` 命令。包括以下必需参数：

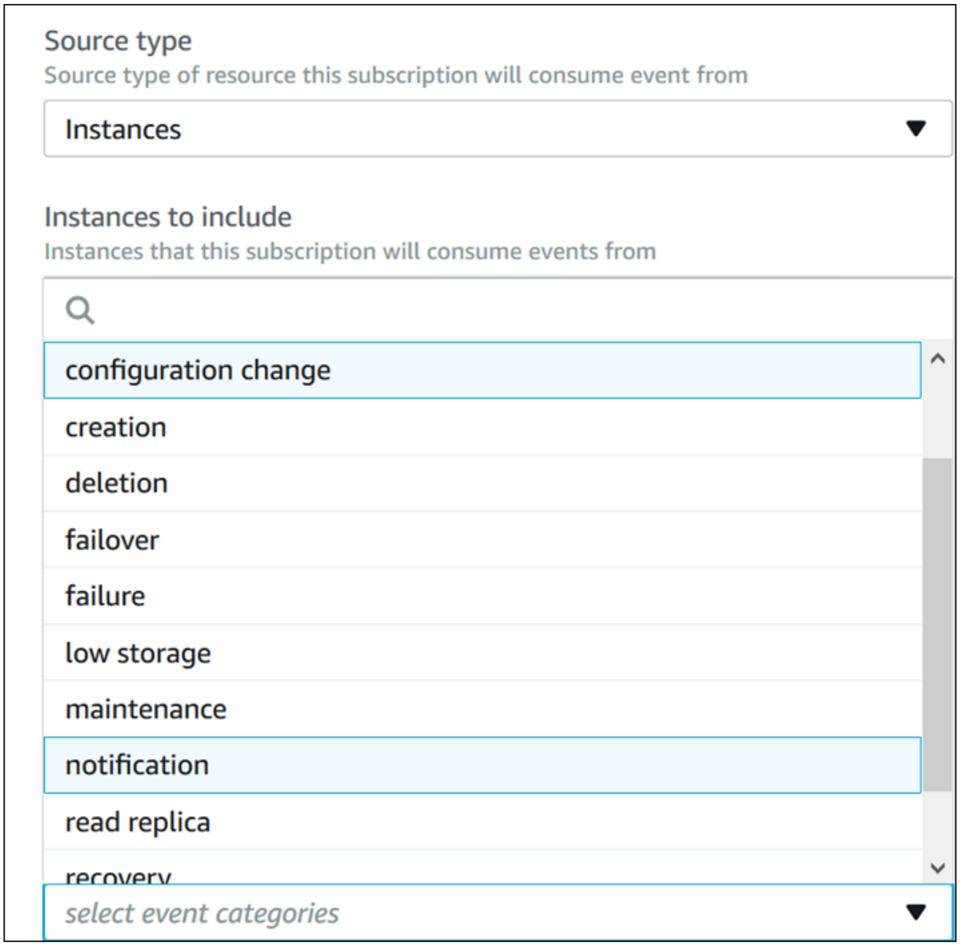
- `SubscriptionName`
- `SourceIdentifier`

列出 Amazon RDS 事件通知类型

资源类型的所有事件都可以分组为类型。要查看可用类型的列表，请使用下列步骤。

控制台

当创建或者修改事件通知订阅时，事件类型会显示在 Amazon RDS 控制台中。有关更多信息，请参阅"修改 Amazon RDS 事件通知订阅"。



Amazon CLI

要列出 Amazon RDS 事件通知类别，请使用 Amazon CLI `describe-event-categories` 命令。此命令没有必需参数。

Example

```
aws rds describe-event-categories
```

API

要列出 Amazon RDS 事件通知类别，请使用 Amazon RDS API `DescribeEventCategories` 命令。此命令没有必需参数。

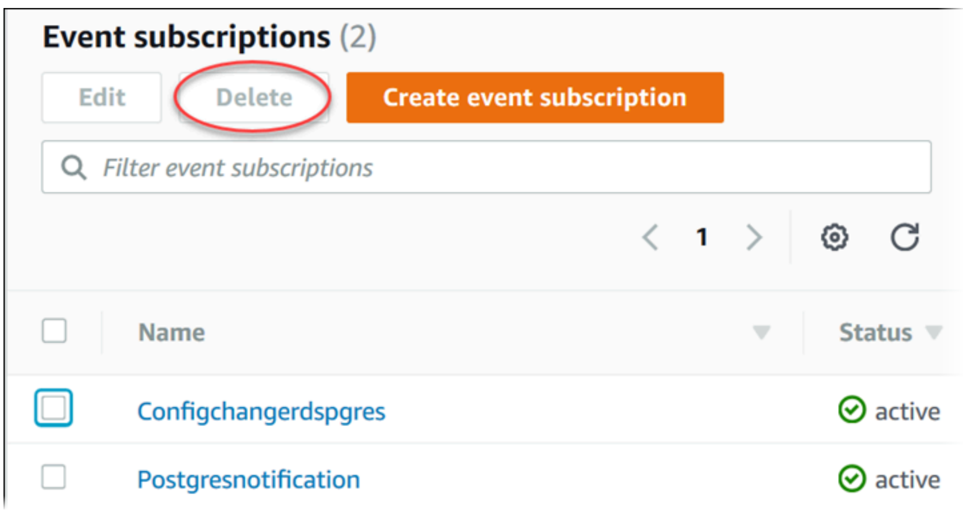
删除 Amazon RDS 事件通知订阅

当您不再需要时，可以删除订阅。该主题的所有用户都将再也不会收到订阅指定的事件通知。

控制台

删除 Amazon RDS 事件通知订阅

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择数据库事件订阅。
3. 在我的数据库事件订阅窗格中，选择您希望删除的订阅。
4. 选择 Delete。
5. Amazon RDS 控制台会表明正在删除订阅。



Amazon CLI

要删除 Amazon RDS 事件通知订阅，请使用 Amazon CLI `delete-event-subscription` 命令。包括以下必需参数：

- `--subscription-name`

Example

以下示例删除订阅 `myrdssubscription`。

```
aws rds delete-event-subscription --subscription-name myrdssubscription
```

API

要删除 Amazon RDS 事件通知订阅，请使用 RDS API `DeleteEventSubscription` 命令。包括以下必需参数：

- `SubscriptionName`

创建对 Amazon RDS 事件触发的规则

使用 Amazon EventBridge，您可以自动执行 Amazon 服务和响应系统事件，例如应用程序可用性问题或资源更改。

主题

创建将 Amazon RDS 事件发送到 Amazon EventBridge 的规则

您可以编写简单规则来指示所关注的 Amazon RDS 事件，并指示在事件匹配规则时要执行的自动化操作。您可以设置接收 JSON 格式的事件的各种目标，例如 Amazon Lambda 函数或 Amazon SNS 主题。例如，您可以将 Amazon RDS 配置为只要创建或删除数据库实例就向 Amazon EventBridge 发送事件。有关更多信息，请参阅 Amazon CloudWatch Events 用户指南和 Amazon EventBridge 用户指南。

创建对 RDS 事件触发的规则：

1. 通过以下网址打开 CloudWatch 控制台：https://console.aws.amazon.com/cloudwatch/。
2. 在导航窗格中的事件下，选择规则。
3. 选择 Create rule (创建规则)。
4. 对于 Event Source (事件源)，执行以下操作：

   a. 选择 Event Pattern (事件模式)。
   b. 对于服务名称，请选择 Relational Database Service (RDS)。
   c. 对于 Event Type（事件类型），请选择触发事件的 Amazon RDS 资源的类型。例如，如果数据库实例触发事件，请选择 RDS DB Instance Event（RDS 数据库实例事件）。
5. 对于目标，选择添加目标，并选择当检测到选定类型的事件时要执行的 Amazon 服务。
6. 在此部分的其他字段中，根据需要输入此目标类型的特定信息。
7. 对于许多目标类型，EventBridge 需要权限以便将事件发送到目标。在这些情况下，EventBridge 可以创建事件运行所需的 IAM 角色：

   - 要自动创建 IAM 角色，请选择为此特定资源创建新角色。
   - 要使用您之前创建的 IAM 角色，请选择使用现有角色。
8. 根据需要，可以重复步骤 5 至 7 为此规则添加另一目标。

9. 选择 Configure details (配置详细信息)。对于 Rule definition (规则定义)，键入规则的名称和描述。

   规则名称在此区域中必须是唯一的。
10. 选择 Create rule (创建规则)。

有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的 Creating an EventBridge Rule That Triggers on an Event。

教程：使用 Amazon EventBridge 记录数据库实例的状态更改

在本教程中，您可以创建 Amazon Lambda 函数来记录 Amazon RDS 实例的状态更改。然后，您可以创建一项规则，只要现有 RDS 数据库实例的状态发生更改，便运行该函数。本教程假定您有一个小型的可以暂时关闭的正在运行的测试实例。

> **Important**
> 请勿在正在运行的生产数据库实例上执行本教程。

主题

步骤 1：创建 Amazon Lambda 函数

创建 Lambda 函数以记录状态更改事件。在创建规则时，您可以指定此函数。

创建 Lambda 函数

1. 打开 Amazon Lambda 控制台，地址：https://console.aws.amazon.com/lambda/。
2. 如果您是首次接触 Lambda，您将看到欢迎页面。选择 Get Started Now (立即开始)。否则，选择创建函数。
3. 选择 Author from scratch (从头创作)。
4. 在 Create function (创建函数) 页面上，执行以下操作：

a. 输入 Lambda 函数的名称和说明。例如，将函数命名为 **RDSInstanceStateChange**。
b. 在 Runtime（运行时）中，选择 Node.js 16x。
c. 对于 Architecture（架构），选择 x86_64。
d. 对于 Execution role（执行角色），请执行以下任一操作：

   - 选择 Create a new role with basic Lambda permissions (创建具有基本 Lambda 权限的新角色)。
   - 对于 Existing role（现有角色），选择 Use an existing role（使用现有角色）。选择要使用的角色。

e. 选择创建函数。

5. 在 RDSInstanceStateChange 页面上，请执行以下操作：

   a. 在 Code source (代码源) 中，请选择index.js。
   b. 在 index.js 窗格中，删除现有代码。
   c. 输入以下代码：

```
console.log('Loading function');

exports.handler = async (event, context) => {
    console.log('Received event:', JSON.stringify(event));
};
```

   d. 选择 Deploy (部署)。

步骤 2：创建规则

创建一个规则，以便启动 Lambda 实例时运行 Amazon RDS 函数。

创建 EventBridge 规则

1. 打开位于 https://console.aws.amazon.com/events/ 的 Amazon EventBridge 控制台。
2. 在导航窗格中，选择规则。
3. 选择创建规则。
4. 为规则输入名称和描述。例如，输入 **RDSInstanceStateChangeRule**。
5. 选择 Rule with an event pattern（具有事件模式的规则），然后选择 Next（下一步）。
6. 对于事件源，选择Amazon 事件或 EventBridge 合作伙伴事件。

7. 向下滚动到 Event pattern（事件模式）部分。
8. 对于事件源，选择 Amazon Web Services 服务。
9. 对于 Amazon 服务，请选择关系数据库服务（RDS）。
10. 对于 Event type (事件类型)，请选择 RDS DB Instance Event (RDS 数据库实例事件)。
11. 保留原定设置事件模式。然后选择下一步。
12. 对于目标类型，选择Amazon 服务。
13. 对于 Select a target（选择目标），选择 Lambda function（Lambda 函数）。
14. 对于 Function（函数），选择您创建的 Lambda 函数。然后选择下一步。
15. 在 Configure tags（配置标签）中，选择 Next（下一步）。
16. 查看您的规则中的步骤。然后，选择 Create rule（创建规则）。

步骤 3：测试规则

要测试您的规则，请关闭 RDS 数据库实例。等待几分钟，在实例关闭后，验证您的 Lambda 函数是否已调用。

通过停止一个数据库实例来测试您的规则

1. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 停止一个 RDS 数据库实例
3. 打开位于 https://console.aws.amazon.com/events/ 的 Amazon EventBridge 控制台。
4. 在导航窗格中，选择 Rules (规则)，再选择所创建规则的名称。
5. 在规则详细信息中，选择监控。

   随后您将被重定向至 Amazon CloudWatch 控制台。如果您未被重定向，请单击查看 CloudWatch 中的指标。
6. 在 All metrics (所有指标)中，选择所创建规则的名称。

   该图表表明规则已被调用。
7. 在导航窗格中，选择 Log groups (日志组)。
8. 选择 Lambda 函数 (/aws/lambda/***function-name***)的日志组的名称。
9. 选择日志流的名称，以查看您启动的实例的函数提供的数据。您应该会看到接收的事件，类似以下内容：

```
{
```

```
    "version": "0",
    "id": "12a345b6-78c9-01d2-34e5-123f4ghi5j6k",
    "detail-type": "RDS DB Instance Event",
    "source": "aws.rds",
    "account": "111111111111",
    "time": "2021-03-19T19:34:09Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:rds:us-east-1:111111111111:db:testdb"
    ],
    "detail": {
        "EventCategories": [
            "notification"
        ],
        "SourceType": "DB_INSTANCE",
        "SourceArn": "arn:aws:rds:us-east-1:111111111111:db:testdb",
        "Date": "2021-03-19T19:34:09.293Z",
        "Message": "DB instance stopped",
        "SourceIdentifier": "testdb",
        "EventID": "RDS-EVENT-0087"
    }
}
```

有关 JSON 格式的 RDS 事件的更多示例，请参阅Amazon RDS 事件概述。

10. （可选）完成后，您可以打开 Amazon RDS 控制台并启动之前停止的实例。

Amazon RDS 事件类别和事件消息

Amazon RDS 会在各种类型中生成许多事件，您可以使用 Amazon RDS 控制台、Amazon CLI 或 API 订阅这些事件。

主题

数据库集群事件

下表显示了数据库集群为源类型时的事件类别和事件列表。

有关多可用区数据库集群部署的更多信息，请参阅Amazon RDS 的多可用区数据库集群部署。

类别	RDS 事件 ID	消息	备注
配置更改	RDS-EVENT-0016	重置主凭证。	无
创建	RDS-EVENT-0170	已创建数据库集群。	无
故障转移	RDS-EVENT-0069	集群故障转移失败，请检查集群实例的运行状况，然后重试。	无
故障转移	RDS-EVENT-0070	再次提升之前的主实例：*name*。	无
故障转移	RDS-EVENT-0071	已完成到数据库实例 *name* 的故障转移。	无

类别	RDS 事件 ID	消息	备注
故障转移	RDS-EVENT-0072	已开始将同一可用区故障转移到数据库实例：*name*。	无
故障转移	RDS-EVENT-0073	已开始跨可用区故障转移到数据库实例：*name*。	无
失败	RDS-EVENT-0354	由于资源不兼容，您无法创建数据库集群。*message*。	*message* 包括有关失败的详细信息。
失败	RDS-EVENT-0355	由于资源不足限制，无法创建数据库集群。*message*。	*message* 包括有关失败的详细信息。
维护	RDS-EVENT-0156	数据库集群具有数据库引擎次要版本升级。	无
maintenance	RDS-EVENT-0173	数据库集群引擎版本已升级。	数据库集群的修补已完成。
maintenance	RDS-EVENT-0174	数据库集群处于无法升级的状态。	无
maintenance	RDS-EVENT-0176	数据库集群引擎主要版本已升级。	无
maintenance	RDS-EVENT-0177	数据库集群升级正在进行中。	无
maintenance	RDS-EVENT-0286	数据库集群引擎 *version_number* 版本升级已启动。集群保持联机状态。	无
maintenance	RDS-EVENT-0287	检测到操作系统升级要求。	无
maintenance	RDS-EVENT-0288	集群操作系统升级开始。	无
maintenance	RDS-EVENT-0289	集群操作系统升级已完成。	无

类别	RDS 事件 ID	消息	备注
maintenance	RDS-EVENT-0290	数据库集群已修补：源版本 *version_number* => *new_version_number* 。	无
maintenance	RDS-EVENT-0410	数据库集群引擎版本升级的预检查已开始。	无
maintenance	RDS-EVENT-0412	数据库集群引擎版本升级的预检查失败或超时。	无
maintenance	RDS-EVENT-0413	数据库集群升级前任务正在进行中。	无
maintenance	RDS-EVENT-0414	数据库集群升级后的任务正在进行中。	无
maintenance	RDS-EVENT-0417	数据库集群引擎版本升级已开始。	无
notification	RDS-EVENT-0172	已将集群从 *name* 重命名为 *name*。	无
只读副本	RDS-EVENT-0411	数据库集群引擎版本升级的预检查已完成。	无

数据库集群快照事件

下表显示了数据库集群快照为源类型时的事件类别和事件列表。

类别	RDS 事件 ID	消息	备注
备份	RDS-EVENT-0074	正在创建手动集群快照。	无
备份	RDS-EVENT-0075	已创建手动集群快照。	无
备份	RDS-EVENT-0168	正在创建自动集群快照。	无

类别	RDS 事件 ID	消息	备注
备份	RDS-EVENT-0169	已创建自动集群快照。	无

数据库实例事件

下表显示的是数据库实例为源类型时的事件类型和事件列表。

类别	RDS 事件 ID	消息	备注
可用性	RDS-EVENT-0004	数据库实例已关闭。	无
可用性	RDS-EVENT-0006	已重新启动数据库实例。	无
可用性	RDS-EVENT-0022	重新启动 mysql 时出错：*message*。	重新启动 MySQL 时出现了错误。
可用性	RDS-EVENT-0221	数据库实例已达到存储已满阈值，并且数据库已关闭。您可以增加分配的存储空间以解决此问题。	无
可用性	RDS-EVENT-0222	数据库实例 *name* 的可用存储容量低至分配存储空间的 *percentage* [分配的存储空间：*amount*，可用存储空间：*amount*]。如果可用存储空间低于 *amount*，将关闭数据库以防止损坏。您可以增加分配的存储空间以解决此问题。	仅当数据库实例消耗的已分配存储空间超过 90% 时，才适用于 RDS for MySQL。使用可用存储空间指标监控数据库实例的存储空间。有关更多信息，请参阅 Amazon RDS 数据库实例存储。
可用性	RDS-EVENT-0330	对于数据库实例 *name*，专用事务日志卷的可用存储空间容量过低。日志卷可用存储空间为 *percentage* 的已分配存储空间。[分配的存储空间：*amount*，可用存	有关更多信息，请参阅 专用日志卷（DLV）。

类别	RDS 事件 ID	消息	备注
		储空间：*amount*] 如果可用存储空间低于 *amount*，将关闭数据库以防止损坏。您可以禁用专用事务日志卷来解决此问题。	
可用性	RDS-EVENT-0331	对于数据库实例 *name*，专用事务日志卷的可用存储空间容量过低。日志卷可用存储空间为 *percentage* 的预置存储空间。[预置存储空间：*amount*，可用存储空间：*amount*] 您可以禁用专用事务日志卷来解决此问题。	有关更多信息，请参阅 专用日志卷（DLV）。
可用性	RDS-EVENT-0396	Amazon RDS 已计划在内部用户密码轮换后，在此实例的下一个维护时段中重启此只读副本。	无
可用性	RDS-EVENT-0419	Amazon RDS 无法访问数据库实例 *name* 的 KMS 加密密钥。此数据库将进入无法访问的状态。有关更多详细信息，请参阅 Amazon RDS 文档中的故障排除部分。	无
备份	RDS-EVENT-0001	备份数据库实例。	无
备份	RDS-EVENT-0002	已完成数据库实例备份。	无

类别	RDS 事件 ID	消息	备注
备份	RDS-EVENT-0086	我们无法将选项组 *name* 与数据库实例 *name* 相关联。确认数据库实例类和配置支持该选项组 *name*。如果是，请验证所有选项组设置并重试。	有关更多信息，请参阅“使用选项组”。
配置更改	RDS-EVENT-0011	已更新为使用 DBParameterGroup *name*。	无
配置更改	RDS-EVENT-0012	将修改应用于数据库实例类。	无
配置更改	RDS-EVENT-0014	已完成将修改应用于数据库实例类。	无
配置更改	RDS-EVENT-0016	重置主凭证。	无
配置更改	RDS-EVENT-0017	已完成对分配的存储应用修改。	无
配置更改	RDS-EVENT-0018	对分配的存储应用修改。	无
配置更改	RDS-EVENT-0024	应用修改以转换为多可用区数据库实例。	无
配置更改	RDS-EVENT-0025	已完成应用修改以转换为多可用区数据库实例。	无
配置更改	RDS-EVENT-0028	已禁用自动备份。	无
配置更改	RDS-EVENT-0029	已完成应用修改以转换为标准（单可用区）数据库实例。	无
配置更改	RDS-EVENT-0030	应用修改以转换为标准（单可用区）数据库实例。	无

类别	RDS 事件 ID	消息	备注
配置更改	RDS-EVENT-0032	已启用自动备份。	无
配置更改	RDS-EVENT-0033	*number* 位用户与主用户名相匹配；仅重置未绑定到特定主机的用户名。	无
配置更改	RDS-EVENT-0067	无法重置密码。错误信息：*message*。	无
配置更改	RDS-EVENT-0078	监控间隔已更改为 *number*。	增强监控配置进行了更改。
配置更改	RDS-EVENT-0092	已完成数据库参数组更新。	无
配置更改	RDS-EVENT-0217	正在向已分配的存储应用启用了弹性伸缩的修改。	无
配置更改	RDS-EVENT-0218	已向已分配的存储应用启用了弹性伸缩的修改。	无
配置更改	RDS-EVENT-0295	存储配置升级已开始。	无
配置更改	RDS-EVENT-0296	存储配置升级已完成。	无
配置更改	RDS-EVENT-0332	专用日志卷已禁用。	有关更多信息，请参阅 专用日志卷（DLV）。
配置更改	RDS-EVENT-0333	已开始禁用专用日志卷。	有关更多信息，请参阅 专用日志卷（DLV）。
配置更改	RDS-EVENT-0334	已开始启用专用日志卷。	有关更多信息，请参阅 专用日志卷（DLV）。
配置更改	RDS-EVENT-0335	专用日志卷已启用。	有关更多信息，请参阅 专用日志卷（DLV）。

类别	RDS 事件 ID	消息	备注
配置更改	RDS-EVENT-0383	`engine version` 不支持 memcached 插件。RDS 将继续升级您的数据库实例并移除此插件。	从 MySQL 8.3.0 开始，不支持 memcached 插件。有关更多信息，请参阅 [Changes in MySQL 8.3.0 (2024-01-16, Innovation Release)]。
创建	RDS-EVENT-0005	数据库实例已创建。	无
删除	RDS-EVENT-0003	已删除数据库实例。	无
故障转移	RDS-EVENT-0013	已启动多可用区实例故障转移。	已启动可导致提升备用数据库实例的多可用区故障转移。
故障转移	RDS-EVENT-0015	多可用区故障转移到待机状态已完成 - DNS 传播可能需要几分钟。	已完成可导致提升备用数据库实例的多可用区故障转移。可能需要几分钟的时间才能让 DNS 传输到新的主数据库实例。
故障转移	RDS-EVENT-0034	由于数据库实例最近发生了故障转移，因此正在放弃用户请求的故障转移。	因为数据库实例上最近发生了故障转移，所以 Amazon RDS 不会尝试所请求的故障转移。
故障转移	RDS-EVENT-0049	多可用区实例故障转移已完成。	无
故障转移	RDS-EVENT-0050	已启动多可用区实例激活。	在成功恢复数据库实例后，启动了多可用区激活。如果 Amazon RDS 将主数据库实例提升到与之前的主数据库实例相同的可用区，则会出现此事件。

类别	RDS 事件 ID	消息	备注
故障转移	RDS-EVENT-0051	多可用区实例激活已完成。	多可用区激活已完成。现在，应该可以访问您的数据库了。
故障转移	RDS-EVENT-0065	已从部分故障转移中恢复。	无
失败	RDS-EVENT-0031	数据库实例进入 *name* 状态。RDS 建议启动时间点还原。	由于某个不兼容配置或底层存储问题，数据库实例已失败。开始对数据库实例进行时间点还原。
失败	RDS-EVENT-0035	数据库实例进入 *state*。*message*。	数据库实例有无效参数。例如，如果数据库实例因为此实例类的内存相关参数设置过高而无法启动，您就应该修改内存参数，并重启数据库实例。
失败	RDS-EVENT-0036	数据库实例处于 *state*。*message*。	数据库实例处于不兼容的网络中。有些指定的子网 ID 无效或者不存在。
失败	RDS-EVENT-0058	Statspack 安装失败。*message*。	创建 Oracle Statspack 用户账户 PERFSTAT 时出错。请先删除该账户，然后添加 STATSPACK 选项。
失败	RDS-EVENT-0079	Amazon RDS 无法创建用于增强监控的凭证，此特征已被禁用。这可能是由于您的账户中 rds-monitoring-role 不存在且配置不正确而致。有关更多详细信息，请参阅 Amazon RDS 文档中的故障排除部分。	如果没有增强监控 IAM 角色，则无法启用增强监控。有关创建 IAM 角色的信息，请参阅为 Amazon RDS 增强监控创建 IAM 角色。

类别	RDS 事件 ID	消息	备注
失败	RDS-EVENT-0080	Amazon RDS 无法在您的实例 *name* 上配置增强监控，此特征已被禁用。这可能是由于您的账户中 rds-monitoring-role 不存在且配置不正确而致。有关更多详细信息，请参阅 Amazon RDS 文档中的故障排除部分。	增强监控因在配置更改期间出错而处于禁用状态。可能是未正确配置增强监控 IAM 角色。有关创建增强监控 IAM 角色的信息，请参阅为 Amazon RDS 增强监控创建 IAM 角色。
失败	RDS-EVENT-0081	Amazon RDS 无法为 *name* 选项创建凭证。这是由于在您的账户中未正确配置 *name* IAM 角色。有关更多详细信息，请参阅 Amazon RDS 文档中的故障排除部分。	用于访问您的 Amazon S3 存储桶以执行 SQL Server 本机备份和恢复的 IAM 角色配置不正确。有关更多信息，请参阅“针对本机备份和还原进行设置”。
失败	RDS-EVENT-0165	RDS Custom 数据库实例位于支持外围之外。	您应负责修复导致 RDS Custom 数据库实例处于 `unsupported-configuration` 状态的配置问题。如果问题与 Amazon 基础设施有关，则您可以使用控制台或 Amazon CLI 修复该问题。如果问题与操作系统或数据库配置有关，则您可以登录到主机进行修复。 有关更多信息，请参阅 RDS Custom 支持外围。

类别	RDS 事件 ID	消息	备注
失败	RDS-EVENT-0188	数据库实例处于无法升级的状态。*message*	由于与数据字典相关的不兼容性，Amazon RDS 无法升级 MySQL 数据库实例。因尝试升级到 8.0 版本失败，数据库实例已回滚到 MySQL 版本 5.7，或者因尝试升级到 8.4 版本失败，数据库实例已回滚到 MySQL 版本 8.0。有关更多信息，请参阅 升级失败后回滚。
失败	RDS-EVENT-0219	数据库实例处于无效状态。无需采取操作。弹性伸缩稍后将重试。	无
失败	RDS-EVENT-0220	由于之前的扩展存储操作，数据库实例正处于冷却期。我们正在优化您的数据库实例。这至少需要 6 个小时。无需采取操作。弹性伸缩将在冷却期结束后重试。	无
失败	RDS-EVENT-0223	存储自动扩缩无法扩展存储空间，原因为：*reason*。	无
失败	RDS-EVENT-0224	存储自动扩缩已触发待处理的扩展存储任务，该任务将达到或超过最大存储阈值。提高最大存储阈值。	无
失败	RDS-EVENT-0237	数据库实例的存储类型目前在可用区中不可用。弹性伸缩稍后将重试。	无

类别	RDS 事件 ID	消息	备注
失败	RDS-EVENT-0254	此客户账户的基础存储限额已超过限制。请增加允许的存储限额，以便在实例上进行扩展。	无
失败	RDS-EVENT-0278	数据库实例创建失败。*message*	*message* 包括有关失败的详细信息。
失败	RDS-EVENT-0279	提升 RDS Custom 只读副本失败。*message*	*message* 包括有关失败的详细信息。
失败	RDS-EVENT-0280	由于预检查失败，RDS Custom 无法升级数据库实例。*message*	*message* 包括有关失败的详细信息。
失败	RDS-EVENT-0281	由于预检查失败，RDS Custom 无法修改数据库实例。*message*	*message* 包括有关失败的详细信息。
失败	RDS-EVENT-0282	RDS Custom 无法修改数据库实例，因为弹性 IP 权限不正确。请确认弹性 IP 地址已标记为 `AWSRDSCustom` 。	无
失败	RDS-EVENT-0283	RDS Custom 无法修改数据库实例，因为您的账户已达到弹性 IP 限制。释放未使用的弹性 IP 或请求增加弹性 IP 地址限制的限额。	无
失败	RDS-EVENT-0284	由于预检查失败，RDS Custom 无法将实例转换为高可用性。*message*	*message* 包括有关失败的详细信息。

类别	RDS 事件 ID	消息	备注
失败	RDS-EVENT-0285	RDS Custom 无法为数据库实例创建最终快照，因为显示了 *message*。	*message* 包括有关失败的详细信息。
失败	RDS-EVENT-0421	RDS Custom 无法将数据库实例转换为多可用区部署：*message*。实例将保持单可用区部署。有关 RDS Custom for Oracle 的多可用区部署的信息，请参阅《RDS 用户指南》。	*message* 包括有关失败的详细信息。
失败	RDS-EVENT-0306	存储配置升级失败。请重试升级。	无
失败	RDS-EVENT-0315	无法将不兼容网络的数据库 *name* 移至可用状态：*message*	数据库网络配置无效。无法将数据库从不兼容的网络移至可用状态。
失败	RDS-EVENT-0328	无法将主机加入域。例如，*instancename* 实例的域成员资格状态已设置为失败。	无
失败	RDS-EVENT-0329	无法将主机加入您的域。在加入域期间，Microsoft Windows 返回了错误代码*##*。验证您的网络和权限配置，然后发出重新尝试加入域的 `modify-db-instance` 请求。	在使用自托管 Active Directory 时，请参阅[自托管式 Active Directory 问题排查]()。
失败	RDS-EVENT-0353	由于资源不足限制，无法创建数据库实例。*message*。	*message* 包括有关失败的详细信息。

类别	RDS 事件 ID	消息	备注
失败	RDS-EVENT-0356	RDS 无法在您的域中配置 Kerberos 端点。这可能会阻止对您的数据库实例进行 Kerberos 身份验证。验证数据库实例和域控制器之间的网络配置。	无
失败	RDS-EVENT-0418	Amazon RDS 无法访问数据库实例 *name* 的 KMS 加密密钥。这可能是由于密钥被禁用或 Amazon RDS 无法访问密钥所致。如果这种情况持续，数据库将进入无法访问的状态。有关更多详细信息，请参阅 Amazon RDS 文档中的故障排除部分。	无
失败	RDS-EVENT-0420	Amazon RDS 现在可以成功访问数据库实例 *name* 的 KMS 加密密钥。	无
存储不足	RDS-EVENT-0007	分配的存储空间已耗尽。分配更多存储空间以解决问题。	分配的数据库实例存储空间已使用。要解决此问题，请为数据库实例分配额外存储。有关更多信息，请参阅 RDS 常见问题。您可以使用可用存储空间指标监控数据库实例的存储空间。

类别	RDS 事件 ID	消息	备注
存储不足	RDS-EVENT-0089	数据库实例 *name* 的可用存储容量低至预调配存储空间的 *percentage* [预调配存储空间：*size*，可用存储空间：*size*]。您可能需要增加预调配存储空间以解决此问题。	数据库实例已使用其分配的存储空间的 90% 以上。您可以使用可用存储空间指标监控数据库实例的存储空间。
存储不足	RDS-EVENT-0227	您的 Aurora 集群的存储空间极其低，只剩下 *amount* 太字节。请采取措施以减少集群上的存储负载。	Aurora 存储子系统的空间不足。
maintenance	RDS-EVENT-0026	将离线补丁应用到数据库实例。	正在进行数据库实例的脱机维护。数据库实例目前无法使用。
维护	RDS-EVENT-0027	已完成将离线补丁应用到数据库实例。	数据库实例的脱机维护已完成。现在可以使用数据库实例。
维护	RDS-EVENT-0047	数据库实例已修补。	无
maintenance	RDS-EVENT-0155	数据库实例具有数据库引擎次要版本升级。	无
maintenance	RDS-EVENT-0178	数据库实例升级正在进行中。	无
maintenance	RDS-EVENT-0264	数据库引擎版本升级的预检查已开始。	无
maintenance	RDS-EVENT-0265	数据库引擎版本升级的预检查已完成。	无
maintenance	RDS-EVENT-0266	数据库实例已开始停机。	无

类别	RDS 事件 ID	消息	备注
maintenance	RDS-EVENT-0267	引擎版本升级已开始。	无
maintenance	RDS-EVENT-0268	引擎版本升级已完成。	无
maintenance	RDS-EVENT-0269	升级后的任务正在进行中。	无
maintenance	RDS-EVENT-0270	数据库引擎版本升级失败。引擎版本升级回滚成功。	无
maintenance	RDS-EVENT-0398	正在等待主数据库实例上的数据库引擎版本升级完成。	在主要引擎版本升级期间，在只读副本上发出。
maintenance	RDS-EVENT-0399	正在等待只读副本上的数据库引擎版本升级完成。	在主要引擎版本升级期间，在源数据库引擎上发出。
maintenance	RDS-EVENT-0422	由于有待处理的维护操作，RDS 将替换数据库实例 *name* 的主机。	无
维护，故障	RDS-EVENT-0195	*message*	Oracle 时区文件的更新失败。有关更多信息，请参阅 Oracle 时区文件自动升级。
维护，通知	RDS-EVENT-0191	时区文件的新版本可供更新。	当更新 RDS for Oracle 数据库引擎时，如果您没有选择时区文件升级并且数据库不使用实例上可用的最新 DST 时区文件，则 Amazon RDS 会生成此事件。有关更多信息，请参阅 Oracle 时区文件自动升级。
维护，通知	RDS-EVENT-0192	时区文件的更新已开始。	Oracle 时区文件的升级已开始。有关更多信息，请参阅 Oracle 时区文件自动升级。

类别	RDS 事件 ID	消息	备注
维护，通知	RDS-EVENT-0193	当前时区文件版本没有可用的更新。	您的 Oracle 数据库实例使用的是最新的时区文件版本，并且以下任一表述为真： • 您最近添加了 `TIMEZONE_FILE_AUTOUPGRADE` 选项。 • 您的 Oracle 数据库引擎正在升级。 有关更多信息，请参阅 Oracle 时区文件自动升级。
维护，通知	RDS-EVENT-0194	时区文件的更新已完成。	Oracle 时区文件的更新已完成。有关更多信息，请参阅 Oracle 时区文件自动升级。
notification	RDS-EVENT-0044	*message*	这是操作员发出的通知。有关详细信息，请参阅事件消息。
通知	RDS-EVENT-0048	延迟数据库引擎升级，因为此实例有需要先升级的只读副本。	数据库实例的修补已延迟。
通知	RDS-EVENT-0054	*message*	您正在使用的 MySQL 存储引擎不是 InnoDB，而 InnoDB 是推荐给 Amazon RDS 使用的 MySQL 存储引擎。有关 MySQL 存储引擎的信息，请参阅支持的 RDS for MySQL 存储引擎。

类别	RDS 事件 ID	消息	备注
notification	RDS-EVENT-0055	*message*	数据库实例拥有的表数量超过了 Amazon RDS 最佳实践推荐的数量。请减少数据库实例上的表个数。有关推荐的最佳实践的信息，请参阅 Amazon RDS 基本操作指导方针。
通知	RDS-EVENT-0056	*message*	数据库实例包含的数据库数量超过了 Amazon RDS 最佳实践推荐的数量。请减少数据库实例中的数据库数。有关推荐的最佳实践的信息，请参阅 Amazon RDS 基本操作指导方针。
通知	RDS-EVENT-0064	已成功轮换 TDE 加密密钥。	有关推荐的最佳实践的信息，请参阅 Amazon RDS 基本操作指导方针。
通知	RDS-EVENT-0084	无法将数据库实例转换为多可用区：*message*。	您尝试将数据库实例转换为多可用区，但是它包含多可用区不支持的内存文件组。有关更多信息，请参阅“Amazon RDS for Microsoft SQL Server 多可用区部署”。
通知	RDS-EVENT-0087	数据库实例已停止。	无
notification	RDS-EVENT-0088	数据库实例已启动。	无
notification	RDS-EVENT-0154	数据库实例由于它超过最大允许停止的时间而正被启动。	无

类别	RDS 事件 ID	消息	备注
notification	RDS-EVENT-0157	无法修改数据库实例类。*message*。	RDS 无法修改数据库实例类，因为目标实例类无法支持源数据库实例上存在的数据库数。错误消息显示“实例具有 N 个数据库，但在转换后，只支持 N 个”。有关更多信息，请参阅“Microsoft SQL Server 数据库实例的限制”。
通知	RDS-EVENT-0158	数据库实例处于无法升级的状态：*message*。	无
notification	RDS-EVENT-0167	*message*	RDS Custom 支持外围配置已更改。
notification	RDS-EVENT-0189	RDS 数据库实例的 gp2 突增余额积分不足。要解决此问题，请降低 IOPS 使用率或修改存储设置，以提高性能。	RDS 数据库实例的 gp2 突增余额积分不足。要解决此问题，请降低 IOPS 使用率或修改存储设置，以提高性能。有关更多信息，请参阅《Amazon Elastic Compute Cloud 用户指南》中的输入/输出积分和突增性能。
notification	RDS-EVENT-0225	分配的存储大小 *amount* GB 已接近最大存储阈值 *amount* GB。提高最大存储阈值。	当分配的存储量达到最大存储阈值的 80% 时，会调用此事件。为避免发生此事件，请提高最大存储阈值。

类别	RDS 事件 ID	消息	备注
notification	RDS-EVENT-0231	您的数据库实例的存储修改遇到了内部错误。修改请求正在等待处理，稍后将重试。	在读取复制过程中出错。有关详细信息，请参阅事件消息。 此外，请参阅您的数据库引擎的只读副本的故障排除部分。 • 排查 MariaDB 只读副本问题 • 排除 SQL Server 只读副本问题 • 排查 MySQL 只读副本问题 • 排查 RDS for Oracle 副本的问题
notification	RDS-EVENT-0253	数据库正在使用双写缓冲区。*message*。有关更多信息，请参阅 *name* 的 RDS 优化型写入功能文档。	RDS 优化写入与实例存储配置不兼容。有关更多信息，请参阅使用适用于 MySQL 的 RDS 优化型写入功能提高写入性能和使用适用于 MariaDB 的 Amazon RDS 优化型写入功能提高写入性能。 您可以通过创建蓝绿部署来执行存储配置升级以启用优化写入。

类别	RDS 事件 ID	消息	备注
notification	RDS-EVENT-0297	数据库实例*##*的存储配置支持的最大大小为 16384 GiB。执行存储配置升级以支持大于 16384 GiB 的存储大小。	您不能将数据库实例的分配存储大小增加到 16384 GiB 以上。要克服此限制，请执行存储配置升级。有关更多信息，请参阅升级数据库实例的存储文件系统。
notification	RDS-EVENT-0298	数据库实例*##*的存储配置支持的最大表大小为 2048 GiB。执行存储配置升级以支持大于 2048 GiB 的表大小。	具有此限制的 RDS MySQL 和 MariaDB 实例的表大小不能超过 2048 GiB。要克服此限制，请执行存储配置升级。有关更多信息，请参阅升级数据库实例的存储文件系统。
notification	RDS-EVENT-0327	Amazon RDS 找不到密钥 *SECRET ARN*。*message*。	无
notification	RDS-EVENT-0365	已更新时区文件。重新启动 RDS 实例以使更改生效。	无
notification	RDS-EVENT-0385	集群拓扑已更新。	数据库实例的数据库集群发生了 DNS 更改。这包括何时添加或删除新的数据库实例，或者进行失效转移。
notification	RDS-EVENT-0403	数据库工作负载导致系统的内存严重不足。为了帮助缓解此问题，RDS 会自动将 innodb_buffer_pool_size 的值设置为 *amount*。	仅适用于 RDS for MySQL 和 RDS for MariaDB 数据库实例。

类别	RDS 事件 ID	消息	备注
notification	RDS-EVENT-0404	数据库工作负载导致系统的内存严重不足。为了帮助缓解此问题，RDS 会自动将 shared_buffers 的值设置为 *amount*。	仅适用于 RDS for PostgreSQL 数据库实例。
只读副本	RDS-EVENT-0045	复制已停止。	当复制过程中出现错误时，会出现此消息。要确定错误的类型，请参阅排查 MySQL 只读副本问题。
只读副本	RDS-EVENT-0046	已恢复只读副本的复制。	此消息会在您首次创建只读副本时出现，或显示为确认复制在正常运行的监控消息。如果此消息在 `RDS-EVENT-0045` 通知之后出现，则复制已在出现错误之后或停止复制之后恢复。
只读副本	RDS-EVENT-0057	复制流式传输已终止。	无
只读副本	RDS-EVENT-0062	已手动停止只读副本的复制。	无
只读副本	RDS-EVENT-0063	已重置非 RDS 实例的复制。	无
只读副本	RDS-EVENT-0202	创建只读副本失败。	无
只读副本	RDS-EVENT-0357	复制通道 *name* 已启动。	有关复制通道的信息，请参阅the section called “配置多源复制”。
只读副本	RDS-EVENT-0358	复制通道 *name* 已停止。	有关复制通道的信息，请参阅the section called “配置多源复制”。

类别	RDS 事件 ID	消息	备注
只读副本	RDS-EVENT-0359	复制通道 *name* 已手动停止。	有关复制通道的信息，请参阅the section called “配置多源复制”。
只读副本	RDS-EVENT-0360	复制通道 *name* 已重置。	有关复制通道的信息，请参阅the section called “配置多源复制”。
只读副本	RDS-EVENT-0415	升级过程恢复了只读副本上的复制。	无
只读副本	RDS-EVENT-0416	升级过程停止了只读副本上的复制。	无
恢复	RDS-EVENT-0020	已启动数据库实例的还原。恢复时间会随待恢复数据量的变化而变化。	无
恢复	RDS-EVENT-0021	数据库实例的恢复已完成。	无
恢复	RDS-EVENT-0023	紧急快照请求：*message*。	已请求手动备份，但 Amazon RDS 目前处于创建数据库快照的过程中。请在 Amazon RDS 完成数据库快照后再次提交请求。
恢复	RDS-EVENT-0052	已启动多可用区实例恢复。	恢复时间会随待恢复数据量的变化而变化。
恢复	RDS-EVENT-0053	已完成多可用区实例恢复。等待故障转移或激活。	此消息表明，Amazon RDS 已准备好您的数据库实例，可在必要时启动失效转移到辅助实例。

类别	RDS 事件 ID	消息	备注
恢复	RDS-EVENT-0066	重新建立镜像时实例将降级：*message*。	SQL Server 数据库实例正在重新建立其镜像。在镜像重新建立之前，性能将下降。发现具有非 FULL 恢复模式的数据库。恢复模式已重新更改为 FULL，镜像恢复已启动。(<dbname>: <recovery model found>[,...])”
恢复	RDS-EVENT-0166	*message*	RDS Custom 数据库实例位于支持外围内。
恢复	RDS-EVENT-0361	备用数据库实例的恢复已开始。	备用数据库实例在恢复过程中会重新构建。数据库性能在恢复过程中会受到影响。
恢复	RDS-EVENT-0362	备用数据库实例的恢复已完成。	备用数据库实例在恢复过程中会重新构建。数据库性能在恢复过程中会受到影响。
还原	RDS-EVENT-0019	已从数据库实例 *name* 还原为 *name*。	已从时间点备份中恢复数据库实例。
安全	RDS-EVENT-0068	解密 hsm 分区密码以更新实例。	RDS 正在解密 Amazon CloudHSM 分区密码，以对数据库实例进行更新。有关更多信息，请参阅《Amazon CloudHSM 用户指南》中的使用 Amazon CloudHSM 的 Oracle 数据库透明数据加密（TDE）。

类别	RDS 事件 ID	消息	备注
安全修补	RDS-EVENT-0230	系统更新可用于您的数据库实例。有关应用更新的信息，请参阅《RDS 用户指南》中的“维护数据库实例”。	推出了新的操作系统更新。 一个新的、次要版本的操作系统更新可用于您的数据库实例。有关应用更新的信息，请参阅为 RDS 数据库实例更新操作系统。

数据库参数组事件

下表显示的是数据库参数组为源类型时的事件类型和事件列表。

类别	RDS 事件 ID	消息	备注
配置更改	RDS-EVENT-0037	已使用 apply 方法 *method* 将参数 *name* 更新为 *value*。	无

数据库安全组事件

下表显示了数据库安全组为源类型时的事件类别和事件列表。

> **Note**
>
> 数据库安全组是 EC2-Classic 的资源。EC2-Classic 已于 2022 年 8 月 15 日停用。如果您尚未从 EC2-Classic 迁移到 VPC，建议您尽快迁移。有关更多信息，请参阅《Amazon EC2 用户指南》中的从 EC2-Classic 迁移到 VPC 和博客 EC2-Classic Networking is Retiring – Here’s How to Prepare。

类别	RDS 事件 ID	消息	备注
配置更改	RDS-EVENT-0038	对安全组应用了更改。	无

类别	RDS 事件 ID	消息	备注
失败	RDS-EVENT-0039	撤消 *user* 的授权。	*user* 拥有的安全组不存在。针对安全组的授权已被吊销，因为它无效。

数据库快照事件

下表显示了数据库快照为源类型时的事件类别和事件列表。

类别	RDS 事件 ID	消息	备注
创建	RDS-EVENT-0040	正在创建手动快照。	无
创建	RDS-EVENT-0042	已创建手动快照。	无
创建	RDS-EVENT-0090	正在创建自动快照。	无
创建	RDS-EVENT-0091	已创建自动快照。	无
删除	RDS-EVENT-0041	已删除用户快照。	无
通知	RDS-EVENT-0059	已开始从区域 *name* 复制快照 *name*。	这是一个跨区域快照副本。
通知	RDS-EVENT-0060	在 *number* 分钟内完成了从区域 *name* 复制快照 *name*。	这是一个跨区域快照副本。
通知	RDS-EVENT-0061	已取消来自区域 *name* 的 *name* 快照复制请求。	这是一个跨区域快照副本。
通知	RDS-EVENT-0159	快照导出任务失败。	无
notification	RDS-EVENT-0160	快照导出任务已取消。	无
notification	RDS-EVENT-0161	快照导出任务已完成。	无

类别	RDS 事件 ID	消息	备注
notification	RDS-EVENT-0196	已开始复制区域 *name* 中的快照 *name*。	这是一个本地快照副本。
通知	RDS-EVENT-0197	已完成复制区域 *name* 中的快照 *name*。	这是一个本地快照副本。
通知	RDS-EVENT-0190	已取消区域 *name* 中的 *name* 快照复制请求。	这是一个本地快照副本。
还原	RDS-EVENT-0043	已从快照 *name* 还原。	正在从数据库快照中恢复数据库实例。

RDS Proxy 事件

下表显示了 RDS Proxy 为源类型时的事件类别和事件列表。

类别	RDS 事件 ID	消息	备注
配置更改	RDS-EVENT-0204	RDS 修改了数据库代理 *name*。	无
配置更改	RDS-EVENT-0207	RDS 修改了数据库代理 *name* 的端点。	无
配置更改	RDS-EVENT-0213	RDS 检测到已添加数据库实例并自动将其添加到数据库代理 *name* 的目标组。	无
配置更改	RDS-EVENT-0214	RDS 检测到已删除数据库实例 *name* 并自动将其从数据库代理 *name* 的目标组 *name* 中删除。	无
配置更改	RDS-EVENT-0215	RDS 检测到已删除数据库集群 *name* 并自动将其从数	无

类别	RDS 事件 ID	消息	备注
		据库代理 *name* 的目标组 *name* 中删除。	
创建	RDS-EVENT-0203	RDS 创建了数据库代理 *name*。	无
创建	RDS-EVENT-0206	RDS 为数据库代理 *name* 创建了端点 *name*。	无
删除	RDS-EVENT-0205	RDS 删除了数据库代理 *name*。	无
删除	RDS-EVENT-0208	RDS 删除了数据库代理 *name* 的端点 *name*。	无
失败	RDS-EVENT-0243	RDS 无法为代理 *name* 预调配容量，因为您的子网 *name* 中没有足够的 IP 地址可用。要解决此问题，请确保您的子网具有最少的未使用 IP 地址数（如 RDS 代理文档中所建议）。	要确定您的实例类的建议数量，请参阅[计划 IP 地址容量](#)。
失败	RDS-EVENT-0275	RDS 限制了一些与数据库代理*##*的连接。从客户端到代理的并发连接请求数已超出了限制。	无

蓝绿部署事件

下表显示了蓝绿部署为源类型时的事件类型和事件列表。

有关蓝绿部署的更多信息，请参阅[使用 Amazon RDS 蓝绿部署进行数据库更新](#)。

类别	Amazon RDS 事件 ID	消息	备注
创建	RDS-EVENT-0244	蓝绿部署任务已完成。您可以对绿色环境数据库进行更多修改或切换部署。	无
失败	RDS-EVENT-0245	由于 *reason*，蓝/绿部署创建失败。	无
删除	RDS-EVENT-0246	蓝绿部署已删除。	无
notification	RDS-EVENT-0247	已开始从##切换到##。	无
notification	RDS-EVENT-0248	蓝绿部署已完成切换。	无
失败	RDS-EVENT-0249	蓝绿部署已取消切换。	无
notification	RDS-EVENT-0250	已开始从主/只读副本##切换为##。	无
notification	RDS-EVENT-0251	已完成从主/只读副本##切换为##。将##重命名为#####，将##重命名为##。	无
失败	RDS-EVENT-0252	由于##，取消了从主/只读副本##到##的切换。	无
notification	RDS-EVENT-0307	已启动将从##切换为##的序列同步。使用序列时切换可能会导致停机时间延长。	无
notification	RDS-EVENT-0308	已完成将从##切换为##的序列同步。	无
失败	RDS-EVENT-0310	由于序列无法同步，因此取消了将从##切换到##的序列同步。	无

类别	Amazon RDS 事件 ID	消息	备注
notification	RDS-EVENT-0405	您的存储卷正在初始化。	无
notification	RDS-EVENT-0406	您的存储卷已初始化。	无
notification	RDS-EVENT-0409	*message*	无

自定义引擎版本事件

下表显示了自定义引擎版本为源类型时的事件类别和事件列表。

类别	Amazon RDS 事件 ID	消息	备注
创建	RDS-EVENT-0316	正在准备创建自定义引擎版本*##*。整个创建过程最多可能需要四个小时才能完成。	无
创建	RDS-EVENT-0317	正在创建自定义引擎版本*##*。	无
创建	RDS-EVENT-0318	正在验证自定义引擎版本*##*。	无
创建	RDS-EVENT-0319	已成功创建自定义引擎版本*##*。	无
创建	RDS-EVENT-0320	由于内部问题，RDS 无法创建自定义引擎版本*##*。我们正在解决该问题，如有必要，我们将与您联系。如需进一步帮助，请联系 [Amazon Premium Support/](https://aws.amazon.com/premiumsupport/)。	无

类别	Amazon RDS 事件 ID	消息	备注
失败	RDS-EVENT-0198	创建自定义引擎版本 *name* 失败。*message*	*message* 包括有关失败的详细信息，例如缺少文件。
失败	RDS-EVENT-0277	删除自定义引擎版本 *name* 时失败。*message*	*message* 包括有关失败的详细信息。
还原	RDS-EVENT-0352	时间点还原支持的最大数据库数量已更改。	*message* 包括有关此事件的详细信息。

监控 Amazon RDS 日志文件

每个 RDS 数据库引擎都会生成日志，您可以访问这些日志以进行审计和故障排除。日志的类型取决于数据库引擎。

可以使用 Amazon Web Services Management Console、Amazon Command Line Interface（Amazon CLI）或 Amazon RDS API 访问数据库实例的数据库日志。您无法查看、监控或下载事务日志。

主题

查看和列出数据库日志文件

您可以使用 Amazon Web Services Management Console 查看 Amazon RDS 数据库引擎的数据库日志文件。您可使用 Amazon CLI 或 Amazon RDS API 列出可下载或监控的日志文件。

> **Note**
>
> 如果无法查看现有 RDS for Oracle 数据库实例的日志文件的列表，请重启该实例以查看列表。

控制台

要查看数据库日志文件，请执行以下操作

1. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。

2. 在导航窗格中，选择 Databases (数据库)。
3. 选择要查看其日志文件的数据库实例的名称。
4. 选择 Logs & events (日志和事件) 选项卡。
5. 向下滚动到日志部分。
6. （可选）输入搜索词以筛选结果。
7. 选择要查看的日志，然后选择 View（查看）。

Amazon CLI

要列出数据库实例的可用数据库日志文件，请使用 Amazon CLI `describe-db-log-files` 命令。

以下示例将返回名为 `my-db-instance` 的数据库实例的日志文件列表。

Example

```
aws rds describe-db-log-files --db-instance-identifier my-db-instance
```

RDS API

要列出数据库实例的可用数据库日志文件，请使用 Amazon RDS API `DescribeDBLogFiles` 操作。

下载数据库日志文件

您可使用 Amazon Web Services Management Console、Amazon CLI 或 API 下载数据库日志文件。

控制台

要下载数据库日志文件，请执行以下操作

1. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。
3. 选择要查看其日志文件的数据库实例的名称。
4. 选择 Logs & events (日志和事件) 选项卡。
5. 向下滚动到日志部分。
6. 在 Logs (日志) 部分中，选择要下载的日志旁边的按钮，然后选择 Download (下载)。

7. 打开提供的链接的上下文 (右键单击) 菜单，然后选择将链接另存为。输入您希望保存日志文件的位置，然后选择保存。

Amazon CLI

要下载数据库日志文件，请使用 Amazon CLI 命令 download-db-log-file-portion。默认情况下，该命令仅下载日志文件的最新部分。但是，您可以通过指定参数 `--starting-token 0` 下载整个文件。

以下示例演示如何下载一个名为 log/ERROR.4 的日志文件的全部内容并将其存储在一个名为 errorlog.txt 的本地文件中。

Example

对于 Linux、macOS 或 Unix：

```
aws rds download-db-log-file-portion \
    --db-instance-identifier myexampledb \
    --starting-token 0 --output text \
    --log-file-name log/ERROR.4 > errorlog.txt
```

对于 Windows：

```
aws rds download-db-log-file-portion ^
    --db-instance-identifier myexampledb ^
    --starting-token 0 --output text ^
    --log-file-name log/ERROR.4 > errorlog.txt
```

RDS API

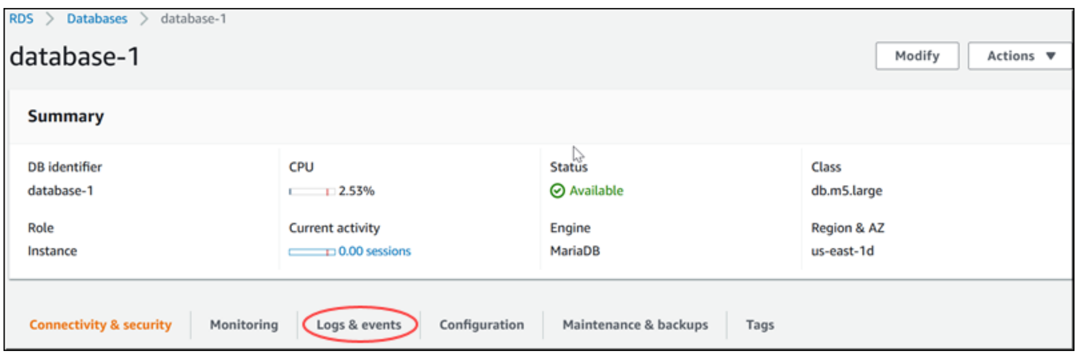
要下载数据库日志文件，请使用 Amazon RDS API `DownloadDBLogFilePortion` 操作。

监视数据库日志文件

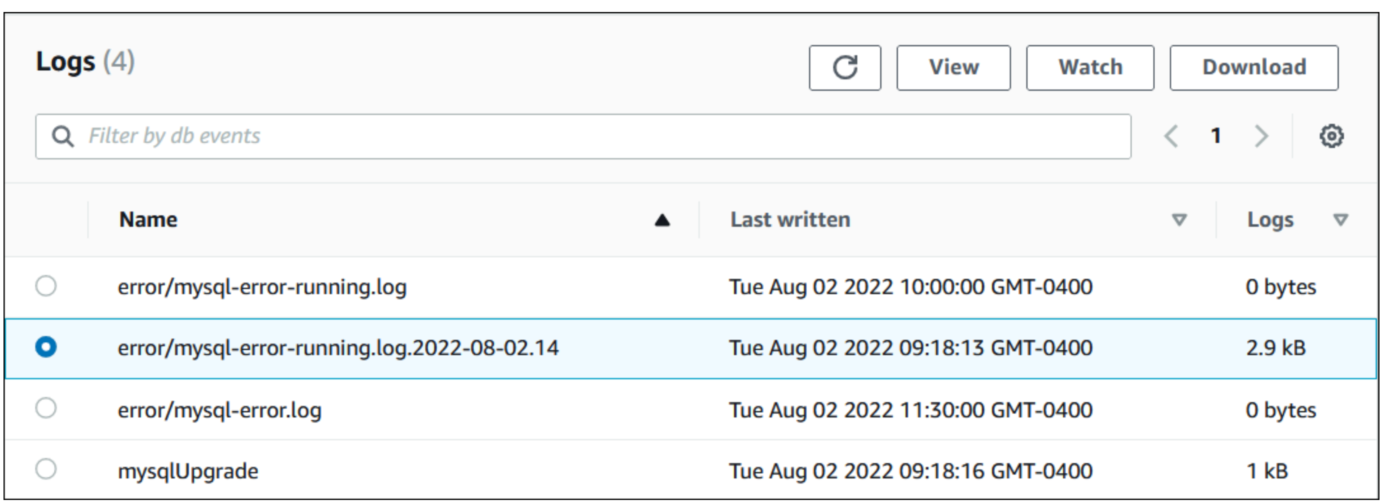
监视数据库日志文件等同于在 UNIX 或 Linux 系统上跟踪该文件。您可使用 Amazon Web Services Management Console监控日志文件。RDS 每 5 秒刷新一次日志的跟踪。

要监视数据库日志文件，请执行以下操作

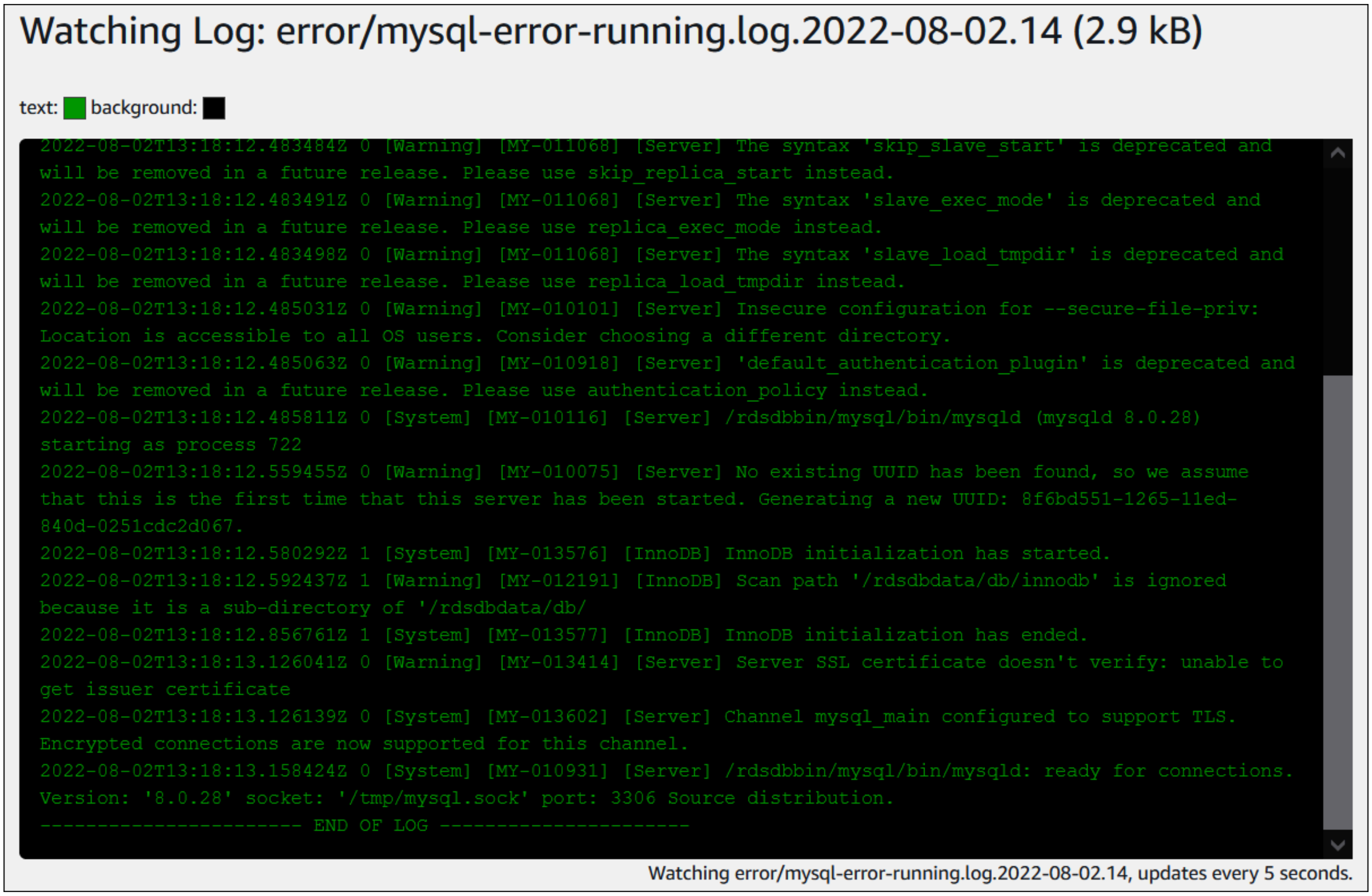
1. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。
3. 选择要查看其日志文件的数据库实例的名称。
4. 选择 Logs & events (日志和事件) 选项卡。



5. 在 Logs (日志) 部分中，选择一个日志文件，然后选择 Watch (监视)。



RDS 显示日志的跟踪，如以下 MySQL 示例所示。



将数据库日志发布到 Amazon CloudWatch Logs

在本地数据库中，数据库日志驻留在文件系统上。Amazon RDS 不向主机提供对数据库实例的文件系统上的数据库日志的访问权限。出于此原因，Amazon RDS 可让您将数据库日志导出到 Amazon CloudWatch Logs。利用 CloudWatch Logs，您可以对日志数据执行实时分析。您还可以将数据存储在高持久性存储中，并使用 CloudWatch Logs 代理管理数据。

主题

- RDS 与 CloudWatch Logs 集成概述
- 决定将哪些日志发布到 CloudWatch Logs
- 指定要发布到 CloudWatch Logs 的日志
- 在 CloudWatch Logs 中搜索和筛选您的日志

RDS 与 CloudWatch Logs 集成概述

在 CloudWatch Logs 中，日志流 是共享同一个源的一系列日志事件。CloudWatch Logs 中每个独立的日志源构成一个独立的日志流。日志组是一组具有相同保留期、监控和访问控制设置的日志流。

Amazon RDS 将您的数据库实例日志记录持续流式传输到日志组。例如，对于您发布的每种类型的日志，您具有日志组 `/aws/rds/instance/`*`instance_name`*`/`*`log_type`*。此日志组与生成日志的数据库实例位于同一 Amazon 区域中。

Amazon 无限期地保留已发布到 CloudWatch Logs 的日志数据，除非您指定了保留期。有关更多信息，请参阅更改 CloudWatch Logs 中的日志数据保留。

决定将哪些日志发布到 CloudWatch Logs

每个 RDS 数据库引擎都支持自己的一组日志。要了解数据库引擎的选项，请查看以下主题：

- the section called “将 Db2 日志发布到 Amazon CloudWatch Logs”
- the section called “将 MariaDB 日志发布到 Amazon CloudWatch Logs”
- the section called “将 MySQL 日志发布到 Amazon CloudWatch Logs”
- the section called “将 Oracle 日志发布到 Amazon CloudWatch Logs”
- the section called “将 PostgreSQL 日志发布到 Amazon CloudWatch Logs”
- the section called “将 SQL Server 日志发布到 Amazon CloudWatch Logs”

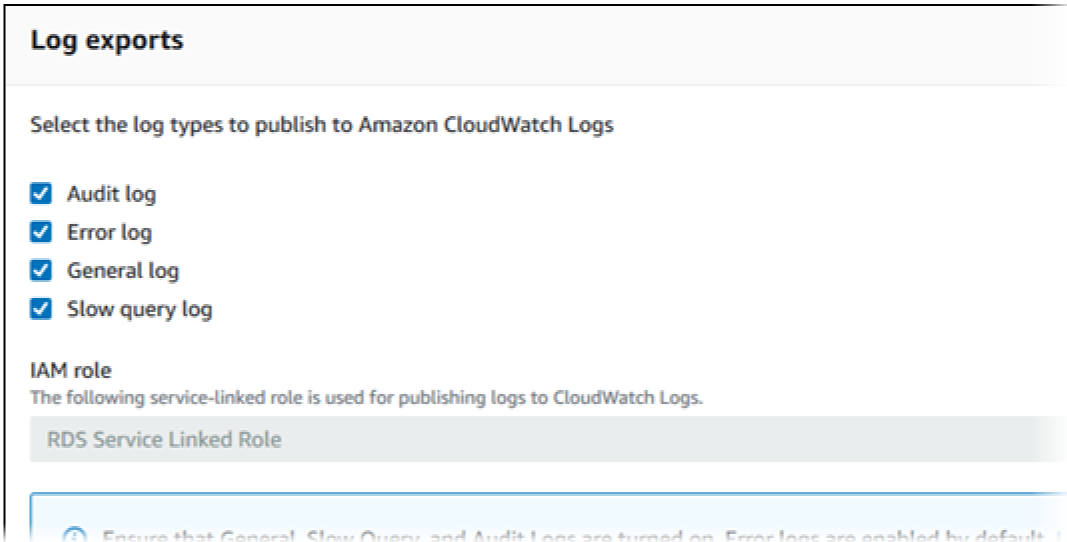
指定要发布到 CloudWatch Logs 的日志

您可以在控制台中指定要发布哪些日志。确保您在 Amazon Identity and Access Management（IAM）中具有服务相关角色。有关服务相关角色的更多信息，请参阅将服务相关角色用于 Amazon RDS。

指定要发布的日志

1. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases（数据库）。
3. 请执行以下任一操作：
   - 选择创建数据库。
   - 从列表中选择数据库，然后选择 Modify（修改）。
4. 在 Logs exports（日志导出）中，选择要发布哪些日志。

以下示例指定 RDS for MySQL 数据库实例的审计日志、错误日志、常规日志，以及慢速查询日志。



在 CloudWatch Logs 中搜索和筛选您的日志

您可以使用 CloudWatch Logs 控制台搜索满足指定条件的日志条目。您可以通过 RDS 控制台访问日志，这会引导您进入 CloudWatch Logs 控制台，也可以直接从 CloudWatch Logs 控制台访问日志。

使用 RDS 控制台搜索 RDS 日志

1. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases（数据库）。
3. 选择数据库实例。
4. 选择配置。
5. 在 Published logs（已发布日志）下，选择要查看的数据库日志。

使用 CloudWatch Logs 控制台搜索 RDS 日志

1. 通过 https://console.aws.amazon.com/cloudwatch/ 打开 CloudWatch 控制台。
2. 在导航窗格中，选择 Log groups（日志组）。
3. 在筛选框中，输入 **`/aws/rds`**。
4. 对于 Log Groups，选择包含要搜索的日志流的日志组的名称。
5. 对于 Log Streams，选择要搜索的日志流的名称。

6. 在 Log events（日志事件）下，输入要使用的筛选条件语法。

有关更多信息，请参阅《Amazon CloudWatch Logs 用户指南》中的搜索和筛选日志数据。有关介绍如何监控 RDS 日志的博客教程，请参阅使用 Amazon CloudWatch Logs、Amazon Lambda 和 Amazon SNS 为 Amazon RDS 构建主动式数据库监控。

使用 REST 读取日志文件内容

Amazon RDS 提供允许访问数据库实例日志文件的 REST 终端节点。如果您需要编写应用程序来流式传输 Amazon RDS 日志文件内容，则这很有用。

语法如下：

```
GET /v13/downloadCompleteLogFile/DBInstanceIdentifier/LogFileName HTTP/1.1
Content-type: application/json
host: rds.region.amazonaws.com
```

以下参数为必需参数：

- *DBInstanceIdentifier* — 包含要下载的日志文件的数据库实例的名称。
- *LogFileName* — 要下载的日志文件的名称。

响应将包含流形式的请求日志文件的内容。

以下示例为 us-west-2 区域中名为 sample-sql 的数据库实例下载名为 log/ERROR.6 的日志文件。

```
GET /v13/downloadCompleteLogFile/sample-sql/log/ERROR.6 HTTP/1.1
host: rds.us-west-2.amazonaws.com
X-Amz-Security-Token: AQoDYXdzEIH//////////
wEa0AIXLhngC5zp9CyB1R6abwKrXHVR5efnAVN3XvR7IwqKYalFSn6UyJuEFTft9nObglx4QJ+GXV9cpACkETq=
X-Amz-Date: 20140903T233749Z
X-Amz-Algorithm: AWS4-HMAC-SHA256
X-Amz-Credential: AKIADQKE4SARGYLE/20140903/us-west-2/rds/aws4_request
X-Amz-SignedHeaders: host
X-Amz-Content-SHA256: e3b0c44298fc1c229afbf4c8996fb92427ae41e4649b934de495991b7852b855
X-Amz-Expires: 86400
X-Amz-Signature: 353a4f14b3f250142d9afc34f9f9948154d46ce7d4ec091d0cdabbcf8b40c558
```

如果您指定的数据库实例不存在，则响应将包含以下错误：

- `DBInstanceNotFound` — *`DBInstanceIdentifier`* 不引用现有数据库实例。(HTTP 状态代码: 404)

Amazon RDS for Db2 数据库日志文件

您可以使用 Amazon RDS 控制台、Amazon CLI 或 RDS API 访问 RDS for Db2 诊断日志和通知日志。有关查看、下载和监视基于文件的数据库日志的更多信息，请参阅 [监控 Amazon RDS 日志文件](#)。

主题

保留计划

日志文件每天都会轮换，并且会在重新启动数据库实例时轮换。以下是 Amazon RDS 上的 RDS for Db2 日志的保留计划。

日志类型	保留计划
诊断日志	Db2 会删除实例级配置中保留设置之外的日志。Amazon RDS 将 `diagsize` 参数设置为 1000。
通知日志	Db2 会删除实例级配置中保留设置之外的日志。Amazon RDS 将 `diagsize` 参数设置为 1000。

将 Db2 日志发布到 Amazon CloudWatch Logs

使用 RDS for Db2，您可以将诊断和通知日志事件直接发布到 Amazon CloudWatch Logs。使用 CloudWatch Logs 分析日志数据，然后使用 CloudWatch 创建警报和查看指标。

利用 CloudWatch Logs，您可以：

- 将日志存储在具有您定义的保留期间的高持久性存储空间中。
- 搜索并筛选日志数据。
- 在账户间共享日志数据。
- 将日志导出到 Amazon S3。
- 将数据流式传输到 Amazon Opensearch 服务。

- 使用 Amazon Kinesis Data Streams 实时处理日志数据。有关更多信息，请参阅《面向 SQL 的适用于 Apache Flink 的亚马逊托管服务的应用程序开发人员指南》中的使用 Amazon CloudWatch Logs。

Amazon RDS 将每个 RDS for Db2 数据库日志作为日志组中的单独数据库流进行发布。例如，如果您发布诊断日志和通知日志，则诊断数据存储在 `/aws/rds/instance/my_instance/diagnostic` 日志组的诊断日志流中，通知日志数据存储在 `/aws/rds/instance/my_instance/notify` 日志组中。

> **Note**
> 默认情况下，没有启用将 RDS for Db2 日志发布到 CloudWatch Logs 的功能。不支持发布自我调整内存管理器（STMM）和优化器统计数据日志。所有区域都支持将 RDS for Db2 日志发布到 CloudWatch Logs，亚太地区（香港）除外。

控制台

从 Amazon Web Services Management Console将 RDS for Db2 日志发布到 CloudWatch Logs

1. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)，然后选择要修改的数据库实例。
3. 选择修改。
4. 在 Log exports (日志导出) 部分中，选择要开始发布到 CloudWatch Logs 的日志。

   您可以选择 diag.log 和/或 notify.log。
5. 选择继续，然后选择摘要页面上的修改数据库实例。

Amazon CLI

要发布 RDS for Db2 日志，您可以使用具有以下参数的 `modify-db-instance` 命令：

- `--db-instance-identifier`
- `--cloudwatch-logs-export-configuration`

> **Note**
>
> 对 `--cloudwatch-logs-export-configuration` 选项进行的更改始终立即应用于数据库实例。因此，`--apply-immediately` 和 `--no-apply-immediately` 选项没有影响。

您还可以使用以下命令来发布 RDS for Db2 日志：

- [`create-db-instance`]
- [`restore-db-instance-from-db-snapshot`]
- [`restore-db-instance-to-point-in-time`]

Example

以下示例将创建一个启用了 CloudWatch Logs 发布的 RDS for Db2 数据库实例。`--enable-cloudwatch-logs-exports` 值是一个可以包括 `diag.log` 和/或 `notify.log` 的 JSON 字符串数组。

对于 Linux、macOS 或 Unix：

```
aws rds create-db-instance \
    --db-instance-identifier mydbinstance \
    --enable-cloudwatch-logs-exports '["diag.log","notify.log"]' \
    --db-instance-class db.m4.large \
    --engine db2-se
```

对于 Windows：

```
aws rds create-db-instance ^
    --db-instance-identifier mydbinstance ^
    --enable-cloudwatch-logs-exports "[\"diag.log\",\"notify.log\"]" ^
    --db-instance-class db.m4.large ^
    --engine db2-se
```

> **Note**
>
> 使用 Windows 命令提示符时，必须在 JSON 代码中转义双引号 (")，方法是使用反斜杠 (\) 作为其前缀。

Example

以下示例修改现有的 RDS for Db2 数据库实例，以将日志文件发布到 CloudWatch Logs。`--cloudwatch-logs-export-configuration` 值是 JSON 对象。此对象的键是 `EnableLogTypes`，其值是可以包括 `diag.log` 和/或 `notify.log` 的字符串数组。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier mydbinstance \
    --cloudwatch-logs-export-configuration '{"EnableLogTypes":
["diag.log","notify.log"]}'
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier mydbinstance ^
    --cloudwatch-logs-export-configuration "{\"EnableLogTypes\":[\"diag.log\",
\"notify.log\"]}"
```

> **Note**
>
> 使用 Windows 命令提示符时，必须在 JSON 代码中转义双引号 (")，方法是使用反斜杠 (\) 作为其前缀。

Example

以下示例修改现有的 RDS for Db2 数据库实例，以禁用将诊断日志文件发布到 CloudWatch Logs 的功能。`--cloudwatch-logs-export-configuration` 值是 JSON 对象。此对象的键是 `DisableLogTypes`，其值是可以包括 `diag.log` 和/或 `notify.log` 的字符串数组。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier mydbinstance \
    --cloudwatch-logs-export-configuration '{"DisableLogTypes":["diag.log"]}'
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier mydbinstance ^
    --cloudwatch-logs-export-configuration "{\"DisableLogTypes\":[\"diag.log\"]}"
```

Note

使用 Windows 命令提示符时，必须在 JSON 代码中转义双引号 (")，方法是使用反斜杠 (\) 作为其前缀。

MariaDB 数据库日志文件

您可以监控 MariaDB 错误日志、慢速查询日志、IAM 数据库身份验证错误日志和常规日志。MariaDB 错误日志默认生成，您可以通过在数据库参数组中设置参数来生成慢速查询日志和一般日志。Amazon RDS 将轮换所有 MariaDB 日志文件；每种类型的间隔如下所示。

您可直接通过 Amazon RDS 控制台、Amazon RDS API、Amazon RDS CLI 或Amazon开发工具包监视 MariaDB 日志。您还可通过将 MariaDB 日志引向主数据库中的数据库表并查询该表，访问这些日志。可使用 mysqlbinlog 实用程序下载二进制日志。

有关查看、下载和监视基于文件的数据库日志的更多信息，请参阅 监控 Amazon RDS 日志文件。

主题

访问 MariaDB 错误日志

MariaDB 错误日志会写入到 `<host-name>.err` 文件中。您可以使用 Amazon RDS 控制台查看此文件，还可以使用 Amazon RDS API、Amazon RDS CLI 或 Amazon SDK 检索日志。`<host-name>.err` 文件每 5 分钟刷新一次，其内容附加到 `mysql-error-running.log` 中。`mysql-error-running.log` 文件每小时轮换一次，保留过去 24 小时内每小时生成的文件。每个日志文件的名称上都会附有生成时间的信息 (以 UTC 时间标记)。日志文件还会拥有时间戳，帮助您确定日志项的写入时间。

MariaDB 仅在启动、关闭和遇到错误时向错误日志写入内容。数据库实例可以运行数小时或者数天，而不向错误日志中写入新项。如果您看不到最近的条目，则是因为服务器未遇到导致生成日志条目的错误。

访问 MariaDB 慢速查询日志和一般日志

通过设置数据库参数组中的参数，可以将 MariaDB 慢速查询日志和一般日志写入文件或者数据库表中。有关创建和修改数据库参数组的信息，请参阅 Amazon RDS 的参数组。您必须先设置这些参数，然后才能在 Amazon RDS 控制台中或使用 Amazon RDS API、Amazon CLI 或Amazon开发工具包查看慢速查询日志或常规日志。

可通过使用下面列表中的参数来控制 MariaDB 日志记录：

- `slow_query_log` 或 `log_slow_query`：要创建慢速查询日志，请设置为 1。默认值是 0。
- `general_log`：要创建一般日志，请设置为 1。默认值是 0。
- `long_query_time` 或 `log_slow_query_time`：要防止在慢速查询日志中记录快速运行的查询，请指定需要记录的最短查询运行时间值，以秒为单位。默认值为 10 秒；最小值为 0。如果 log_output = FILE，则可以指定精确到微秒的浮点值。如果 log_output = TABLE，则必须指定精确到秒的整数值。系统只记录运行时间超过 `long_query_time` 或 `log_slow_query_time` 值的查询。例如，将 `long_query_time` 或 `log_slow_query_time` 设置为 0.1 可防止记录任何运行时间少于 100 毫秒的查询。
- `log_queries_not_using_indexes`：若要将所有不使用索引的查询记录到慢速查询日志，请将该参数设置为 1。默认值为 0。将记录不使用索引的查询，即使它们的运行时间小于 `long_query_time` 参数的值。
- `log_output` *`option`*：您可为 `log_output` 参数指定下列选项之一：
  - TABLE（默认）– 将一般查询写入 `mysql.general_log` 表，将慢速查询写入 `mysql.slow_log` 表。
  - FILE– 将一般查询日志和慢速查询日志写入文件系统。日志文件每小时轮换一次。
  - NONE– 禁用日志记录。

启用了日志记录时，Amazon RDS 会定期轮换表日志或删除日志文件。这是一种预防措施，用于降低大型日志文件阻止数据库使用或影响性能的可能性。`FILE` 和 `TABLE` 日志记录按如下所示进行轮换和删除：

- 启用了 `FILE` 日志记录时，会每小时检查日志文件并删除 24 小时之前的日志文件。在一些情况下，删除之后的剩余日志文件的总体大小可能超过了数据库实例的分配空间的 2% 阈值。在这些情况下，将删除最大的日志文件，直到日志文件大小不再超过此阈值。
- 启用了 `TABLE` 日志记录时，在某些情况下，日志表每 24 小时轮换一次。如果表日志使用的空间大于所分配存储空间的 20%，则会执行此轮换。如果所有日志的总体大小超过 10GB，也会执行此轮换。如果用于数据库实例的空间量大于数据库实例的分配存储空间的 90%，则减小日志轮换的阈

值。如果表日志使用的空间大于分配存储空间的 10%，则轮换日志表。如果所有日志的总体大小超过 5GB，也会轮换这些日志表。

轮换日志表时，会将当前日志表复制到备份日志表，随后删除当前日志表中的条目。如果备份日志表已存在，则先将其删除，然后将当前日志表复制到备份。如果需要，您可以查询备份日志表。`mysql.general_log` 表的备份日志表名为 `mysql.general_log_backup`。`mysql.slow_log` 表的备份日志表名为 `mysql.slow_log_backup`。

您可以通过调用 `mysql.general_log` 过程来轮换 `mysql.rds_rotate_general_log` 表。您可以通过调用 `mysql.slow_log` 过程来轮换 `mysql.rds_rotate_slow_log` 表。

表日志在数据库版本升级期间会进行轮换。

Amazon RDS 将在 Amazon RDS 事件中记录 `TABLE` 和 `FILE` 日志轮换并向您发送通知。

要通过 Amazon RDS 控制台、Amazon RDS API、Amazon RDS CLI 或Amazon开发工具包使用日志，请将 `log_output` 参数设置为 FILE。就像 MariaDB 错误日志一样，这些日志文件也每小时轮换一次。将保留过去 24 小时内生成的日志文件。

有关慢速查询日志和一般日志的更多信息，请参阅 MariaDB 文档中的以下主题：

- 慢速查询日志
- 一般查询日志

将 MariaDB 日志发布到 Amazon CloudWatch Logs

您可以配置 MariaDB 数据库实例以将日志数据发布到 Amazon CloudWatch Logs 中的日志组。利用 CloudWatch Logs，可以对日志数据进行实时分析并使用 CloudWatch 创建警报和查看指标。您可以使用 CloudWatch Logs 在高持久性存储中存储日志记录。

Amazon RDS 将每个 MariaDB 数据库日志作为日志组中的单独数据库流进行发布。例如，假设您将导出功能配置为包括慢速查询日志。然后，慢速查询数据将存储在 `/aws/rds/instance/my_instance/slowquery` 日志组中的慢速查询日志流中。

默认情况下，启用错误日志。下表总结了其他 MariaDB 日志的要求。

日志	要求
审核日志	数据库实例必须使用具有 MARIADB_AUDIT_PLUGIN 选项的自定义选项组。
常规日志	数据库实例必须使用具有参数设置 general_log = 1 的自定义参数组才能启用常规日志。
慢速查询日志	数据库实例必须使用具有参数设置 slow_query_log = 1 或 log_slow_query = 1 的自定义参数组才能启用慢速查询日志。
IAM 数据库身份验证错误日志	您必须通过创建或修改数据库实例来启用数据库实例的日志类型 iam-db-auth-error 。
日志输出	数据库实例必须使用具有参数设置 log_output = FILE 的自定义参数组，才能将日志写入文件系统并将日志发布到 CloudWatch Logs。

控制台

从控制台将 MariaDB 日志发布到 CloudWatch Logs

1. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)，然后选择要修改的数据库实例。
3. 选择修改。
4. 在 Log exports (日志导出) 部分中，选择要开始发布到 CloudWatch Logs 的日志。
5. 选择继续，然后选择摘要页面上的修改数据库实例。

Amazon CLI

您可以使用 Amazon CLI 来发布 MariaDB 日志。您可以调用带以下参数的 `modify-db-instance` 命令：

- `--db-instance-identifier`
- `--cloudwatch-logs-export-configuration`

> **Note**
>
> 对 `--cloudwatch-logs-export-configuration` 选项进行的更改始终立即应用于数据库实例。因此，`--apply-immediately` 和 `--no-apply-immediately` 选项没有影响。

您还可以通过调用以下 Amazon CLI 命令来发布 MariaDB 日志：

- [`create-db-instance`]
- [`restore-db-instance-from-db-snapshot`]
- [`restore-db-instance-from-s3`]
- [`restore-db-instance-to-point-in-time`]

通过以下选项运行上述 Amazon CLI 命令之一：

- `--db-instance-identifier`
- `--enable-cloudwatch-logs-exports`
- `--db-instance-class`
- `--engine`

根据您运行的 Amazon CLI 命令，可能需要其他选项。

Example

以下示例修改现有的 MariaDB 数据库实例以将日志文件发布到 CloudWatch Logs。`--cloudwatch-logs-export-configuration` 值是 JSON 对象。此对象的键是 `EnableLogTypes`，其值是具有 `audit`、`error`、`general` 和 `slowquery` 的任意组合的字符串的数组。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier mydbinstance \
    --cloudwatch-logs-export-configuration '{"EnableLogTypes":
["audit","error","general","slowquery"]}'
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier mydbinstance ^
    --cloudwatch-logs-export-configuration '{"EnableLogTypes":
["audit","error","general","slowquery"]}'
```

Example

以下命令会创建一个 MariaDB 数据库实例并将日志文件发布到 CloudWatch Logs。`--enable-cloudwatch-logs-exports` 值是 JSON 字符串数组。这些字符串可以是 `audit`、`error`、`general` 和 `slowquery` 的任意组合。

对于 Linux、macOS 或 Unix：

```
aws rds create-db-instance \
    --db-instance-identifier mydbinstance \
    --enable-cloudwatch-logs-exports '["audit","error","general","slowquery"]' \
    --db-instance-class db.m4.large \
    --engine mariadb
```

对于 Windows：

```
aws rds create-db-instance ^
    --db-instance-identifier mydbinstance ^
    --enable-cloudwatch-logs-exports '["audit","error","general","slowquery"]' ^
    --db-instance-class db.m4.large ^
    --engine mariadb
```

RDS API

您可以使用 RDS API 来发布 MariaDB 日志。您可以使用以下参数调用 [ModifyDBInstance](#) 操作：

- `DBInstanceIdentifier`
- `CloudwatchLogsExportConfiguration`

> **Note**
>
> 对 `CloudwatchLogsExportConfiguration` 参数进行的更改始终立即应用于数据库实例。因此，`ApplyImmediately` 参数没有影响。

您还可以通过调用以下 RDS API 操作来发布 MariaDB 日志：

- `CreateDBInstance`
- `RestoreDBInstanceFromDBSnapshot`
- `RestoreDBInstanceFromS3`
- `RestoreDBInstanceToPointInTime`

使用以下参数运行这些 RDS API 操作之一：

- `DBInstanceIdentifier`
- `EnableCloudwatchLogsExports`
- `Engine`
- `DBInstanceClass`

根据您运行的 Amazon CLI 命令，可能需要其他参数。

MariaDB 的日志轮换和保留

启用了日志记录时，Amazon RDS 会定期轮换表日志或删除日志文件。这是一种预防措施，用于降低大型日志文件阻止数据库使用或影响性能的可能性。

MariaDB 慢速查询日志、错误日志和一般日志文件的大小限制为不超过为数据库实例分配的存储空间的 2%。为了维护此阈值，日志每小时自动轮换一次，并且将删除 24 小时之前生成的日志文件。如果在删除旧日志文件后，日志文件的总体大小超出此阈值，则将删除最大的日志文件，直到日志文件大小不再超出此阈值。

Amazon RDS 会轮换大于 10 MB 的 IAM 数据库身份验证错误日志文件。Amazon RDS 会移除存在时间超过五天或大于 100 MB 的 IAM 数据库身份验证错误日志文件。

管理基于表的 MariaDB 日志

您可以将一般查询日志和慢速查询日志定向到数据库实例上的表。为此，请创建数据库参数组并将 log_output 服务器参数设置为 TABLE。系统随后会将一般查询记录到 mysql.general_log 表，并将慢速查询记录到 mysql.slow_log 表。可以查询表来访问日志信息。启用此日志记录功能会增加写入数据库的数据量，导致性能降低。

默认情况下，一般日志和慢速查询日志处于禁用状态。要启用将日志记录到表的功能，还必须将以下服务器参数设置为 1：

- general_log
- slow_query_log 或 log_slow_query

日志表将不断增大，直至通过将相应的参数重置为 0 来关闭相应的日志记录活动。随着时间的推移，通常会累积大量的数据，这些数据会占用相当大比例的分配存储空间。Amazon RDS 不支持截断日志表，但您可以移动这些表的内容。表的交替会将表的内容保存到备份表，然后创建一个新的空日志表。可用以下命令行过程手动轮换日志表，其中 PROMPT> 表示命令提示符：

```
PROMPT> CALL mysql.rds_rotate_slow_log;
PROMPT> CALL mysql.rds_rotate_general_log;
```

要完全移除旧数据并回收磁盘空间，请连续两次调用相应的程序。

配置 MariaDB 二进制日志记录

二进制日志是一组日志文件，其中包含有关对 MariaDB 服务器实例所做的数据修改的信息。二进制日志包含以下信息：

- 描述数据库更改的事件，例如表创建或行修改
- 有关更新数据的各语句的持续时间的信息
- 本应更新但未更新数据的语句的事件

复制过程中发送的二进制日志记录语句。一些恢复操作也需要用到这些语句。有关更多信息，请参阅 MariaDB 文档中的 Binary Log。

自动备份功能会判断是否为 MariaDB 开启或关闭二进制日志记录。您有以下选项：

打开二进制日志记录

将备份保留期设置为非零正值。

关闭二进制日志记录

将备份保留期设置为 0。

有关更多信息，请参阅 启用自动备份。

Amazon RDS 上的 MariaDB 支持基于行、基于语句和混合二进制日志记录格式。默认二进制日志记录的格式是混合的。有关不同的 MariaDB 二进制日志格式的详细信息，请参阅 MariaDB 文档中的二进制日志格式。

如果您计划使用复制，则二进制日志记录格式很重要。这是因为它确定在源中记录和发送到复制目标的数据更改的记录。有关用于复制的不同二进制日志记录格式的优缺点的信息，请参阅 MySQL 文档中的基于语句和基于行的复制的优点和缺点。

> **Important**
>
> 将二进制日志记录格式设置为基于行会生成非常大的二进制日志文件。大型二进制日志文件会减少可用于数据库实例的存储空间量。它们还会增加执行数据库实例还原操作所需的时间。
> 基于语句的复制可能在源数据库实例和只读副本之间导致不一致。有关更多信息，请参阅 MariaDB 文档中的基于语句的复制的不安全语句。
> 启用二进制日志记录会增加数据库实例的写入磁盘 I/O 操作数。您可以使用 `WriteIOPS` CloudWatch 指标监控 IOPS 使用情况。

设置 MariaDB 二进制日志记录格式

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择参数组。
3. 选择您要修改的数据库实例所使用的参数组。

   您无法修改默认参数组。如果数据库实例使用默认参数组，则创建新的参数组并将其与数据库实例关联。

   有关数据库参数组的更多信息，请参阅 Amazon RDS 的参数组。

4. 对于 Parameter group actions (参数组操作)，选择 Edit (编辑)。
5. 将 `binlog_format` 参数设置为您选择的二进制日志记录格式（ROW、STATEMENT 或 MIXED）。

   您可以通过将数据库实例的备份保留期设置为零来关闭二进制日志记录，但这会禁用每日自动备份。禁用自动备份会关闭或禁用 `log_bin` 会话变量。这样将禁用 RDS for MariaDB 数据库实例上的二进制日志记录，这反过来又会将数据库中的 `binlog_format` 会话变量重置为默认值 `ROW`。我们建议您不要禁用备份。有关设置备份保留期的更多信息，请参阅数据库实例的设置。
6. 选择保存更改以保存对数据库参数组的更新。

由于 `binlog_format` 参数在 RDS for MariaDB 中是动态的，因此您无需重启数据库实例即可应用更改。

Important

更改数据库参数组会影响使用该参数组的所有数据库实例。如果要为 Amazon 区域中的不同 MariaDB 数据库实例指定不同的二进制日志记录格式，数据库实例必须使用不同的数据库参数组。这些参数组标识不同的日志记录格式。为每个数据库实例分配相应的数据库参数组。

访问 MariaDB 二进制日志

您可以使用 mysqlbinlog 实用工具从 MariaDB 数据库实例中以文本格式下载二进制日志。二进制日志将下载到本地计算机。有关使用 mysqlbinlog 实用工具的更多信息，请转到 MariaDB 文档中的使用 mysqlbinlog。

要针对 Amazon RDS 实例运行 mysqlbinlog 实用工具，请使用下列选项：

- 指定 `--read-from-remote-server` 选项。
- `--host`：指定该实例所在的端点中的 DNS 名称。
- `--port`：指定该实例使用的端口。
- `--user`：指定已授予了复制从属权限的 MariaDB 用户。
- `--password`：指定用户的密码，或忽略密码值以让实用工具将提示您输入密码。
- `--result-file`：指定接收输出的本地文件。
- 指定一个或多个二进制日志文件的名称。要获取可用日志的列表，请使用 SQL 命令 SHOW BINARY LOGS。

有关 mysqlbinlog 选项的更多信息，请参阅 MariaDB 文档中的 mysqlbinlog 选项。

以下是 示例：

对于 Linux、macOS 或 Unix：

```
mysqlbinlog \
    --read-from-remote-server \
    --host=mariadbinstance1.1234abcd.region.rds.amazonaws.com \
    --port=3306  \
    --user ReplUser \
    --password <password> \
    --result-file=/tmp/binlog.txt
```

对于 Windows：

```
mysqlbinlog ^
    --read-from-remote-server ^
    --host=mariadbinstance1.1234abcd.region.rds.amazonaws.com ^
    --port=3306  ^
    --user ReplUser ^
    --password <password> ^
    --result-file=/tmp/binlog.txt
```

mazon RDS 通常会尽快清除二进制日志。但是，二进制日志必须仍在实例上提供以供 mysqlbinlog 访问。要指定 RDS 保留二进制日志的小时数，请使用 `mysql.rds_set_configuration` 存储过程。指定一个时间段，以便您有足够的时间来下载日志。设置保留期后，监视数据库实例的存储用量以确认保留的二进制日志不会占用太多存储空间。

以下示例将保留期设置为 1 天。

```
call mysql.rds_set_configuration('binlog retention hours', 24);
```

要显示当前设置，请使用 `mysql.rds_show_configuration` 存储过程。

```
call mysql.rds_show_configuration;
```

启用 MariaDB 二进制日志注释

在 MariaDB 数据库实例中，您可以使用 Annotate_rows 事件注释某个行事件，并在注释中包含导致行事件的 SQL 查询的副本。此方法提供了与对 RDS for MySQL 数据库实例启用 binlog_rows_query_log_events 参数类似的功能。

您可以通过创建自定义参数组并将 binlog_annotate_row_events 参数设置为 **1** 来全局启用二进制日志注释。您还可以通过调用 SET SESSION binlog_annotate_row_events = 1 在会话级别启用注释。使用 replicate_annotate_row_events 将二进制日志注释复制到副本实例（如果已对其启用二进制日志记录）。使用这些设置需要特殊权限。

下面是 MariaDB 中的基于行的事务的示例。基于行的日志记录的使用通过将事务隔离级别设置为“已提交读”来触发。

```
CREATE DATABASE IF NOT EXISTS test;
USE test;
CREATE TABLE square(x INT PRIMARY KEY, y INT NOT NULL) ENGINE = InnoDB;
SET SESSION TRANSACTION ISOLATION LEVEL READ COMMITTED;
BEGIN
INSERT INTO square(x, y) VALUES(5, 5 * 5);
COMMIT;
```

没有注释时，事务的二进制日志条目与下面的类似：

```
BEGIN
/*!*/;
# at 1163
# at 1209
#150922  7:55:57 server id 1855786460  end_log_pos 1209         Table_map:
 `test`.`square` mapped to number 76
#150922  7:55:57 server id 1855786460  end_log_pos 1247         Write_rows: table id 76
 flags: STMT_END_F
### INSERT INTO `test`.`square`
### SET
###   @1=5
###   @2=25
# at 1247
#150922  7:56:01 server id 1855786460  end_log_pos 1274         Xid = 62
COMMIT/*!*/;
```

以下语句为此相同事务启用会话级注释，然后在提交事务后禁用这些注释：

```
CREATE DATABASE IF NOT EXISTS test;
USE test;
CREATE TABLE square(x INT PRIMARY KEY, y INT NOT NULL) ENGINE = InnoDB;
SET SESSION TRANSACTION ISOLATION LEVEL READ COMMITTED;
SET SESSION binlog_annotate_row_events = 1;
BEGIN;
INSERT INTO square(x, y) VALUES(5, 5 * 5);
COMMIT;
SET SESSION binlog_annotate_row_events = 0;
```

当有注释时，事务的二进制日志条目与下面的类似：

```
BEGIN
/*!*/;
# at 423
# at 483
# at 529
#150922  8:04:24 server id 1855786460  end_log_pos 483  Annotate_rows:
#Q> INSERT INTO square(x, y) VALUES(5, 5 * 5)
#150922  8:04:24 server id 1855786460  end_log_pos 529  Table_map: `test`.`square`
 mapped to number 76
#150922  8:04:24 server id 1855786460  end_log_pos 567  Write_rows: table id 76 flags:
 STMT_END_F
### INSERT INTO `test`.`square`
### SET
###   @1=5
###   @2=25
# at 567
#150922  8:04:26 server id 1855786460  end_log_pos 594  Xid = 88
COMMIT/*!*/;
```

Amazon RDS for Microsoft SQL Server 数据库日志文件

您可以使用 Amazon RDS 控制台、Amazon CLI 或 RDS API 访问 Microsoft SQL Server 错误日志、代理日志、跟踪文件和转储文件。有关查看、下载和监视基于文件的数据库日志的更多信息，请参阅监控 Amazon RDS 日志文件。

保留计划

日志文件每天都会轮换，并且会在重新启动数据库实例时轮换。以下是 Amazon RDS 上的 Microsoft SQL Server 日志的保留计划。

日志类型	保留计划
错误日志	最多可保留 30 个错误日志。Amazon RDS 可能会删除超过 7 天的错误日志。
Agent 日志	最多可保留 10 个代理日志。Amazon RDS 可能会删除超过 7 天的代理日志。
跟踪文件	根据数据库实例的跟踪文件保留期保留跟踪文件。默认的跟踪文件保留期为 7 天。要修改数据库实例的跟踪文件保留期，请参阅设置跟踪文件和转储文件的保留期。
转储文件	根据数据库实例的转储文件保留期保留转储文件。默认的转储文件保留期为 7 天。要修改数据库实例的转储文件保留期，请参阅设置跟踪文件和转储文件的保留期。

使用 rds_read_error_log 过程查看 SQL Server 错误日志

您可以使用 Amazon RDS 存储过程 `rds_read_error_log` 查看错误日志和代理日志。有关更多信息，请参阅"查看错误和代理日志"。

将 SQL Server 日志发布到 Amazon CloudWatch Logs

使用 Amazon RDS for SQL Server，您可以将错误和代理日志事件直接发布到 Amazon CloudWatch Logs。使用 CloudWatch Logs 分析日志数据，然后使用 CloudWatch 创建警报和查看指标。

利用 CloudWatch Logs，您可以：

- 将日志存储在具有您定义的保留期间的高持久性存储空间中。

- 搜索并筛选日志数据。
- 在账户间共享日志数据。
- 将日志导出到 Amazon S3。
- 将数据流式传输到 Amazon Opensearch 服务。
- 使用 Amazon Kinesis Data Streams 实时处理日志数据。有关更多信息，请参阅《面向 SQL 的适用于 Apache Flink 的亚马逊托管服务的应用程序开发人员指南》中的使用 Amazon CloudWatch Logs。

Amazon RDS 将每个 SQL Server 数据库日志作为日志组中的单独数据库流进行发布。例如，如果您发布代理日志和错误日志，则错误数据将存储在 `/aws/rds/instance/my_instance.node1/error` 日志组中的错误日志流中，而代理日志数据将存储在 `/aws/rds/instance/my_instance.node1/agent` 日志组中。

对于多可用区数据库实例，Amazon RDS 将数据库日志作为日志组中的两个独立流发布。例如，如果您发布错误日志，则错误数据将分别存储在错误日志流 `/aws/rds/instance/my_instance.node1/error` 和 `/aws/rds/instance/my_instance.node2/error` 中。日志流在失效转移期间不会发生变化，每个节点的错误日志流可能包含来自主实例或辅助实例的错误日志。使用多可用区，系统会自动为 `/aws/rds/instance/my_instance/rds-events` 创建日志流，以存储诸如数据库实例失效转移等事件数据。

> **Note**
>
> 默认情况下，没有启用将 SQL Server 日志发布到 CloudWatch Logs 的功能。不支持发布跟踪和转储文件。所有区域都支持将 SQL Server 日志发布到 CloudWatch Logs，亚太地区（香港）除外。

控制台

从Amazon Web Services Management Console将 SQL Server 数据库日志发布到 CloudWatch Logs

1. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)，然后选择要修改的数据库实例。
3. 选择修改。
4. 在 Log exports (日志导出) 部分中，选择要开始发布到 CloudWatch Logs 的日志。

   您可以选择代理日志、错误日志或两者。

5. 选择继续，然后选择摘要页面上的修改数据库实例。

Amazon CLI

要发布 SQL Server 日志，您可以使用具有以下参数的 `modify-db-instance` 命令：

- `--db-instance-identifier`
- `--cloudwatch-logs-export-configuration`

> **Note**
>
> 对 `--cloudwatch-logs-export-configuration` 选项进行的更改始终立即应用于数据库实例。因此，`--apply-immediately` 和 `--no-apply-immediately` 选项没有影响。

您还可以使用以下命令来发布 SQL Server 日志：

- `create-db-instance`
- `restore-db-instance-from-db-snapshot`
- `restore-db-instance-to-point-in-time`

Example

以下示例将创建一个启用了 CloudWatch Logs 发布的 SQL Server 数据库实例。`--enable-cloudwatch-logs-exports` 值是一个可以包括 `error` 和/或 `agent` 的 JSON 字符串数组。

对于 Linux、macOS 或 Unix：

```
aws rds create-db-instance \
    --db-instance-identifier mydbinstance \
    --enable-cloudwatch-logs-exports '["error","agent"]' \
    --db-instance-class db.m4.large \
    --engine sqlserver-se
```

对于 Windows：

```
aws rds create-db-instance ^
    --db-instance-identifier mydbinstance ^
```

```
    --enable-cloudwatch-logs-exports "[\"error\",\"agent\"]" ^
    --db-instance-class db.m4.large ^
    --engine sqlserver-se
```

> **Note**
>
> 使用 Windows 命令提示符时，必须在 JSON 代码中转义双引号 (")，方法是使用反斜杠 (\) 作为其前缀。

Example

以下示例修改现有的 SQL Server 数据库实例以将日志文件发布到 CloudWatch Logs。`--cloudwatch-logs-export-configuration` 值是 JSON 对象。此对象的键是 `EnableLogTypes`，其值是可以包括 `error` 和/或 `agent` 的字符串数组。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier mydbinstance \
    --cloudwatch-logs-export-configuration '{"EnableLogTypes":["error","agent"]}'
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier mydbinstance ^
    --cloudwatch-logs-export-configuration "{\"EnableLogTypes\":[\"error\",\"agent\"]}"
```

> **Note**
>
> 使用 Windows 命令提示符时，必须在 JSON 代码中转义双引号 (")，方法是使用反斜杠 (\) 作为其前缀。

Example

以下示例修改现有的 SQL Server 数据库实例以禁用将代理日志文件发布到 CloudWatch Logs 的功能。`--cloudwatch-logs-export-configuration` 值是 JSON 对象。此对象的键是 `DisableLogTypes`，其值是可以包括 `error` 和/或 `agent` 的字符串数组。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier mydbinstance \
    --cloudwatch-logs-export-configuration '{"DisableLogTypes":["agent"]}'
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier mydbinstance ^
    --cloudwatch-logs-export-configuration "{\"DisableLogTypes\":[\"agent\"]}"
```

> **Note**
>
> 使用 Windows 命令提示符时，必须在 JSON 代码中转义双引号 (")，方法是使用反斜杠 (\) 作为其前缀。

MySQL 数据库日志文件

您可直接通过 Amazon RDS 控制台、Amazon RDS API、Amazon CLI 或 Amazon 开发工具包监控 MySQL 日志。您还可通过将 MySQL 日志引向主数据库中的数据库表并查询该表，访问这些日志。可使用 mysqlbinlog 实用程序下载二进制日志。

有关查看、下载和监视基于文件的数据库日志的更多信息，请参阅 监控 Amazon RDS 日志文件。

主题

- RDS for MySQL 数据库日志概览
- 将 MySQL 日志发布到 Amazon CloudWatch Logs
- 将 MySQL 日志输出发送到表
- 为单可用区数据库配置 RDS for MySQL 二进制日志记录
- 为多可用区数据库集群配置 MySQL 二进制日志记录
- 访问 MySQL 二进制日志

RDS for MySQL 数据库日志概览

您可以监控以下类型的 RDS for MySQL 日志文件：

- 错误日志
- 慢速查询日志
- 常规日志
- 审计日志
- 实例日志
- IAM 数据库身份验证错误日志

默认情况下，系统会生成 RDS for MySQL 错误日志。您可以通过在数据库参数组中设置参数来生成慢速查询日志和一般日志。

主题

- RDS for MySQL 错误日志
- RDS for MySQL 慢速查询日志和一般日志
- MySQL 审计日志
- RDS for MySQL 的日志轮换和保留

- 重做日志的大小限制

RDS for MySQL 错误日志

RDS for MySQL 在 `mysql-error.log` 文件中写入错误。每个日志文件的名称上都会附有生成时间的信息 (以 UTC 时间标记)。日志文件还会拥有时间戳，帮助您确定日志项的写入时间。

RDS for MySQL 仅在启动、关闭和遇到错误时才向错误日志写入内容。数据库实例可以运行数小时或者数天，而不向错误日志中写入新项。如果看不到最近的条目，则原因是服务器未遇到导致生成日志条目的错误。

根据设计，将筛选错误日志，以便仅显示错误等意外事件。但是，错误日志还包含一些未显示的其他数据库信息，例如查询进度。因此，即使没有任何实际错误，由于持续进行的数据库活动，错误日志的大小也可能会增加。虽然您可能会在 Amazon Web Services Management Console 中看到错误日志有一定大小（以字节或千字节为单位），但当您下载它们时，它们可能为 0 字节。

RDS for MySQL 每 5 分钟将 `mysql-error.log` 写入磁盘。它会将日志的内容附加到 `mysql-error-running.log` 中。

RDS for MySQL 每小时轮换一次 `mysql-error-running.log` 文件。它会保留过去两周生成的日志。

> **Note**
>
> Amazon RDS 和 Aurora 之间的日志保留期不同。

RDS for MySQL 慢速查询日志和一般日志

可以将 RDS for MySQL 慢速查询日志和一般日志写入文件或数据库表中。为此，请设置数据库参数组中的参数。有关创建和修改数据库参数组的信息，请参阅 Amazon RDS 的参数组。您必须先设置这些参数，然后才能在 Amazon RDS 控制台中或使用 Amazon RDS API、Amazon RDS CLI 或Amazon软件开发工具包查看慢速查询日志或一般日志。

可通过使用下面列表中的参数来控制 RDS for MySQL 日志记录：

- `slow_query_log`：要创建慢速查询日志，请设置为 1。默认值为 0。
- `general_log`：要创建一般日志，请设置为 1。默认值是 0。
- `long_query_time`：要防止在慢速查询日志中记录快速运行的查询，请指定需要记录的最短查询运行时间值，以秒为单位。默认值为 10 秒；最小值为 0。如果 log_output = FILE，则可以指定精确到

微秒的浮点值。如果 log_output = TABLE，则必须指定精确到秒的整数值。系统只记录运行时间超过 `long_query_time` 值的查询。例如，将 `long_query_time` 设置为 0.1 可防止记录任何运行时间少于 100 毫秒的查询。

- `log_queries_not_using_indexes`：要将所有不使用索引的查询记录到慢速查询日志，请设置为 1。将记录不使用索引的查询，即使它们的运行时间小于 `long_query_time` 参数的值。默认值是 0。
- `log_output` *option*：您可为 `log_output` 参数指定下列选项之一。
  - TABLE（默认）— 将一般查询写入 `mysql.general_log` 表，将慢速查询写入 `mysql.slow_log` 表。
  - FILE– 将一般查询日志和慢速查询日志写入文件系统。
  - NONE– 禁用日志记录。

要使慢速查询数据出现在 Amazon CloudWatch Logs 中，必须满足以下条件：

- 必须将 CloudWatch Logs 配置为包含慢速查询日志。
- 必须启用 `slow_query_log`。
- `log_output` 必须设置为 `FILE`。
- 查询花费的时间必须比为 `long_query_time` 配置的时间长。

有关慢速查询日志和一般日志的更多信息，请参阅 MySQL 文档中的以下主题：

- [慢速查询日志]
- [一般查询日志]

MySQL 审计日志

若要访问审计日志，数据库实例必须使用具有 `MARIADB_AUDIT_PLUGIN` 选项的自定义选项组。有关更多信息，请参阅 [MySQL 的 MariaDB 审计插件支持]。

RDS for MySQL 的日志轮换和保留

启用了日志记录时，Amazon RDS 会定期轮换表日志或删除日志文件。这是一种预防措施，用于降低大型日志文件阻止数据库使用或影响性能的可能性。RDS for MySQL 按如下方式处理轮换和删除：

- MySQL 慢速查询日志、错误日志和一般日志文件的大小限制为不超过为数据库实例分配的存储空间的 2%。为了维护此阈值，日志每小时自动轮换一次。MySQL 会删除超过两周的日志文件。如果在

删除旧日志文件后，日志文件的总体大小超出此阈值，则将删除最早的日志文件，直到日志文件大小不再超出此阈值。

- 启用了 FILE 日志记录时，系统会每小时检查一次日志文件，并删除超过两周的日志文件。在一些情况下，删除之后的剩余日志文件的总体大小可能超过了数据库实例的分配空间的 2% 阈值。在这些情况下，将删除最旧的日志文件，直到日志文件大小不再超过此阈值。
- 启用了 TABLE 日志记录时，在某些情况下，日志表每 24 小时轮换一次。如果表日志使用的空间大于所分配存储空间的 20%，则会执行此轮换。如果所有日志的总体大小超过 10GB，也会执行此轮换。如果用于数据库实例的空间量大于数据库实例的分配存储空间的 90%，则减小日志轮换的阈值。如果表日志使用的空间大于分配存储空间的 10%，则轮换日志表。如果所有日志的总体大小超过 5GB，也会轮换这些日志表。您可以订阅 low storage 事件类别，在轮换日志表以释放空间时，会发送相关通知。有关更多信息，请参阅 使用 Amazon RDS 事件通知。

  轮换日志表时，首先将当前日志表复制到备份日志表。然后，删除当前日志表中的条目。如果备份日志表已存在，则先将其删除，然后将当前日志表复制到备份。如果需要，您可以查询备份日志表。mysql.general_log 表的备份日志表名为 mysql.general_log_backup。mysql.slow_log 表的备份日志表名为 mysql.slow_log_backup。

  您可以通过调用 mysql.general_log 过程来轮换 mysql.rds_rotate_general_log 表。您可以通过调用 mysql.slow_log 过程来轮换 mysql.rds_rotate_slow_log 表。

  表日志在数据库版本升级期间会进行轮换。

要通过 Amazon RDS 控制台、Amazon RDS API、Amazon RDS CLI 或 Amazon 开发工具包使用日志，请将 log_output 参数设置为 FILE。就像 MySQL 错误日志一样，这些日志文件也每小时轮换一次。将保留过去两周内生成的日志文件。请注意，Amazon RDS 和 Aurora 之间的保留期不同。

重做日志的大小限制

对于 RDS for MySQL 版本 8.0.32 及更低版本，此参数的默认值为 256MB。该数量是通过将 innodb_log_file_size 参数的默认值（128MB）乘以 innodb_log_files_in_group 参数的默认值（2）得出的。有关更多信息，请参阅为 Amazon RDS for MySQL 配置参数的最佳实践，第 1 部分：与性能相关的参数。

对于 RDS for MySQL 版本 8.0.33 及更高的次要版本，Amazon RDS 使用 innodb_redo_log_capacity 参数而不是 innodb_log_file_size 参数。innodb_redo_log_capacity 参数的 Amazon RDS 默认值为 2GB。有关更多信息，请参阅 MySQL 文档中的 MySQL 8.0.30 中的更改。

从 MySQL 8.4 开始，Amazon RDS 默认启用 innodb_dedicated_server 参数。使用 innodb_dedicated_server 参数，数据库引擎可计算 innodb_buffer_pool_size 和 innodb_redo_log_capacity 参数。有关更多信息，请参阅 在 MySQL 8.4 中配置缓冲池大小和重做日志容量。

将 MySQL 日志发布到 Amazon CloudWatch Logs

您可以配置 MySQL 数据库实例以将日志数据发布到 Amazon CloudWatch Logs 中的日志组。利用 CloudWatch Logs，可以对日志数据进行实时分析并使用 CloudWatch 创建警报和查看指标。您可以使用 CloudWatch Logs 在高持久性存储中存储日志记录。

Amazon RDS 将每个 MySQL 数据库日志作为日志组中的单独数据库流进行发布。例如，如果您将导出功能配置为包括慢速查询日志，则慢速查询数据将存储在 /aws/rds/instance/*my_instance*/slowquery 日志组中的慢速查询日志流中。

默认情况下，启用错误日志。下表总结了其他 MySQL 日志的要求。

日志	要求
审核日志	数据库实例必须使用具有 MARIADB_AUDIT_PLUGIN 选项的自定义选项组。
常规日志	数据库实例必须使用具有参数设置 general_log = 1 的自定义参数组才能启用常规日志。
慢速查询日志	数据库实例必须使用具有参数设置 slow_query_log = 1 的自定义参数组才能启用慢速查询日志。
IAM 数据库身份验证错误日志	您必须通过创建或修改数据库实例来启用数据库实例的日志类型 iam-db-auth-error 。
日志输出	数据库实例必须使用具有参数设置 log_output = FILE 的自定义参数组，才能将日志写入文件系统并将日志发布到 CloudWatch Logs。

控制台

使用控制台将 MySQL 日志发布到 CloudWatch Logs

1. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)，然后选择要修改的数据库实例。
3. 选择修改。
4. 在 Log exports (日志导出) 部分中，选择要开始发布到 CloudWatch Logs 的日志。
5. 选择继续，然后选择摘要页面上的修改数据库实例。

Amazon CLI

您可以使用 Amazon CLI 来发布 MySQL 日志。您可以调用带以下参数的 `modify-db-instance` 命令：

- `--db-instance-identifier`
- `--cloudwatch-logs-export-configuration`

> **Note**
> 对 `--cloudwatch-logs-export-configuration` 选项进行的更改始终立即应用于数据库实例。因此，`--apply-immediately` 和 `--no-apply-immediately` 选项没有影响。

您还可以通过调用以下 Amazon CLI 命令来发布 MySQL 日志：

- `create-db-instance`
- `restore-db-instance-from-db-snapshot`
- `restore-db-instance-from-s3`
- `restore-db-instance-to-point-in-time`

通过以下选项运行上述 Amazon CLI 命令之一：

- `--db-instance-identifier`
- `--enable-cloudwatch-logs-exports`

- `--db-instance-class`
- `--engine`

根据您运行的 Amazon CLI 命令，可能需要其他选项。

Example

以下示例修改现有的 MySQL 数据库实例以将日志文件发布到 CloudWatch Logs。`--cloudwatch-logs-export-configuration` 值是 JSON 对象。此对象的键是 `EnableLogTypes`，其值是具有 `audit`、`error`、`general` 和 `slowquery` 的任意组合的字符串的数组。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier mydbinstance \
    --cloudwatch-logs-export-configuration '{"EnableLogTypes":
["audit","error","general","slowquery"]}'
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier mydbinstance ^
    --cloudwatch-logs-export-configuration '{"EnableLogTypes":
["audit","error","general","slowquery"]}'
```

Example

以下示例将创建一个 MySQL 数据库实例并将日志文件发布到 CloudWatch Logs。`--enable-cloudwatch-logs-exports` 值是 JSON 字符串数组。这些字符串可以是 `audit`、`error`、`general` 和 `slowquery` 的任意组合。

对于 Linux、macOS 或 Unix：

```
aws rds create-db-instance \
    --db-instance-identifier mydbinstance \
    --enable-cloudwatch-logs-exports '["audit","error","general","slowquery"]' \
    --db-instance-class db.m4.large \
    --engine MySQL
```

对于 Windows：

```
aws rds create-db-instance ^
    --db-instance-identifier mydbinstance ^
    --enable-cloudwatch-logs-exports '["audit","error","general","slowquery"]' ^
    --db-instance-class db.m4.large ^
    --engine MySQL
```

RDS API

您可以使用 RDS API 来发布 MySQL 日志。您可以使用以下参数调用 ModifyDBInstance 操作：

- DBInstanceIdentifier
- CloudwatchLogsExportConfiguration

> **Note**
>
> 对 CloudwatchLogsExportConfiguration 参数进行的更改始终立即应用于数据库实例。因此，ApplyImmediately 参数没有影响。

您还可以通过调用以下 RDS API 操作来发布 MySQL 日志：

- CreateDBInstance
- RestoreDBInstanceFromDBSnapshot
- RestoreDBInstanceFromS3
- RestoreDBInstanceToPointInTime

使用以下参数运行这些 RDS API 操作之一：

- DBInstanceIdentifier
- EnableCloudwatchLogsExports
- Engine
- DBInstanceClass

根据您运行的 Amazon CLI 命令，可能需要其他参数。

将 MySQL 日志输出发送到表

可通过创建数据库参数组并将 log_output 服务器参数设置为 TABLE，将一般日志和慢速查询日志引向数据库实例上的表。系统随后会将一般查询记录到 mysql.general_log 表，并将慢速查询记录到 mysql.slow_log 表。可以查询表来访问日志信息。启用此日志记录功能会增加写入数据库的数据量，导致性能降低。

默认情况下，一般日志和慢速查询日志处于禁用状态。要启用将日志记录到表的功能，必须将 general_log 和 slow_query_log 服务器参数设置为 1。

日志表将不断增大，直至通过将相应的参数重置为 *0* 来关闭相应的日志记录活动。随着时间的推移，通常会累积大量的数据，这些数据会占用相当大比例的分配存储空间。Amazon RDS 不允许截断日志表，但可以移动其中的内容。表的交替会将表的内容保存到备份表，然后创建一个新的空日志表。可用以下命令行过程手动轮换日志表，其中 PROMPT> 表示命令提示符：

```
PROMPT> CALL mysql.rds_rotate_slow_log;
PROMPT> CALL mysql.rds_rotate_general_log;
```

要完全移除旧数据并回收磁盘空间，请连续两次调用相应的程序。

为单可用区数据库配置 RDS for MySQL 二进制日志记录

二进制日志是一组日志文件，其中包含对 MySQL 服务器实例所做的数据修改的信息。二进制日志包含以下信息：

- 描述数据库更改的事件，例如表创建或行修改
- 有关更新数据的各语句的持续时间的信息
- 本应更新但未更新数据的语句的事件

复制过程中发送的二进制日志记录语句。一些恢复操作也需要用到这些语句。有关更多信息，请参阅 MySQL 文档中的二进制日志。

自动备份功能会判断是否为 MySQL 开启或关闭二进制日志记录。您有以下选项：

打开二进制日志记录

  将备份保留期设置为非零正值。

关闭二进制日志记录

将备份保留期设置为 0。

有关更多信息，请参阅 启用自动备份。

Amazon RDS 上的 MariaDB 支持基于行、基于语句和混合二进制日志记录格式。我们建议混合使用，除非您需要特定的二进制日志格式。有关不同的 MySQL 二进制日志格式的详细信息，请参阅 MySQL 文档中的 Binary Logging Formats。

如果您计划使用复制，则二进制日志记录格式很重要，因为它确定了在源中记录和发送到复制目标的数据更改记录。有关用于复制的不同二进制日志记录格式的优缺点的信息，请参阅 MySQL 文档中的基于语句和基于行的复制的优点和缺点。

> **Important**
>
> 在 MySQL 8.0.34 中，MySQL 弃用了 `binlog_format` 参数。在更高版本的 MySQL 中，MySQL 计划删除该参数，并且仅支持基于行的复制。因此，我们建议对新的 MySQL 复制设置使用基于行的日志记录。有关更多信息，请参阅 MySQL 文档中的 binlog_format。
> MySQL 版本 8.0 和 8.4 接受参数 `binlog_format`。使用此参数时，MySQL 会发出弃用警告。在将来的主要版本中，MySQL 将移除参数 `binlog_format`。
> 基于语句的复制可能在源数据库实例和只读副本之间导致不一致。有关详细信息，请参阅 MySQL 文档中的确定二进制日志记录中的安全和不安全语句。
> 启用二进制日志记录会增加数据库实例的写入磁盘 I/O 操作数。您可以使用 `WriteIOPS` CloudWatch 指标监控 IOPS 使用情况。

设置 MySQL 二进制日志记录格式

1. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择参数组。
3. 选择要修改的与数据库实例关联的数据库参数组。

   您无法修改默认参数组。如果数据库实例使用默认参数组，则创建新的参数组并将其与数据库实例关联。

   有关参数组的更多信息，请参阅Amazon RDS 的参数组。
4. 从操作中，选择编辑。

5. 将 `binlog_format` 参数设置为您选择的二进制日志记录格式（`ROW`、`STATEMENT` 或 `MIXED`）

   您可以通过将数据库实例的备份保留期设置为零来关闭二进制日志记录，但这会禁用每日自动备份。禁用自动备份会关闭或禁用 `log_bin` 会话变量。这将禁用 RDS for MySQL 数据库实例上的二进制日志记录，这反过来又会将数据库中的 `binlog_format` 会话变量重置为默认值 `ROW`。我们建议您不要禁用备份。有关设置备份保留期的更多信息，请参阅[数据库实例的设置](#)。

6. 选择保存更改以保存对数据库参数组的更新。

由于 `binlog_format` 参数在 RDS for MySQL 中是动态的，因此您无需重启数据库实例即可应用更改。（请注意，在 Aurora MySQL 中，此参数是静态的。有关更多信息，请参阅[配置 Aurora MySQL 二进制日志记录](#)。）

> **Important**
>
> 更改数据库参数组会影响使用该参数组的所有数据库实例。如果要为 Amazon 区域中的不同 MySQL 数据库实例指定不同的二进制日志记录格式，数据库实例必须使用不同的数据库参数组。这些参数组标识不同的日志记录格式。为每个数据库实例分配相应的数据库参数组。

为多可用区数据库集群配置 MySQL 二进制日志记录

Amazon RDS for MySQL 多可用区数据库集群中的二进制日志记录会记录所有数据库更改，以支持复制、时间点故障恢复和审计。在多可用区数据库集群中，二进制日志将辅助节点与主节点同步，从而确保跨可用区的数据一致性并实现无缝失效转移。

为了优化二进制日志记录，Amazon RDS 支持二进制日志事务压缩，这样可以降低二进制日志的存储要求并提高复制效率。

主题

多可用区数据库集群的二进制日志事务压缩

二进制日志事务压缩使用 zstd 算法来减小存储在二进制日志中的事务数据的大小。启用后，MySQL 数据库引擎会将事务有效载荷压缩为单个事件，从而显著减少 I/O 和存储开销。此功能可提高数据库性能，减小二进制日志的大小，并优化在多可用区数据库集群中管理和复制日志的资源使用。

Amazon RDS 通过以下参数为 RDS for MySQL 多可用区数据库集群提供二进制日志事务压缩：

- `binlog_transaction_compression` – 启用 (1) 后，数据库引擎会压缩事务有效载荷，并将其作为单个事件写入二进制日志。这样可减少存储空间使用量并降低 I/O 开销。默认情况下，将禁用此参数。
- `binlog_transaction_compression_level_zstd` – 为二进制日志事务配置 zstd 压缩级别。较高的值会提高压缩比，从而进一步降低存储要求，但会增加压缩的 CPU 和内存使用量。默认值为 3，范围为 1-22。

这些参数允许您根据工作负载特征和资源可用性来微调二进制日志压缩。有关更多信息，请参阅 MySQL 文档中的 Binary Log Transaction Compression。

二进制日志事务压缩具有以下主要优势：

- 压缩会减小二进制日志的大小，对于大型事务或写入量大的工作负载尤其如此。
- 较小的二进制日志可减少网络和 I/O 开销，提高复制性能。
- `binlog_transaction_compression_level_zstd` 参数提供对在压缩比和资源消耗之间权衡的控制。

为多可用区数据库集群配置二进制日志事务压缩

要为 RDS for MySQL 多可用区数据库集群配置二进制日志事务压缩，请修改相关的集群参数设置以满足您的工作负载要求。

控制台

启用二进制日志事务压缩

1. 修改数据库集群参数组，以将 `binlog_transaction_compression` 参数设置为 `1`。
2. （可选）根据您的工作负载要求和资源可用性调整 `binlog_transaction_compression_level_zstd` 参数的值。

有关更多信息，请参阅 the section called “修改数据库集群参数组中的参数”。

Amazon CLI

要使用 Amazon CLI 配置二进制日志事务压缩，请执行 modify-db-cluster-parameter-group 命令。

Example

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-cluster-parameter-group \
  --db-cluster-parameter-group-name your-cluster-parameter-group \
  --parameters
 "ParameterName=binlog_transaction_compression,ParameterValue=1,ApplyMethod=pending-
reboot"
```

对于 Windows：

```
aws rds modify-db-cluster-parameter-group ^
  --db-cluster-parameter-group-name your-cluster-parameter-group ^
  --parameters
 "ParameterName=binlog_transaction_compression,ParameterValue=1,ApplyMethod=pending-
reboot"
```

RDS API

要使用 Amazon RDS API 配置二进制日志事务压缩，请执行 `ModifyDBClusterParameterGroup` 操作。

访问 MySQL 二进制日志

可使用 mysqlbinlog 实用程序从 RDS for MySQL 数据库实例下载或流式传输二进制日志。二进制日志下载到本地计算机，可以执行一些操作，例如使用 mysql 实用程序执行重放日志。有关使用 mysqlbinlog 实用程序的更多信息，请参阅 MySQL 文档中的使用 mysqlbinlog 备份二进制日志文件。

要针对 Amazon RDS 实例运行 mysqlbinlog 实用工具，请使用下列选项：

- `--read-from-remote-server` – 必需。
- `--host` – 来自实例的端点的 DNS 名称。
- `--port` – 实例使用的端口。
- `--user` – 已获得 `REPLICATION SLAVE` 权限的 MySQL 用户。
- `--password` – MySQL 用户的密码，或忽略密码值以让实用程序提示您输入密码。
- `--raw` – 以二进制格式下载文件。
- `--result-file` – 用于接收原始输出的本地文件。
- `--stop-never` – 流式传输二进制日志文件。

- `--verbose` – 使用 `ROW` 二进制日志格式时，包括此选项以将行事件视为伪 SQL 语句。有关 `--verbose` 选项的更多信息，请参阅 MySQL 文档中的 mysqlbinlog 行事件显示。
- 指定一个或多个二进制日志文件的名称。要获取可用日志的列表，请使用 SQL 命令 `SHOW BINARY LOGS`。

有关 mysqlbinlog 选项的更多信息，请参阅 MySQL 文档中的 mysqlbinlog - 处理二进制日志文件的实用程序。

以下示例显示如何使用 mysqlbinlog 实用程序。

对于 Linux、macOS 或 Unix：

```
mysqlbinlog \
    --read-from-remote-server \
    --host=MySQLInstance1.cg034hpkmmjt.region.rds.amazonaws.com \
    --port=3306  \
    --user ReplUser \
    --password \
    --raw \
    --verbose \
    --result-file=/tmp/ \
    binlog.00098
```

对于 Windows：

```
mysqlbinlog ^
    --read-from-remote-server ^
    --host=MySQLInstance1.cg034hpkmmjt.region.rds.amazonaws.com ^
    --port=3306  ^
    --user ReplUser ^
    --password ^
    --raw ^
    --verbose ^
    --result-file=/tmp/ ^
    binlog.00098
```

数据库实例上的二进制日志必须保持可用，以备 mysqlbinlog 实用程序访问。要确保这些日志的可用性，请使用 mysql.rds_set_configuration 存储过程并指定一个时段，以便您有足够的时间来下载日志。如果未设置此配置，Amazon RDS 会尽可能快地清除二进制日志，导致 mysqlbinlog 实用程序检索的二进制日志出现空白。

以下示例将保留期设置为 1 天。

```
call mysql.rds_set_configuration('binlog retention hours', 24);
```

要显示当前设置，请使用 mysql.rds_show_configuration 存储过程。

```
call mysql.rds_show_configuration;
```

Amazon RDS for Oracle 数据库日志文件

可以使用 Amazon RDS 控制台或 API 访问 Oracle 警报日志、审核文件和跟踪文件。有关查看、下载和监视基于文件的数据库日志的更多信息，请参阅 监控 Amazon RDS 日志文件。

提供的 Oracle 审核文件是标准 Oracle 审核文件。Amazon RDS 支持 Oracle 精细审核（FGA）功能。但是，日志访问不提供对存储在 `SYS.FGA_LOG$` 表中并且可通过 `DBA_FGA_AUDIT_TRAIL` 视图访问的 FGA 事件的访问权限。

`DescribeDBLogFiles` API 操作列出可用于数据库实例的 Oracle 日志文件，该操作会忽略 `MaxRecords` 参数，最多返回 1000 个记录。该调用返回 `LastWritten` 作为 POSIX 日期（单位：毫秒）。

主题

- 保留计划
- 使用 Oracle 跟踪文件
- 将 Oracle 日志发布到 Amazon CloudWatch Logs
- 访问警报日志和侦听器日志

保留计划

如果日志文件变得非常大，则 Oracle 数据库引擎可能会轮换这些文件。要保留审核文件或跟踪文件，请下载这些文件。如果您将文件存储在本地，则可以降低 Amazon RDS 存储成本并为您的数据留出更多空间。

下表显示了 Amazon RDS 上的 Oracle 警报日志、审计文件和跟踪文件的保留计划。

日志类型	保留计划
警报日志	文本警报日志每天轮换，并具有由 Amazon RDS 管理的 30 天保留期。XML 警报日志至少保留七天。可以使用 `ALERTLOG` 视图访问此日志。
审核文件	审核文件默认的保留期为七天。Amazon RDS 可能会删除超过七天的审核文件。
跟踪文件	跟踪文件默认的保留期为七天。Amazon RDS 可能会删除超过七天的跟踪文件。

日志类型	保留计划
侦听器日志	侦听器日志默认的保留期为七天。Amazon RDS 可能会删除超过 7 天的侦听器日志。

> **Note**
> 审核文件和跟踪文件共享相同的保留配置。

使用 Oracle 跟踪文件

下面介绍用于创建、刷新、访问和删除跟踪文件的 Amazon RDS 过程。

主题

- 列出文件
- 生成跟踪文件并跟踪会话
- 检索跟踪文件
- 清除跟踪文件

列出文件

可以使用两个过程中的任意一个过程来允许访问 `background_dump_dest` 路径中的任意文件。第一个过程可刷新视图（其中包含当前 `background_dump_dest` 中所有文件的列表）。

```
EXEC rdsadmin.manage_tracefiles.refresh_tracefile_listing;
```

刷新视图后，请查询以下视图来访问结果。

```
SELECT * FROM rdsadmin.tracefile_listing;
```

上述过程的替代方法是，使用 `FROM table` 以类似表的格式流式处理非关系数据以列出数据库目录内容。

```
SELECT * FROM TABLE(rdsadmin.rds_file_util.listdir('BDUMP'));
```

下面的查询显示日志文件的文本。

```
SELECT text FROM
 TABLE(rdsadmin.rds_file_util.read_text_file('BDUMP','alert_dbname.log.date'));
```

在只读副本上，通过查询 V$DATABASE.DB_UNIQUE_NAME 获取 BDUMP 目录的名称。如果唯一名称为 DATABASE_B，则 BDUMP 目录为 BDUMP_B。以下示例查询副本上的 BDUMP 名称，然后使用此名称查询 alert_DATABASE.log.2020-06-23 的内容。

```
SELECT 'BDUMP' || (SELECT regexp_replace(DB_UNIQUE_NAME,'.*(_[A-Z])', '\1') FROM V
$DATABASE) AS BDUMP_VARIABLE FROM DUAL;

BDUMP_VARIABLE
--------------
BDUMP_B

SELECT TEXT FROM
 table(rdsadmin.rds_file_util.read_text_file('BDUMP_B','alert_DATABASE.log.2020-06-23'));
```

生成跟踪文件并跟踪会话

由于没有任何 ALTER SESSION 限制，Oracle 中用于生成跟踪文件的许多标准方法对于 Amazon RDS 数据库实例仍然可用。为需要更大访问权限的跟踪文件提供了以下程序。

Oracle 方法	Amazon RDS 方法
oradebug hanganalyze 3	EXEC rdsadmin.manage_tracefiles.hanganalyze;
oradebug dump systemstate 266	EXEC rdsadmin.manage_tracefiles.dump_systemstate;

您可以使用很多标准方法来跟踪连接到 Amazon RDS 中的 Oracle 数据库实例的各个会话。要启用对会话的跟踪，您可以运行 Oracle 提供的 PL/SQL 程序包中的子程序，如 DBMS_SESSION 和 DBMS_MONITOR。有关更多信息，请参阅 Oracle 文档中的启用会话跟踪。

检索跟踪文件

对于 Amazon RDS 管理的外部表，您可以使用标准 SQL 查询检索 background_dump_dest 中的任何跟踪文件。要使用此方法，必须执行此程序以将此表的位置设置到指定跟踪文件。

例如，可以使用前面提到的 rdsadmin.tracefile_listing 视图列出系统上的所有跟踪文件。然后，可以使用以下过程设置 tracefile_table 视图，从而使视图指向预定的跟踪文件。

```
EXEC
 rdsadmin.manage_tracefiles.set_tracefile_table_location('CUST01_ora_3260_SYSTEMSTATE.trc');
```

以下示例在当前架构中创建了一个外部表，并将表位置设定在提供的文件中。可以使用 SQL 查询将内容检索到本地文件中。

```
SPOOL /tmp/tracefile.txt
SELECT * FROM tracefile_table;
SPOOL OFF;
```

清除跟踪文件

跟踪文件会累积并占用磁盘空间。Amazon RDS 默认清除跟踪文件和超过 7 天的日志文件。可以使用 show_configuration 过程查看和设置跟踪文件保留期。您应运行命令 SET SERVEROUTPUT ON，以便查看配置结果。

以下示例显示了当前的跟踪文件保留期，然后设置了新的跟踪文件保留期。

```
# Show the current tracefile retention
SQL> EXEC rdsadmin.rdsadmin_util.show_configuration;
NAME:tracefile retention
VALUE:10080
DESCRIPTION:tracefile expiration specifies the duration in minutes before tracefiles in
 bdump are automatically deleted.

# Set the tracefile retention to 24 hours:
SQL> EXEC rdsadmin.rdsadmin_util.set_configuration('tracefile retention',1440);
SQL> commit;

#show the new tracefile retention
SQL> EXEC rdsadmin.rdsadmin_util.show_configuration;
NAME:tracefile retention
VALUE:1440
DESCRIPTION:tracefile expiration specifies the duration in minutes before tracefiles in
 bdump are automatically deleted.
```

除定期清除过程外，您可以从 background_dump_dest 中手动删除文件。以下示例显示了如何清除保留时间已超过五分钟的所有文件。

```
EXEC rdsadmin.manage_tracefiles.purge_tracefiles(5);
```

您还可清除与特定模式匹配的所有文件（如果这样做，则不要包括 .trc 等文件扩展名）。以下示例介绍如何清除以 SCHPOC1_ora_5935 开头的所有文件。

```
EXEC rdsadmin.manage_tracefiles.purge_tracefiles('SCHPOC1_ora_5935');
```

将 Oracle 日志发布到 Amazon CloudWatch Logs

您可以配置 RDS for Oracle 数据库实例，以将日志数据发布到 Amazon CloudWatch Logs 中的日志组。利用 CloudWatch Logs，可以对日志数据进行分析并使用 CloudWatch 创建警报和查看指标。您可以使用 CloudWatch Logs 在高持久性存储中存储日志记录。

Amazon RDS 将每个 Oracle 数据库日志作为日志组中的单独数据库流进行发布。例如，如果将导出功能配置为包括审核日志，则审核数据将存储在 /aws/rds/instance/my_instance/audit 日志组中的审核日志流中。下表总结了 RDS for Oracle 将日志发布到 Amazon CloudWatch Logs 的要求。

日志名称	要求	默认
提醒日志	无。您无法禁用此日志。	已启用
跟踪日志	将 `trace_enabled` 参数设置 TRUE 或将其保留为默认值。	TRUE
审计日志	将 `audit_trail` 参数设置为以下允许的任何值： `{ none | os | db [, extended] | xml [, extended] }`	none
侦听器日志	无。您无法禁用此日志。	已启用
Oracle Management Agent 日志	无。您无法禁用此日志。	已启用

此 Oracle Management Agent 日志由下表中显示的日志组组成。

日志名称	CloudWatch 日志组
emctl.log	oemagent-emctl
emdctlj.log	oemagent-emdctlj
gcagent.log	oemagent-gcagent
gcagent_errors.log	oemagent-gcagent-errors
emagent.nohup	oemagent-emagent-nohup
secure.log	oemagent-secure

有关详细信息，请参阅 Oracle 文档中的 查找 Management Agent 日志和跟踪文件。

控制台

从Amazon Web Services Management Console将 Oracle 数据库日志发布到 CloudWatch Logs

1. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)，然后选择要修改的数据库实例。
3. 选择修改。
4. 在 Log exports (日志导出) 部分中，选择要开始发布到 CloudWatch Logs 的日志。
5. 选择继续，然后选择摘要页面上的修改数据库实例。

Amazon CLI

要发布 Oracle 日志，您可以使用具有以下参数的 `modify-db-instance` 命令：

- `--db-instance-identifier`
- `--cloudwatch-logs-export-configuration`

> **Note**
>
> 对 `--cloudwatch-logs-export-configuration` 选项进行的更改始终立即应用于数据库实例。因此，`--apply-immediately` 和 `--no-apply-immediately` 选项没有影响。

您还可以使用以下命令来发布 Oracle 日志：

- [create-db-instance]
- [restore-db-instance-from-db-snapshot]
- [restore-db-instance-from-s3]
- [restore-db-instance-to-point-in-time]

Example

以下示例将创建一个启用了 CloudWatch Logs 发布的 Oracle 数据库实例。`--cloudwatch-logs-export-configuration` 值是 JSON 字符串数组。这些字符串可以是 `alert`、`audit`、`listener` 和 `trace` 的任意组合。

对于 Linux、macOS 或 Unix：

```
aws rds create-db-instance \
    --db-instance-identifier mydbinstance \
    --cloudwatch-logs-export-configuration
 '["trace","audit","alert","listener","oemagent"]' \
    --db-instance-class db.m5.large \
    --allocated-storage 20 \
    --engine oracle-ee \
    --engine-version 19.0.0.0.ru-2024-04.rur-2024-04.r1 \
    --license-model bring-your-own-license \
    --master-username myadmin \
    --manage-master-user-password
```

对于 Windows：

```
aws rds create-db-instance ^
    --db-instance-identifier mydbinstance ^
    --cloudwatch-logs-export-configuration trace alert audit listener oemagent ^
    --db-instance-class db.m5.large ^
    --allocated-storage 20 ^
    --engine oracle-ee ^
    --engine-version 19.0.0.0.ru-2024-04.rur-2024-04.r1 ^
    --license-model bring-your-own-license ^
    --master-username myadmin ^
    --manage-master-user-password
```

Example

以下示例修改现有的 Oracle 数据库实例以将日志文件发布到 CloudWatch Logs。`--cloudwatch-logs-export-configuration` 值是 JSON 对象。此对象的键是 `EnableLogTypes`，其值是具有 `alert`、`audit`、`listener` 和 `trace` 的任意组合的字符串的数组。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier mydbinstance \
    --cloudwatch-logs-export-configuration '{"EnableLogTypes":
["trace","alert","audit","listener","oemagent"]}'
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier mydbinstance ^
    --cloudwatch-logs-export-configuration EnableLogTypes=\"trace\",\"alert\",\"audit
\",\"listener\",\"oemagent\"
```

Example

以下示例修改现有的 Oracle 数据库实例以禁用将审核和侦听器日志文件发布到 CloudWatch Logs 的功能。`--cloudwatch-logs-export-configuration` 值是 JSON 对象。此对象的键是 `DisableLogTypes`，其值是具有 `alert`、`audit`、`listener` 和 `trace` 的任意组合的字符串的数组。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier mydbinstance \
    --cloudwatch-logs-export-configuration '{"DisableLogTypes":["audit","listener"]}'
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier mydbinstance ^
    --cloudwatch-logs-export-configuration DisableLogTypes=\"audit\",\"listener\"
```

RDS API

您可以使用 RDS API 来发布 Oracle 数据库日志。您可以使用以下参数调用 [ModifyDBInstance] 操作：

- `DBInstanceIdentifier`
- `CloudwatchLogsExportConfiguration`

> **Note**
> 对 `CloudwatchLogsExportConfiguration` 参数进行的更改始终立即应用于数据库实例。因此，`ApplyImmediately` 参数没有影响。

您还可以通过调用以下 RDS API 操作来发布 Oracle 日志：

- `CreateDBInstance`
- `RestoreDBInstanceFromDBSnapshot`
- `RestoreDBInstanceFromS3`
- `RestoreDBInstanceToPointInTime`

使用以下参数运行这些 RDS API 操作之一：

- `DBInstanceIdentifier`
- `EnableCloudwatchLogsExports`
- `Engine`
- `DBInstanceClass`

根据您运行的 RDS 操作，可能需要其他参数。

访问警报日志和侦听器日志

您可以使用 Amazon RDS 控制台查看警告日志。您还可以使用以下 SQL 语句。

```
SELECT message_text FROM alertlog;
```

使用 Amazon CloudWatch Logs 访问侦听器日志。

> **Note**
>
> 当警报和侦听器日志超过 10MB 时，Oracle 会轮换这些日志，此时它们不会显示在 Amazon RDS 视图中。

RDS for PostgreSQL 数据库日志文件

您可以监控以下类型的 日志文件：

- PostgreSQL 日志
- 升级日志
- IAM 数据库身份验证错误日志

> **Note**
> 要启用 IAM 数据库身份验证错误日志，您必须先为 RDS for PostgreSQL 数据库实例启用 IAM 数据库身份验证。有关启用 IAM 数据库身份验证的更多信息，请参阅启用和禁用 IAM 数据库身份验证。

RDS for PostgreSQL 将数据库活动记录到原定设置 PostgreSQL 日志文件中。对于本地 PostgreSQL 数据库实例，这些消息本地存储在 `log/postgresql.log` 中。对于 RDS for PostgreSQL 数据库实例，日志文件可用于 Amazon RDS 实例上。也可以通过 Amazon Web Services Management Console访问这些日志，您可以在此处查看或下载它们。原定设置日志记录级别可捕获登录失败、严重服务器错误、死锁和查询失败。

有关如何查看、下载和监视基于文件的数据库日志的更多信息，请参阅 监控 Amazon RDS 日志文件。要了解有关 PostgreSQL 日志的更多信息，请参阅 Working with Amazon RDS and Aurora PostgreSQL logs: Part 1 和 Working with Amazon RDS and Aurora PostgreSQL logs: Part 2。

除了本主题中讨论的标准 PostgreSQL 日志外，RDS for PostgreSQL 还支持 PostgreSQL Audit 扩展（`pgAudit`）。大多数受监管行业和政府机构需要保留对数据所做更改的审计日志或审计跟踪，以符合法律要求。有关安装和使用 pgAudit 的更多信息，请参阅 使用 pgAudit 记录数据库活动。

主题

RDS for PostgreSQL 中的日志记录参数

您可以通过修改各种参数来自定义 RDS for PostgreSQL 数据库实例的日志记录行为。在下表中，您可以找到影响日志存储时间长度、何时轮换日志和是否以 CSV（逗号分隔值）格式输出日志的参数。您还可以找到发送到 STDERR 的文本输出以及其他设置。要更改可修改的参数的设置，请为以下项目使用自定义数据库参数组： RDS for PostgreSQL 实例。有关更多信息，请参阅Amazon RDS 数据库实例的数据库参数组。

参数	默认值	描述
log_destination	stderr	设置日志的输出格式。原定设置为 `stderr`，但您也可以通过向设置中添加 `csvlog` 来指定逗号分隔值（CSV）。有关更多信息，请参阅 设置日志目标（`stderr`、`csvlog`）。
log_filename	postgresql.log.%Y-%m-%d-%H	指定日志文件名的模式。除原定设置外，此参数还支持使用 `postgresql.log.%Y-%m-%d` 和 `postgresql.log.%Y-%m-%d-%H%M` 作为文件名模式。
log_line_prefix	%t:%r:%u@%d:[%p]:	为写入 `stderr` 的每个日志行定义前缀，以记录时间（%t）、远程主机（%r）、用户（%u）、数据库（%d）和进程 ID（%p）。
log_rotation_age	60	此分钟数后，将自动轮换日志文件。您可以在 1 到 1440 分钟的范围内更改此值。有关更多信息，请参阅设置日志文件轮换。
log_rotation_size	–	日志自动轮换的大小（kB）。默认情况下，不使用此参数，因为日志是根据 `log_rotation_age` 参数轮换的。要了解更多信息，请参阅设置日志文件轮换。
rds.log_retention_period	4320	超过指定分钟数的 PostgreSQL 日志将被删除。默认值为 4320 分钟，表示系统将在 3 天后删除日志文件。有关更多信息，请参阅 设置日志保留期。

要识别应用程序问题，您可以在日志中查找查询失败、登录失败、死锁和致命的服务器错误。例如，假设您将旧版应用程序从 Oracle 转换为 Amazon RDS PostgreSQL，但部分查询可能未正确转换。这些格式不正确的查询会生成错误消息，您可以在日志中找到这些错误消息以帮助识别问题。有关日志记录查询的更多信息，请参阅 为您的 RDS for PostgreSQL 数据库实例开启查询日志记录。

在以下主题中，您可以找到有关如何设置各种参数（用于控制 PostgreSQL 日志的基本详细信息）的信息。

主题

设置日志保留期

`rds.log_retention_period` 参数指定 RDS for PostgreSQL 数据库实例将其日志文件保留多长时间。原定设置为 3 天（4320 分钟），但您可以将此值设置为 1 天（1440 分钟）到 7 天（10080 分钟）之间的任意值。确保您的 RDS for PostgreSQL 数据库实例具有足够的存储空间，可将日志文件保留一段时间。

建议您将日志定期发布到 Amazon CloudWatch Logs，以便您可以在日志从 RDS for PostgreSQL 数据库实例中删除很久之后查看和分析系统数据。有关更多信息，请参阅 将 PostgreSQL 日志发布到 Amazon CloudWatch Logs。

设置日志文件轮换

原定设置情况下，Amazon RDS 每小时都会创建新的日志文件。计时由 `log_rotation_age` 参数控制。此参数的原定设置值为 60（分钟），但您可以将其设置为 1 分钟到 24 小时（1440 分钟）之间的任意值。轮换时，系统将创建一个新的不同日志文件。该文件根据 `log_filename` 参数指定的模式命名。

也可以根据日志文件的大小进行轮换（如 `log_rotation_size` 参数指定）。此参数指定在日志达到指定的大小（以千字节为单位）时应轮换日志。对于 RDS for PostgreSQL 数据库实例，`log_rotation_size` 未设置，也就是说，未指定任何值。但是，您可以将该参数设置为 0-2097151KB（千字节）之间。

日志文件名基于在 `log_filename` 参数中指定的文件名模式。此参数的可用设置如下：

- postgresql.log.%Y-%m-%d – 日志文件名的原定设置格式。在日志文件的名称中包括年、月和日。
- postgresql.log.%Y-%m-%d-%H – 在日志文件名格式中包含小时。

有关更多信息，请参阅 PostgreSQL 文档中的 log_rotation_age 和 log_rotation_size。

设置日志目标（**stderr**、**csvlog**）

默认情况下，Amazon RDS PostgreSQL 以标准错误 (stderr) 格式生成日志。此格式是 log_destination 参数的原定设置。每条消息都使用在 log_line_prefix 参数中指定的模式作为前缀。有关更多信息，请参阅 了解 log_line_prefix 参数。

RDS for PostgreSQL 也可以按 csvlog 格式生成日志。csvlog 用于将日志数据作为逗号分隔值（CSV）数据进行分析。例如，假设您使用 log_fdw 扩展将日志作为外部表处理。在 stderr 日志文件上创建的外部表包含一个列，其中包含日志事件数据。通过向 csvlog 参数添加 log_destination，可以获得 CSV 格式的日志文件，其中对外部表的多个列进行了分界。您现在可以更轻松地对日志进行排序和分析。要了解如何将 log_fdw 与 csvlog 结合使用，请参阅 使用 log_fdw 扩展通过 SQL 访问数据库日志。

如果您为此参数指定 csvlog，请注意将同时生成 stderr 和 csvlog 文件。确保监控日志所占用的存储空间，同时考虑 rds.log_retention_period 以及影响日志存储和周转的其他设置。使用 stderr 和 csvlog 会使日志占用的存储空间增加一倍以上。

如果将 csvlog 添加到 log_destination 并想单独恢复到 stderr，则需要重置参数。为此，请打开 Amazon RDS 控制台，然后为您的实例打开自定义数据库参数组。选择 log_destination 参数，选择 Edit parameter（编辑参数），然后选择 Reset（重置）。

有关配置日志记录的更多信息，请参阅使用 Amazon RDS 和 Aurora PostgreSQL 日志：第 1 部分。

了解 log_line_prefix 参数

stderr 日志格式将由 log_line_prefix 参数指定的详细信息作为每条日志消息的前缀。默认值为：

```
%t:%r:%u@%d:[%p]:t
```

从 Aurora PostgreSQL 版本 16 开始，还可以选择：

```
%m:%r:%u@%d:[%p]:%l:%e:%s:%v:%x:%c:%q%a
```

发送到 stderr 的每个日志条目都包含以下基于所选值的信息：

- `%t`：日志条目的时间，不带毫秒
- `%m`：日志条目的时间，带毫秒
- `%r` – 远程主机地址
- `%u@%d` – 用户名 @ 数据库名称
- `[%p]` – 进程 ID（如果有）
- `%l`：每个会话的日志行号
- `%e`：SQL 错误代码
- `%s`：进程开始时间戳
- `%v`：虚拟事务 ID
- `%x`：事务 ID
- `%c`：会话 ID
- `%q`：非会话终止器
- `%a`：应用程序名称

为您的 RDS for PostgreSQL 数据库实例开启查询日志记录

通过设置下表中列出的一些参数，您可以收集有关数据库活动的更多详细信息，包括查询、等待锁定的查询、检查点和许多其他详细信息。本主题重点介绍日志记录查询。

参数	默认值	描述
log_connections	–	记录每个成功的连接。
log_disconnections	–	记录每个会话的结束及其持续时间。
log_checkpoints	1	记录每个检查点。
log_lock_waits	–	记录长锁定等待次数。默认情况下，不设置此参数。
log_min_duration_sample	–	设置如超出则记录语句示例的最短执行时间（ms）。示例数量使用 `log_statement_sample_rate` 参数进行设置。

参数	默认值	描述
log_min_duration_statement	–	任何至少运行指定时间或更长时间的 SQL 语句都会被记录下来。默认情况下，不设置此参数。开启该参数可帮助查找未优化的查询。
log_statement	–	设置所记录的语句类型。默认情况下，未设置此参数，但您可以将其更改为 `all`、`ddl` 或 `mod`，以指定要记录的 SQL 语句的类型。如果您为此参数指定了与 none 不同的任何值，还应采取其他措施来防止日志文件中的密码泄露。有关更多信息，请参阅 降低使用查询日志记录时泄露密码的风险。
log_statement_sample_rate	–	超过在 `log_min_duration_sample` 中指定的时间的语句百分比，以介于 0.0 和 1.0 之间的浮点值表示。
log_statement_stats	–	向服务器日志写入累计性能统计数据。

使用日志记录查找执行缓慢的查询

您可以记录 SQL 语句和查询，以帮助查找执行缓慢的查询。您可以通过修改本节中概述的 `log_statement` 和 `log_min_duration` 参数的设置来开启此功能。在为 RDS for PostgreSQL 数据库实例开启查询日志记录之前，您应该了解日志中可能存在的密码泄露以及如何降低风险。有关更多信息，请参阅 降低使用查询日志记录时泄露密码的风险。

接下来，您可以了解有关 `log_statement` 和 `log_min_duration` 参数的参考信息。

log_statement

此参数指定应发送到日志的 SQL 语句的类型。默认值为 none。如果您将此参数更改为 `all`、`ddl` 或 `mod`，请务必采取建议的操作来降低在日志中泄露密码的风险。有关更多信息，请参阅 降低使用查询日志记录时泄露密码的风险。

全部

记录所有语句。建议将此设置用于调试目的。

ddl

记录所有数据定义语言（DDL）语句，例如 CREATE、ALTER、DROP 等。

mod

记录所有可修改数据的 DDL 语句和数据操作语言（DML）语句，例如 INSERT、UPDATE 和 DELETE。

none

不记录任何 SQL 语句。我们建议使用此设置，以避免在日志中泄露密码的风险。

log_min_duration_statement

任何至少运行指定时间或更长时间的 SQL 语句都会被记录下来。默认情况下，不设置此参数。开启该参数可帮助查找未优化的查询。

–1–2147483647

如超过即记录语句的运行时间的毫秒数（ms）。

设置查询日志记录

这些步骤假设您的 RDS for PostgreSQL 数据库实例使用自定义数据库参数组。

1. 将 `log_statement` 参数设置为 `all`。以下示例显示了使用此参数设置写入 `postgresql.log` 文件的信息。

```
2022-10-05 22:05:52 UTC:52.95.4.1(11335):postgres@labdb:[3639]:LOG: statement:
 SELECT feedback, s.sentiment,s.confidence
FROM support,aws_comprehend.detect_sentiment(feedback, 'en') s
ORDER BY s.confidence DESC;
2022-10-05 22:05:52 UTC:52.95.4.1(11335):postgres@labdb:[3639]:LOG: QUERY
 STATISTICS
2022-10-05 22:05:52 UTC:52.95.4.1(11335):postgres@labdb:[3639]:DETAIL: ! system
 usage stats:
! 0.017355 s user, 0.000000 s system, 0.168593 s elapsed
! [0.025146 s user, 0.000000 s system total]
! 36644 kB max resident size
! 0/8 [0/8] filesystem blocks in/out
! 0/733 [0/1364] page faults/reclaims, 0 [0] swaps
! 0 [0] signals rcvd, 0/0 [0/0] messages rcvd/sent
```

```
! 19/0 [27/0] voluntary/involuntary context switches
2022-10-05 22:05:52 UTC:52.95.4.1(11335):postgres@labdb:[3639]:STATEMENT: SELECT
 feedback, s.sentiment,s.confidence
FROM support,aws_comprehend.detect_sentiment(feedback, 'en') s
ORDER BY s.confidence DESC;
2022-10-05 22:05:56 UTC:52.95.4.1(11335):postgres@labdb:[3639]:ERROR: syntax error
 at or near "ORDER" at character 1
2022-10-05 22:05:56 UTC:52.95.4.1(11335):postgres@labdb:[3639]:STATEMENT: ORDER BY
 s.confidence DESC;
----------------------- END OF LOG ----------------------
```

2. 设置 `log_min_duration_statement` 参数。以下示例显示了参数设置为 `postgresql.log` 时写入 1 文件的信息。

   将记录超过在 `log_min_duration_statement` 参数中指定的持续时间的查询。下面是一个示例。您可以在 Amazon RDS 控制台中查看 RDS for PostgreSQL 数据库实例的日志文件。

```
2022-10-05 19:05:19 UTC:52.95.4.1(6461):postgres@labdb:[6144]:LOG: statement: DROP
 table comments;
2022-10-05 19:05:19 UTC:52.95.4.1(6461):postgres@labdb:[6144]:LOG: duration:
 167.754 ms
2022-10-05 19:08:07 UTC::@:[355]:LOG: checkpoint starting: time
2022-10-05 19:08:08 UTC::@:[355]:LOG: checkpoint complete: wrote 11 buffers
 (0.0%); 0 WAL file(s) added, 0 removed, 0 recycled; write=1.013 s, sync=0.006 s,
 total=1.033 s; sync files=8, longest=0.004 s, average=0.001 s; distance=131028 kB,
 estimate=131028 kB
----------------------- END OF LOG ----------------------
```

降低使用查询日志记录时泄露密码的风险

我们建议您将 `log_statement` 保持设置为 `none`，以避免泄露密码。如果您将 `log_statement` 设置为 `all`、`ddl` 或 `mod`，我们建议您采取以下一个或多个步骤。

- 对于客户端，加密敏感信息。有关更多信息，请参阅 PostgreSQL 文档中的加密选项。使用 `ENCRYPTED` 和 `UNENCRYPTED` 语句的 `CREATE`（和 `ALTER`）选项。有关更多信息，请参阅 PostgreSQL 文档中的 CREATE USER。
- 对于您的 RDS for PostgreSQL 数据库实例，请设置并使用 PostgreSQL Auditing（pgAudit）扩展。此扩展编辑发送到日志的 CREATE 和 ALTER 语句中的敏感信息。有关更多信息，请参阅 使用 pgAudit 记录数据库活动。
- 限制对 CloudWatch Logs 的访问。

- 使用更强的身份验证机制，如 IAM。

将 PostgreSQL 日志发布到 Amazon CloudWatch Logs

要在高持久性存储中存储 PostgreSQL 日志记录，您可以使用 Amazon CloudWatch Logs。利用 CloudWatch Logs，您还可以对日志数据进行实时分析，并使用 CloudWatch 查看指标和创建警报。例如，如果您将 `log_statement` 设置为 `ddl`，则可以将告警设置为只要执行 DDL 语句就发出提示。在创建 RDS for PostgreSQL 数据库实例的过程中，您可以选择让 PostgreSQL 日志上载到 CloudWatch Logs。如果您当时选择不上载日志，可以稍后修改您的实例，以便从那时起开始上载日志。换句话说，不会上载现有日志。只有在修改后的 RDS for PostgreSQL 数据库实例上创建了新日志时，才上载这些新日志。

所有目前可用的 RDS for PostgreSQL 版本都支持将日志文件发布到 CloudWatch Logs。有关更多信息，请参阅《Amazon RDS for PostgreSQL 版本注释》中的 [Amazon RDS for PostgreSQL 更新]。

要使用 CloudWatch Logs，请为 PostgreSQL 数据库实例配置 RDS 以将日志数据发布到日志组。

您可以将以下日志类型发布到 CloudWatch Logs for RDS for PostgreSQL：

- PostgreSQL 日志
- 升级日志
- IAM 数据库身份验证错误日志

在完成配置后，Amazon RDS 将日志事件发布到 CloudWatch 日志组内的日志流。例如，PostgreSQL 日志数据存储在日志组 `/aws/rds/instance/`*`my_instance`*`/postgresql` 中。要查看您的日志，请在 https://console.aws.amazon.com/cloudwatch/ 处打开 CloudWatch 控制台。

控制台

使用控制台将 PostgreSQL 日志发布到 CloudWatch Logs

1. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。
3. 选择要修改的数据库实例，然后选择 Modify (修改)。
4. 在 Log exports (日志导出) 部分中，选择要开始发布到 CloudWatch Logs 的日志。

   Log exports (日志导出) 部分仅适用于支持发布到 CloudWatch Logs 的 PostgreSQL 版本。
5. 选择继续，然后选择摘要页面上的修改数据库实例。

Amazon CLI

您可以使用 Amazon CLI 来发布 PostgreSQL 日志。您可以调用带以下参数的 `modify-db-instance` 命令。

- `--db-instance-identifier`
- `--cloudwatch-logs-export-configuration`

> **Note**
>
> 对 `--cloudwatch-logs-export-configuration` 选项进行的更改始终立即应用于数据库实例。因此，`--apply-immediately` 和 `--no-apply-immediately` 选项没有影响。

您还可以通过调用以下 CLI 命令来发布 PostgreSQL 日志：

- `create-db-instance`
- `restore-db-instance-from-db-snapshot`
- `restore-db-instance-to-point-in-time`

通过以下选项运行上述 CLI 命令之一：

- `--db-instance-identifier`
- `--enable-cloudwatch-logs-exports`
- `--db-instance-class`
- `--engine`

根据您运行的 CLI 命令，可能需要其他选项。

Example 修改实例以将日志发布到 CloudWatch Logs

以下示例修改现有的 PostgreSQL 数据库实例以将日志文件发布到 CloudWatch Logs。`--cloudwatch-logs-export-configuration` 值是 JSON 对象。此对象的键是 `EnableLogTypes`，其值是具有 `postgresql` 和 `upgrade` 的任意组合的字符串的数组。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier mydbinstance \
    --cloudwatch-logs-export-configuration '{"EnableLogTypes":["postgresql",
 "upgrade"]}'
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier mydbinstance ^
    --cloudwatch-logs-export-configuration '{"EnableLogTypes":
["postgresql","upgrade"]}'
```

Example 创建实例以将日志发布到 CloudWatch Logs

以下示例将创建一个 PostgreSQL 数据库实例并将日志文件发布到 CloudWatch Logs。`--enable-cloudwatch-logs-exports` 值是 JSON 字符串数组。这些字符串可以是 `postgresql` 和 `upgrade` 的任意组合。

对于 Linux、macOS 或 Unix：

```
aws rds create-db-instance \
    --db-instance-identifier mydbinstance \
    --enable-cloudwatch-logs-exports '["postgresql","upgrade"]' \
    --db-instance-class db.m4.large \
    --engine postgres
```

对于 Windows：

```
aws rds create-db-instance ^
    --db-instance-identifier mydbinstance ^
    --enable-cloudwatch-logs-exports '["postgresql","upgrade"]' ^
    --db-instance-class db.m4.large ^
    --engine postgres
```

RDS API

您可以使用 RDS API 来发布 PostgreSQL 日志。您可以使用以下参数调用 `ModifyDBInstance` 操作：

- `DBInstanceIdentifier`
- `CloudwatchLogsExportConfiguration`

> **Note**
>
> 对 `CloudwatchLogsExportConfiguration` 参数进行的更改始终立即应用于数据库实例。因此，`ApplyImmediately` 参数没有影响。

您还可以通过调用以下 RDS API 操作来发布 PostgreSQL 日志：

- `CreateDBInstance`
- `RestoreDBInstanceFromDBSnapshot`
- `RestoreDBInstanceToPointInTime`

使用以下参数运行这些 RDS API 操作之一：

- `DBInstanceIdentifier`
- `EnableCloudwatchLogsExports`
- `Engine`
- `DBInstanceClass`

根据您运行的操作，可能需要其他参数。

监控 Amazon CloudTrail 中的 Amazon RDS API 调用

Amazon CloudTrail 是一项可帮助您审计 Amazon 账户的 Amazon 服务。在您创建 Amazon CloudTrail 账户时，对您的账户开启 Amazon。有关 CloudTrail 的更多信息，请参阅《Amazon CloudTrail 用户指南》https://docs.amazonaws.cn/awscloudtrail/latest/userguide/。

主题

CloudTrail 与 Amazon RDS 集成

所有 Amazon RDS 操作均由 CloudTrail 记录。CloudTrail 提供了用户、角色或 Amazon 服务在 Amazon RDS 中所执行操作的记录。

CloudTrail 事件

CloudTrail 将 Amazon RDS 的 API 调用作为事件捕获。一个事件表示来自任何源的一个请求，包括有关所请求的操作、操作的日期和时间、请求参数等方面的信息。事件包含来自 Amazon RDS 控制台的调用和对 Amazon RDS API 操作的代码调用。

Amazon RDS 活动记录在 Event history（事件历史记录）中的 CloudTrail 事件中。您可以使用 CloudTrail 控制台查看Amazon区域中过去 90 天内记录的 API 活动和事件。有关更多信息，请参阅使用 CloudTrail 事件历史记录查看事件。

CloudTrail 跟踪

要持续记录 Amazon 账户中的事件（包括 Amazon RDS 的事件），请创建跟踪记录。跟踪是一种配置，可用于将事件传送到指定的 Amazon S3 存储桶。CloudTrail 通常会在账户活动发生后的 15 分钟内传送日志文件。

> **Note**
>
> 如果您不配置跟踪，则仍可在 CloudTrail 控制台中的事件历史记录中查看最新事件。

您可以为 Amazon 账户创建两种类型的跟踪：应用于所有区域的跟踪，或应用于一个区域的跟踪。预设情况下，在控制台中创建跟踪时，此跟踪应用于所有 区域。

此外，您可以配置其他 Amazon 服务，进一步分析在 CloudTrail 日志中收集的事件数据并采取行动。有关更多信息，请参阅：

- 创建跟踪概览
- CloudTrail 支持的服务和集成
- 为 CloudTrail 配置 Amazon SNS 通知
- 从多个区域接收 CloudTrail 日志文件和从多个账户接收 CloudTrail 日志文件

Amazon RDS 日志文件条目

CloudTrail 日志文件包含一个或多个日志条目。CloudTrail 日志文件不是公用 API 调用的有序堆栈跟踪，因此它们不会按任何特定顺序显示。

下面的示例显示了一个 CloudTrail 日志条目，该条目说明了 `CreateDBInstance` 操作。


```
{
    "eventVersion": "1.04",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AKIAIOSFODNN7EXAMPLE",
        "arn": "arn:aws:iam::123456789012:user/johndoe",
        "accountId": "123456789012",
        "accessKeyId": "AKIAI44QH8DHBEXAMPLE",
        "userName": "johndoe"
    },
    "eventTime": "2018-07-30T22:14:06Z",
    "eventSource": "rds.amazonaws.com",
    "eventName": "CreateDBInstance",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "aws-cli/1.15.42 Python/3.6.1 Darwin/17.7.0 botocore/1.10.42",
    "requestParameters": {
        "enableCloudwatchLogsExports": [
            "audit",
            "error",
            "general",
            "slowquery"
        ],
        "dBInstanceIdentifier": "test-instance",
        "engine": "mysql",
```

```
        "masterUsername": "myawsuser",
        "allocatedStorage": 20,
        "dBInstanceClass": "db.m1.small",
        "masterUserPassword": "****"
    },
    "responseElements": {
        "dBInstanceArn": "arn:aws:rds:us-east-1:123456789012:db:test-instance",
        "storageEncrypted": false,
        "preferredBackupWindow": "10:27-10:57",
        "preferredMaintenanceWindow": "sat:05:47-sat:06:17",
        "backupRetentionPeriod": 1,
        "allocatedStorage": 20,
        "storageType": "standard",
        "engineVersion": "8.0.28",
        "dbInstancePort": 0,
        "optionGroupMemberships": [
            {
                "status": "in-sync",
                "optionGroupName": "default:mysql-8-0"
            }
        ],
        "dBParameterGroups": [
            {
                "dBParameterGroupName": "default.mysql8.0",
                "parameterApplyStatus": "in-sync"
            }
        ],
        "monitoringInterval": 0,
        "dBInstanceClass": "db.m1.small",
        "readReplicaDBInstanceIdentifiers": [],
        "dBSubnetGroup": {
            "dBSubnetGroupName": "default",
            "dBSubnetGroupDescription": "default",
            "subnets": [
                {
                    "subnetAvailabilityZone": {"name": "us-east-1b"},
                    "subnetIdentifier": "subnet-cbfff283",
                    "subnetStatus": "Active"
                },
                {
                    "subnetAvailabilityZone": {"name": "us-east-1e"},
                    "subnetIdentifier": "subnet-d7c825e8",
                    "subnetStatus": "Active"
                },
```

```
                {
                    "subnetAvailabilityZone": {"name": "us-east-1f"},
                    "subnetIdentifier": "subnet-6746046b",
                    "subnetStatus": "Active"
                },
                {
                    "subnetAvailabilityZone": {"name": "us-east-1c"},
                    "subnetIdentifier": "subnet-bac383e0",
                    "subnetStatus": "Active"
                },
                {
                    "subnetAvailabilityZone": {"name": "us-east-1d"},
                    "subnetIdentifier": "subnet-42599426",
                    "subnetStatus": "Active"
                },
                {
                    "subnetAvailabilityZone": {"name": "us-east-1a"},
                    "subnetIdentifier": "subnet-da327bf6",
                    "subnetStatus": "Active"
                }
            ],
            "vpcId": "vpc-136a4c6a",
            "subnetGroupStatus": "Complete"
        },
        "masterUsername": "myawsuser",
        "multiAZ": false,
        "autoMinorVersionUpgrade": true,
        "engine": "mysql",
        "cACertificateIdentifier": "rds-ca-2015",
        "dbiResourceId": "db-ETDZIIXHEWY5N7GXVC4SH7H5IA",
        "dBSecurityGroups": [],
        "pendingModifiedValues": {
            "masterUserPassword": "****",
            "pendingCloudwatchLogsExports": {
                "logTypesToEnable": [
                    "audit",
                    "error",
                    "general",
                    "slowquery"
                ]
            }
        },
        "dBInstanceStatus": "creating",
        "publiclyAccessible": true,
```

```
        "domainMemberships": [],
        "copyTagsToSnapshot": false,
        "dBInstanceIdentifier": "test-instance",
        "licenseModel": "general-public-license",
        "iAMDatabaseAuthenticationEnabled": false,
        "performanceInsightsEnabled": false,
        "vpcSecurityGroups": [
            {
                "status": "active",
                "vpcSecurityGroupId": "sg-f839b688"
            }
        ]
    },
    "requestID": "daf2e3f5-96a3-4df7-a026-863f96db793e",
    "eventID": "797163d3-5726-441d-80a7-6eeb7464acd4",
    "eventType": "AwsApiCall",
    "recipientAccountId": "123456789012"
}
```

如前面示例中的 `userIdentity` 元素所示，每个事件或日志条目都包含有关生成请求的人员的信息。身份信息可帮助您确定以下内容：

- 请求是使用根用户凭证还是 IAM 用户凭证发出的。
- 请求是使用角色还是联合用户的临时安全凭证发出的。
- 请求是否由其他 Amazon 服务发出。

有关 `userIdentity` 的更多信息，请参阅 CloudTrail userIdentity 元素。有关 `CreateDBInstance` 和其他 Amazon RDS 操作的更多信息，请参阅 Amazon RDS API 参考。

使用数据库活动流监控 Amazon RDS

通过使用数据库活动流，您可以监控近乎实时的数据库活动流。

主题

数据库活动流概览

作为 Amazon RDS 数据库管理员，您需要保障数据库的安全，并满足合规性和法规要求。一种策略是集成数据库活动流与监控工具。通过这种方式，您可以在数据库中监控审计活动并相应设置警报。

安全威胁既可以来自外部，也可以来自内部。要防范内部威胁，您可以通过配置数据库活动流功能控制管理员对数据流的访问。Amazon RDS 数据管理员无权收集、传输、存储和处理流。

目录

- 区域和版本可用性
- 数据库活动流支持的数据库实例类

数据库活动流的工作原理

Amazon RDS 会近乎实时地将活动推送到 Amazon Kinesis 数据流。系统将自动创建 Kinesis 流。在 Kinesis 中，您可以配置 Amazon 服务（如 Amazon Data Firehose）和 Amazon Lambda 来使用 Kinesis 流并存储数据。

> **Important**
>
> 使用 Amazon RDS 中的 数据库活动流功能是免费的，但 Amazon Kinesis 会针对数据流收费。有关更多信息，请参阅 Amazon Kinesis Data Streams 定价。

您可以为合规性管理配置应用程序，以使用数据库活动流。这些应用程序可以使用流生成警报，并审计数据库上的活动。

Amazon RDS 在多可用区部署中支持数据库活动流。在这种情况下，数据库活动流会审计主实例和备用实例。

Oracle 数据库和 Microsoft SQL Server 数据库中的审计

审计是监控和记录已配置的数据库操作。Amazon RDS 在默认情况下不捕获数据库活动。您可自行在数据库中创建和管理审核策略。

主题

- Oracle 数据库中的统一审计
- Microsoft SQL Server 中的审计
- Oracle 数据库和 SQL Server 的非原生审计字段
- 数据库参数组覆盖

Oracle 数据库中的统一审计

在 Oracle 数据库中，统一审核策略是一组已命名的审核设置，可用于审核某个方面的用户行为。策略可以是简单的审计单个用户的活动。您还可以创建使用条件的复杂审核策略。

Oracle 数据库将审计记录（包括 SYS 审计记录）写入统一审计跟踪记录。例如，如果在 INSERT 语句期间发生错误，标准审计会指示错误编号和所运行的 SQL。审计跟踪记录位于 AUDSYS 架构的只读表中。要访问这些记录，请查询 UNIFIED_AUDIT_TRAIL 数据词典视图。

通常，您可以按如下方式配置数据库活动流：

1. 使用 CREATE AUDIT POLICY 命令创建 Oracle 数据库审核策略。

   Oracle 数据库会生成审计记录。
2. 使用 AUDIT POLICY 命令激活审核策略。
3. 配置数据库活动流。

   仅捕获与 Oracle 数据库审核策略匹配的活动，并将其发送到 Amazon Kinesis 数据流。启用数据库活动流后，Oracle 数据库管理员无法更改审核策略或删除审核日志。

要了解有关统一审核策略的更多信息，请参阅 Oracle Database 安全指南中的关于使用统一审核策略和审核审核活动。

Microsoft SQL Server 中的审计

数据库活动流使用 SQLAudit 功能审计 SQL Server 数据库。

RDS for SQL Server 实例包含以下各项功能：

- 服务器审计 – SQL Server 审计收集服务器或数据库级操作的单个实例以及一组要监视的操作。服务器级审计 RDS_DAS_AUDIT 和 RDS_DAS_AUDIT_CHANGES 由 RDS 管理。
- 服务器审计规范 - 服务器审计规范记录服务器级别的事件。您可以修改 RDS_DAS_SERVER_AUDIT_SPEC 规范。该规范链接到服务器审计 RDS_DAS_AUDIT。RDS_DAS_CHANGES_AUDIT_SPEC 规范由 RDS 管理。
- 数据库审计规范 - 数据库审计规范记录数据库级别的事件。您可以创建数据库审计规范 RDS_DAS_DB_<name> 并将其链接到 RDS_DAS_AUDIT 服务器审计。

您可以使用控制台或 CLI 配置数据库活动流。通常，您可以按如下方式配置数据库活动流：

1. （可选）使用 CREATE DATABASE AUDIT SPECIFICATION 命令创建数据库审计规范并将其链接到 RDS_DAS_AUDIT 服务器审计。
2. （可选）使用 ALTER SERVER AUDIT SPECIFICATION 命令修改服务器审计规范并定义策略。
3. 激活数据库和服务器审核策略。例如：

```
ALTER DATABASE AUDIT SPECIFICATION [<Your database specification>] WITH
(STATE=ON)
```

```
ALTER SERVER AUDIT SPECIFICATION [RDS_DAS_SERVER_AUDIT_SPEC] WITH
(STATE=ON)
```

4. 配置数据库活动流。

   仅捕获与服务器和数据库审核策略匹配的活动，并将其发送到 Amazon Kinesis 数据流。启用数据库活动流且锁定策略后，数据库管理员无法更改审核策略或删除审核日志。

   > **Important**
   >
   > 如果启用了特定数据库的数据库审计规范并且策略处于锁定状态，则无法删除该数据库。

有关 SQL Server 审计的更多信息，请参阅《Microsoft SQL Server 文档》中的 SQL Server 审计组件。

Oracle 数据库和 SQL Server 的非原生审计字段

启动数据库活动流时，每个数据库事件都会生成相应的活动流事件。例如，数据库用户可能会运行 `SELECT` 和 `INSERT` 语句。数据库会审计这些事件并将结果发送到 Amazon Kinesis 数据流。

活动流中的事件表示为 JSON 对象。JSON 对象包含一个 `DatabaseActivityMonitoringRecord`，其中包含一个 `databaseActivityEventList` 数组。数组中的预定义字段包括 `class`、`clientApplication` 和 `command`。

默认情况下，活动流不包括原生引擎审计字段。您可以配置 Amazon RDS for Oracle 和 SQL Server，使其在 `engineNativeAuditFields` JSON 对象中包含这些额外的字段。

在 Oracle 数据库中，统一审计跟踪中的大多数事件都映射到 RDS 数据活动流中的字段。例如，统一审计中的 `UNIFIED_AUDIT_TRAIL.SQL_TEXT` 字段映射到数据库活动流中的 `commandText` 字段。但是，Oracle 数据库审计字段（如 `OS_USERNAME`）不映射到数据库活动流中的预定义字段。

在 SQL Server 中，SQLAudit 记录的大多数事件字段都映射到 RDS 数据库活动流中的字段。例如，来自审计中 `sys.fn_get_audit_file` 的 `code` 字段映射到数据库活动流中的 `commandText` 字段。但是，SQL Server 数据库审计字段（如 `permission_bitmask`）不映射到数据库活动流中的预定义字段。

有关 databaseActivityEventList 的更多信息，请参阅数据库活动流的 databaseActivityEventList JSON 数组。

数据库参数组覆盖

通常，您可以通过附加参数组来打开 RDS for Oracle 中的统一审计。但是，数据库活动流需要额外的配置。为了改善您的客户体验，Amazon RDS 执行以下操作：

- 如果激活活动流，RDS for Oracle 将忽略参数组中的审计参数。
- 如果停用活动流，RDS for Oracle 将停止忽略审计参数。

SQL Server 的数据库活动流与您在 SQL 审计选项中设置的任何参数均无关。

数据库活动流的异步模式

Amazon RDS 中的活动流始终是异步的。当数据库会话生成活动流事件时，会话将立即返回到正常活动。在后台，Amazon RDS 使活动流事件成为持久记录。

如果后台任务出错，则 Amazon RDS 将生成事件。此事件指示活动流事件记录可能已丢失的任何时间段的开始和结束时间。异步模式可提高数据库性能，而不是活动流的准确性。

数据库活动流的要求和限制

在 RDS 中，数据库活动流具有以下要求和限制：

- 数据库活动流需要使用 Amazon Kinesis。
- 数据库活动流需要使用 Amazon Key Management Service (Amazon KMS)，因为这些活动流始终是加密的。
- 对 Amazon Kinesis 数据流应用额外加密与数据库活动流不兼容，因为数据库活动流已使用 Amazon KMS 密钥进行了加密。
- 您可自行创建和管理审核策略。与 Amazon Aurora 不同，RDS for Oracle 默认不捕获数据库活动。
- 您可自行创建和管理审核策略或规范。与 Amazon Aurora 不同，Amazon RDS 在默认情况下不捕获数据库活动。
- 在多可用区部署中，仅在主数据库实例上启动数据库活动流。该活动流会自动审计主数据库实例和备用数据库实例。失效转移期间无需执行其他步骤。
- 重命名数据库实例不会创建新的 Kinesis 流。
- RDS for Oracle 不支持 CDB。

- 不支持只读副本。

区域和版本可用性

功能可用性和支持因每个数据库引擎的特定版本以及 Amazon Web Services 区域而异。有关适用于数据库活动流的版本和区域可用性的更多信息，请参阅支持 Amazon RDS 中数据库活动流的区域和数据库引擎。

数据库活动流支持的数据库实例类

对于 RDS for Oracle，您可以将数据库活动流与以下数据库实例类一起使用：

- db.m4.*large
- db.m5.*large
- db.m5d.*large
- db.m6i.*large
- db.r4.*large
- db.r5.*large
- db.r5.*large.tpc*.mem*x
- db.r5b.*large
- db.r5b.*large.tpc*.mem*x
- db.r5d.*large
- db.r6i.*large
- db.r6i.*large.tpc*.mem*x
- db.x2idn.*large
- db.x2iedn.*large
- db.x2iezn.*large
- db.z1d.*large

对于 RDS for SQL Server，您可以将数据库活动流与以下数据库实例类一起使用：

- db.m4.*large
- db.m5.*large

- db.m5d.*large
- db.m6i.*large
- db.r4.*large
- db.r5.*large
- db.r5b.*large
- db.r5d.*large
- db.r6i.*large
- db.x1e.*large
- db.x2iedn.*large
- db.z1d.*large

有关实例类类型的更多信息，请参阅 数据库实例类。

为 Oracle 数据库配置统一审计

配置统一审计以用于数据库活动流时，可能会出现以下情况：

- 未为 Oracle 数据库配置统一审计。

  在这种情况下，使用 `CREATE AUDIT POLICY` 命令创建新策略，然后使用 `AUDIT POLICY` 命令激活这些策略。以下示例创建并激活策略来监控具有特定权限和角色的用户。

  ```
  CREATE AUDIT POLICY table_pol
  PRIVILEGES CREATE ANY TABLE, DROP ANY TABLE
  ROLES emp_admin, sales_admin;

  AUDIT POLICY table_pol;
  ```

  有关完整说明，请参阅 Oracle 数据库文档中配置审计策略。
- 已为 Oracle 数据库配置了统一审计。

  激活数据库活动流时，RDS for Oracle 会自动清除现有审计数据。还会撤消审计跟踪权限。RDS for Oracle 无法再执行以下操作：
  - 清除统一审计跟踪记录。
  - 添加、删除或修改统一审计策略。
  - 更新上次归档的时间戳。

> **Important**
>
> 强烈建议您在激活数据库活动流之前备份审计数据。

有关 UNIFIED_AUDIT_TRAIL 视图的描述，请参阅 UNIFIED_AUDIT_TRAIL。如果您有 Oracle Support 账户，请参阅如何清除统一审计跟踪。

为 Amazon RDS for Microsoft SQL Server 配置审计策略

SQL Server 数据库实例具有服务器审计 RDS_DAS_AUDIT（由 Amazon RDS 管理）。您可以在服务器审计规范 RDS_DAS_SERVER_AUDIT_SPEC 中定义策略以记录服务器事件。您可以创建数据库审计规范，例如 RDS_DAS_DB_<name>，并定义记录数据库事件的策略。有关服务器和数据库级审计操作组的列表，请参阅《Microsoft SQL Server 文档》中的 SQL Server 审计操作组和操作。

原定设置服务器策略仅监控失败的登录，以及对数据库活动流的任何数据库或服务器审计规范进行的更改。

审计和审计规范的限制包括以下内容：

- 当数据库活动流处于锁定状态时，您无法修改服务器或数据库审计规范。
- 您无法修改服务器审计 RDS_DAS_AUDIT 规范。
- 您无法修改 SQL Server 审计 RDS_DAS_CHANGES 或其相关的服务器审计规范 RDS_DAS_CHANGES_AUDIT_SPEC。
- 创建数据库审计规范时，必须使用格式 RDS_DAS_DB_<name>，例如 RDS_DAS_DB_databaseActions。

> **Important**
>
> 对于较小的实例类，我们建议您不要审计所有数据，而只审计所需的数据。这有助于减少数据库活动流对这些实例类的性能影响。

以下示例代码修改了服务器审计规范 RDS_DAS_SERVER_AUDIT_SPEC，并对所有注销和成功的登录操作进行审计：

```
ALTER SERVER AUDIT SPECIFICATION [RDS_DAS_SERVER_AUDIT_SPEC]
      WITH (STATE=OFF);
ALTER SERVER AUDIT SPECIFICATION [RDS_DAS_SERVER_AUDIT_SPEC]
      ADD (LOGOUT_GROUP),
      ADD (SUCCESSFUL_LOGIN_GROUP)
      WITH (STATE = ON );
```

以下示例代码创建数据库审计规范 RDS_DAS_DB_database_spec，并将其附加到服务器审计 RDS_DAS_AUDIT：

```
USE testDB;
CREATE DATABASE AUDIT SPECIFICATION [RDS_DAS_DB_database_spec]
     FOR SERVER AUDIT [RDS_DAS_AUDIT]
     ADD ( INSERT, UPDATE, DELETE
          ON testTable BY testUser )
     WITH (STATE = ON);
```

配置审计规范后，确保将规范 RDS_DAS_SERVER_AUDIT_SPEC 和 RDS_DAS_DB_<name> 设置为 ON 状态。现在，它们可以将审计数据发送到您的数据库活动流。

启动数据库活动流

在启动数据库实例的活动流时，在审计策略中配置的每个数据库活动事件都会生成一个活动流事件。SQL 命令（例如 CONNECT 和 SELECT）可生成访问事件。SQL 命令（例如 CREATE 和 INSERT）可生成更改事件。

> **Important**
>
> 打开 Oracle 数据库实例的活动流会清除现有的审计数据。还会撤消审计跟踪权限。启用数据流后，Oracle RDS 将无法再执行以下操作：
>
> - 清除统一审计跟踪记录。
> - 添加、删除或修改统一审计策略。
> - 更新上次存档的时间戳。

Console

要启动数据库活动流

1. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。
3. 选择要对其启动活动流的 Amazon RDS 数据库实例。在多可用区部署中，仅在主实例上启动活动流。活动流会审计主实例和备用实例。
4. 对于 Actions (操作)，选择 Start activity stream (启动活动流)。

   启动数据库活动流：##窗口出现，其中##是您的 RDS 实例。
5. 输入以下设置：

   - 对于 Amazon KMS key，从 Amazon KMS keys 列表中选择一个密钥。

     Amazon RDS 使用 KMS 密钥加密密钥，从而加密数据库活动。请选择原定设置密钥以外的 KMS 密钥。有关加密密钥和 Amazon KMS 的更多信息，请参阅 Amazon Key Management Service 开发人员指南中的什么是 Amazon Key Management Service？。
   - 对于数据库活动事件，选择启用引擎原生审计字段以包含引擎特定的审计字段。
   - 选择 Immediately (立即)。

     当您选择 Immediately (立即) 时，RDS 实例会立即重新启动。如果选择 During the next maintenance window (在下一维护时段内)，RDS 实例不会立即重新启动。在这种情况下，数据库活动流不会启动，直到下一个维护时段。
6. 选择 Start database activity stream（启动数据库活动流）。

   数据库的状态显示活动流正在启动。

   > **Note**
   >
   > 如果您收到错误 `You can't start a database activity stream in this configuration`，请检查 数据库活动流支持的数据库实例类，以了解您的RDS 实例是否正在使用受支持的实例类。

Amazon CLI

要为数据库实例启动数据库活动流，请使用 Amazon CLI 命令 配置数据库集群数据库。

- `--resource-arn` *`arn`* – 指定数据库实例的 Amazon Resource Name (ARN)。
- `--kms-key-id` *`key`* – 指定用于加密数据库活动流中的消息的 KMS 密钥标识符。Amazon KMS 密钥标识符是密钥 ARN、密钥 ID、别名 ARN 或者 Amazon KMS key 的别名。
- `--engine-native-audit-fields-included` – 包括数据流中引擎特定的审计字段。要排除这些字段，请指定 `--no-engine-native-audit-fields-included`（默认值）。

以下示例以异步模式启动数据库实例的数据库活动流。

对于 Linux、macOS 或 Unix：

```
aws rds start-activity-stream \
    --mode async \
    --kms-key-id my-kms-key-arn \
    --resource-arn my-instance-arn \
    --engine-native-audit-fields-included \
    --apply-immediately
```

对于 Windows：

```
aws rds start-activity-stream ^
    --mode async ^
    --kms-key-id my-kms-key-arn ^
    --resource-arn my-instance-arn ^
    --engine-native-audit-fields-included ^
    --apply-immediately
```

Amazon RDS API

要为数据库实例启动数据库活动流，请使用 StartActivityStream 操作配置实例。

使用以下参数调用操作：

- `Region`
- `KmsKeyId`
- `ResourceArn`
- `Mode`
- `EngineNativeAuditFieldsIncluded`

修改 Amazon RDS 的数据库活动流

启动活动流时，您可能需要自定义 Amazon RDS 审计策略。如果您不想通过停止活动流而损失时间和数据，可以将审计策略状态 更改为以下设置之一：

Locked (default) [已锁定（原定设置）]

数据库中的审计策略是只读的。

Unlocked（已解锁）

数据库中的审计策略为读/写状态。

基本步骤如下所示：

1. 将审计策略状态修改为已解锁。
2. 自定义您的审计策略。
3. 将审计策略状态修改为已锁定。

控制台

修改活动流的审计策略状态

1. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases（数据库）。
3. 对于 Actions（操作），选择 Modify database activity stream（修改数据库活动流）。

   Modify database activity stream: ***name***（修改数据库活动流：name）窗口，其中 *name* 是您的 RDS 实例。
4. 请选择以下任一选项：

   Locked（已锁定）

   当您锁定审计策略时，它会变为只读状态。除非解锁策略或停止活动流，否则您无法编辑审计策略。

   Unlocked（已解锁）

   当您解锁审计策略时，它变为读/写状态。您可以在启动活动流时编辑您的审计策略。
5. 选择 Modify DB activity stream（修改数据库活动流）。

Amazon RDS 数据库的状态显示正在配置活动流。

6. （可选）选择数据库实例链接。然后，选择配置选项卡。

   Audit policy status（审计策略状态）字段显示下列值之一：

   - Locked（已锁定）
   - Unlocked（已解锁）
   - Locking policy（锁定策略）
   - Unlocking policy（解锁策略）

Amazon CLI

要修改数据库实例的活动流状态，请使用 Amazon CLI 命令 modify-activity-stream。

选项	必填？	Description
`--resource-arn` *`my-instance-ARN`*	是	RDS 数据库实例的 Amazon 资源名称（ARN）。
`--audit-policy-state`	否	您的实例上数据库活动流的审计策略的新状态：`locked` 或 `unlocked`。

以下示例为在 *`my-instance-ARN`* 上启动的活动流解锁审计策略。

对于 Linux、macOS 或 Unix：

```
aws rds modify-activity-stream \
    --resource-arn my-instance-ARN \
    --audit-policy-state unlocked
```

对于 Windows：

```
aws rds modify-activity-stream ^
    --resource-arn my-instance-ARN ^
    --audit-policy-state unlocked
```

以下示例描述了实例 *`my-instance`*。部分示例输出显示审计策略已解锁。

```
aws rds describe-db-instances --db-instance-identifier my-instance

{
    "DBInstances": [
        {
            ...
            "Engine": "oracle-ee",
            ...
            "ActivityStreamStatus": "started",
            "ActivityStreamKmsKeyId": "ab12345e-1111-2bc3-12a3-ab1cd12345e",
            "ActivityStreamKinesisStreamName": "aws-rds-das-db-
AB1CDEFG23GHIJK4LMNOPQRST",
            "ActivityStreamMode": "async",
            "ActivityStreamEngineNativeAuditFieldsIncluded": true,
            "ActivityStreamPolicyStatus": "unlocked",
            ...
        }
    ]
}
```

RDS API

要修改数据库活动流的策略状态，请使用 ModifyActivityStream 操作。

使用以下参数调用操作：

- `AuditPolicyState`
- `ResourceArn`

获取数据库活动流的状态

您可以使用控制台或 Amazon CLI 获取 Amazon RDS 数据库实例的活动流的状态。

控制台

要获取数据库活动流的状态，请执行以下操作

1. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)，然后选择数据库实例链接。
3. 选择 Configuration (配置) 选项卡，并选择 Database activity stream (数据库活动流) 以查看状态。

Amazon CLI

您可以获取数据库实例的活动流配置，作为对 describe-db-instances CLI 请求的响应。

以下示例描述 *my-instance*。

```
aws rds --region my-region describe-db-instances --db-instance-identifier my-db
```

下面的示例介绍一个 JSON 响应。显示了以下字段：

- `ActivityStreamKinesisStreamName`
- `ActivityStreamKmsKeyId`
- `ActivityStreamStatus`
- `ActivityStreamMode`
- `ActivityStreamPolicyStatus`

```
{
    "DBInstances": [
        {
            ...
            "Engine": "oracle-ee",
            ...
            "ActivityStreamStatus": "starting",
            "ActivityStreamKmsKeyId": "ab12345e-1111-2bc3-12a3-ab1cd12345e",
            "ActivityStreamKinesisStreamName": "aws-rds-das-db-
AB1CDEFG23GHIJK4LMNOPQRST",
            "ActivityStreamMode": "async",
            "ActivityStreamEngineNativeAuditFieldsIncluded": true,
            "ActivityStreamPolicyStatus": locked",
            ...
        }
    ]
}
```

RDS API

您可以获取数据库的活动流配置，作为对 DescribeDBInstances 操作的响应。

停止数据库活动流

您可以使用控制台或 Amazon CLI 停止活动流。

如果您删除 Amazon RDS 数据库实例，活动流将停止，并且会自动删除底层 Amazon Kinesis 流。

控制台

关闭活动流

1. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。
3. 选择要为其停止数据库活动流的数据库。
4. 对于 Actions (操作)，选择 Stop activity stream (停止活动流)。此时将显示 Database Activity Stream (数据库活动流) 窗口。

    a. 选择 Immediately (立即)。

       当您选择 Immediately (立即) 时，RDS 实例会立即重新启动。如果选择 During the next maintenance window (在下一维护时段内)，RDS 实例不会立即重新启动。在这种情况下，数据库活动流直到下一个维护时段才会停止。

    b. 选择 Continue (继续)。

Amazon CLI

要为数据库停止数据库活动流，请使用 Amazon CLI 命令 stop-activity-stream 配置数据库实例。使用 `--region` 参数标识数据库实例的Amazon区域。`--apply-immediately` 参数是可选的。

对于 Linux、macOS 或 Unix：

```
aws rds --region MY_REGION \
    stop-activity-stream \
    --resource-arn MY_DB_ARN \
    --apply-immediately
```

对于 Windows：

```
aws rds --region MY_REGION ^
    stop-activity-stream ^
```

```
    --resource-arn MY_DB_ARN ^
    --apply-immediately
```

RDS API

要为数据库停止数据库活动流，请使用 StopActivityStream 操作配置数据库实例。使用 `Region` 参数标识数据库实例的Amazon区域。`ApplyImmediately` 参数是可选的。

监控数据库活动流

数据库活动流监控并报告活动。收集活动流并将其传输到 Amazon Kinesis。从 Kinesis 中，您可以监控活动流，或者其他服务和应用程序可以使用活动流进行进一步分析。您可以使用 Amazon CLI 命令 `describe-db-instances` 或 RDS API `DescribeDBInstances` 操作查找底层 Kinesis 流名称。

Amazon RDS 为您管理 Kinesis 流，如下所示：

- Amazon RDS 自动创建具有 24 小时保留期的 Kinesis 流。
- 如有必要，Amazon RDS 会扩展 Kinesis 流。
- 如果停止了数据库活动流或删除了数据库实例，Amazon RDS 将删除 Kinesis 流。

监控以下类别的活动并将其放入活动流审核日志中：

- SQL 命令 – 将审计所有 SQL 命令，以及预编译语句、内置函数和采用 PL/SQL 的函数。将审核对存储过程的调用。还将审核在存储过程或函数中发出的任何 SQL 语句。
- 其他数据库信息 – 受监控的活动包括完整的 SQL 语句、来自 DML 命令的受影响行的行数、访问的对象以及唯一的数据库名称。数据库活动流还监控绑定变量和存储过程参数。

  > **Important**
  >
  > 每个语句的完整 SQL 文本在活动流审核日志中可见，包括任何敏感数据。但是，如果 Oracle 可以从上下文中确定数据库用户密码，则会对该密码进行修订，如下面的 SQL 语句所示。
  >
  > ```
  > ALTER ROLE role-name WITH password
  > ```

- 连接信息 – 受监控的活动包括会话和网络信息、服务器进程 ID 和退出代码。

如果在监控数据库实例时活动流发生故障，则会使用 RDS 事件通知您。

在以下各节中，您可以访问、审计和处理数据库活动流。

主题

从 Amazon Kinesis 访问活动流

在为数据库启用活动流时，将会为您创建一个 Kinesis 流。您可以从 Kinesis 实时监控数据库活动。要进一步分析数据库活动，您可以将 Kinesis 流连接到使用者应用程序。您还可以将流连接到合规性管理应用程序，例如 IBM 的 Security Guardium 或 Imperva 的 SecureSphere Database Audit and Protection。

您可以从 RDS 控制台或 Kinesis 控制台访问您的 Kinesis 流。

使用 RDS 控制台从 Kinesis 中访问活动流

1. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases（数据库）。
3. 选择已启动活动流的 Amazon RDS 数据库实例。
4. 选择配置。
5. 在 Database activity stream（数据库活动流）下，选择 Kinesis stream（Kinesis 流）下的链接。
6. 在 Kinesis 控制台中，选择 Monitoring（监控）以开始观察数据库活动。

使用 Kinesis 控制台从 Kinesis 中访问活动流

1. 打开 Kinesis 控制台，网址为：https://console.aws.amazon.com/kinesis。
2. 从 Kinesis 流列表中选择您的活动流。

   活动流的名称包含前缀 `aws-rds-das-db-`，后跟数据库的资源 ID。以下是示例。

   ```
   aws-rds-das-db-NHVOV4PCLWHGF52NP
   ```

要使用 Amazon RDS 控制台查找数据库的资源 ID，请从数据库列表中选择数据库实例，然后选择 Configuration（配置）选项卡。

要使用 Amazon CLI 查找活动流的完整 Kinesis 流名称，请使用 describe-db-instances CLI 请求并记下响应中的 `ActivityStreamKinesisStreamName` 值。

3. 选择监控以开始观察数据库活动。

有关使用 Amazon Kinesis 的更多信息，请参阅什么是 Amazon Kinesis Data Streams？

数据库活动流的审计日志内容和示例

受监控的事件在数据库活动流中表示为 JSON 字符串。结构包含一个 JSON 对象，该对象包含一个 `DatabaseActivityMonitoringRecord`，后者反过来包含活动事件的 `databaseActivityEventList` 阵列。

> **Note**
>
> 对于数据库活动流，`paramList` JSON 数组不包含来自 Hibernate 应用程序的空值。

主题

- 活动流的审计日志示例
- DatabaseActivityMonitoringRecords JSON 对象
- databaseActivityEvents JSON 对象

活动流的审计日志示例

以下是活动事件记录的已解密 JSON 审核日志示例。

Example CONNECT SQL 语句的活动事件记录

以下活动事件记录显示 JDBC 瘦客户端（`clientApplication`）使用 `CONNECT` SQL 语句（command）针对 Oracle 数据库进行的登录。

```
{
    "class": "Standard",
    "clientApplication": "JDBC Thin Client",
    "command": "LOGON",
```

```
    "commandText": null,
    "dbid": "0123456789",
    "databaseName": "ORCL",
    "dbProtocol": "oracle",
    "dbUserName": "TEST",
    "endTime": null,
    "errorMessage": null,
    "exitCode": 0,
    "logTime": "2021-01-15 00:15:36.233787",
    "netProtocol": "tcp",
    "objectName": null,
    "objectType": null,
    "paramList": [],
    "pid": 17904,
    "remoteHost": "123.456.789.012",
    "remotePort": "25440",
    "rowCount": null,
    "serverHost": "987.654.321.098",
    "serverType": "oracle",
    "serverVersion": "19.0.0.0.ru-2020-01.rur-2020-01.r1.EE.3",
    "serviceName": "oracle-ee",
    "sessionId": 987654321,
    "startTime": null,
    "statementId": 1,
    "substatementId": null,
    "transactionId": "0000000000000000",
    "engineNativeAuditFields": {
        "UNIFIED_AUDIT_POLICIES": "TEST_POL_EVERYTHING",
        "FGA_POLICY_NAME": null,
        "DV_OBJECT_STATUS": null,
        "SYSTEM_PRIVILEGE_USED": "CREATE SESSION",
        "OLS_LABEL_COMPONENT_TYPE": null,
        "XS_SESSIONID": null,
        "ADDITIONAL_INFO": null,
        "INSTANCE_ID": 1,
        "DBID": 123456789
        "DV_COMMENT": null,
        "RMAN_SESSION_STAMP": null,
        "NEW_NAME": null,
        "DV_ACTION_NAME": null,
        "OLS_PROGRAM_UNIT_NAME": null,
        "OLS_STRING_LABEL": null,
        "RMAN_SESSION_RECID": null,
        "OBJECT_PRIVILEGES": null,
```

```
        "OLS_OLD_VALUE": null,
        "XS_TARGET_PRINCIPAL_NAME": null,
        "XS_NS_ATTRIBUTE": null,
        "XS_NS_NAME": null,
        "DBLINK_INFO": null,
        "AUTHENTICATION_TYPE": "(TYPE\u003d(DATABASE));(CLIENT ADDRESS\u003d((ADDRESS
\u003d(PROTOCOL\u003dtcp)(HOST\u003d205.251.233.183)(PORT\u003d25440))));",
        "OBJECT_EDITION": null,
        "OLS_PRIVILEGES_GRANTED": null,
        "EXCLUDED_USER": null,
        "DV_ACTION_OBJECT_NAME": null,
        "OLS_LABEL_COMPONENT_NAME": null,
        "EXCLUDED_SCHEMA": null,
        "DP_TEXT_PARAMETERS1": null,
        "XS_USER_NAME": null,
        "XS_ENABLED_ROLE": null,
        "XS_NS_ATTRIBUTE_NEW_VAL": null,
        "DIRECT_PATH_NUM_COLUMNS_LOADED": null,
        "AUDIT_OPTION": null,
        "DV_EXTENDED_ACTION_CODE": null,
        "XS_PACKAGE_NAME": null,
        "OLS_NEW_VALUE": null,
        "DV_RETURN_CODE": null,
        "XS_CALLBACK_EVENT_TYPE": null,
        "USERHOST": "a1b2c3d4e5f6.amazon.com",
        "GLOBAL_USERID": null,
        "CLIENT_IDENTIFIER": null,
        "RMAN_OPERATION": null,
        "TERMINAL": "unknown",
        "OS_USERNAME": "sumepate",
        "OLS_MAX_READ_LABEL": null,
        "XS_PROXY_USER_NAME": null,
        "XS_DATASEC_POLICY_NAME": null,
        "DV_FACTOR_CONTEXT": null,
        "OLS_MAX_WRITE_LABEL": null,
        "OLS_PARENT_GROUP_NAME": null,
        "EXCLUDED_OBJECT": null,
        "DV_RULE_SET_NAME": null,
        "EXTERNAL_USERID": null,
        "EXECUTION_ID": null,
        "ROLE": null,
        "PROXY_SESSIONID": 0,
        "DP_BOOLEAN_PARAMETERS1": null,
        "OLS_POLICY_NAME": null,
```

```
        "OLS_GRANTEE": null,
        "OLS_MIN_WRITE_LABEL": null,
        "APPLICATION_CONTEXTS": null,
        "XS_SCHEMA_NAME": null,
        "DV_GRANTEE": null,
        "XS_COOKIE": null,
        "DBPROXY_USERNAME": null,
        "DV_ACTION_CODE": null,
        "OLS_PRIVILEGES_USED": null,
        "RMAN_DEVICE_TYPE": null,
        "XS_NS_ATTRIBUTE_OLD_VAL": null,
        "TARGET_USER": null,
        "XS_ENTITY_TYPE": null,
        "ENTRY_ID": 1,
        "XS_PROCEDURE_NAME": null,
        "XS_INACTIVITY_TIMEOUT": null,
        "RMAN_OBJECT_TYPE": null,
        "SYSTEM_PRIVILEGE": null,
        "NEW_SCHEMA": null,
        "SCN": 5124715
    }
}
```


以下活动事件记录显示 SQL Server 数据库的登录失败。


```
{
    "type": "DatabaseActivityMonitoringRecord",
    "clusterId": "",
    "instanceId": "db-4JCWQLUZVFYP7DIWP6JVQ77O3Q",
    "databaseActivityEventList": [
        {
            "class": "LOGIN",
            "clientApplication": "Microsoft SQL Server Management Studio",
            "command": "LOGIN FAILED",
            "commandText": "Login failed for user 'test'. Reason: Password did not
 match that for the login provided. [CLIENT: local-machine]",
            "databaseName": "",
            "dbProtocol": "SQLSERVER",
            "dbUserName": "test",
            "endTime": null,
            "errorMessage": null,
            "exitCode": 0,
            "logTime": "2022-10-06 21:34:42.7113072+00",
```

```
            "netProtocol": null,
            "objectName": "",
            "objectType": "LOGIN",
            "paramList": null,
            "pid": null,
            "remoteHost": "local machine",
            "remotePort": null,
            "rowCount": 0,
            "serverHost": "172.31.30.159",
            "serverType": "SQLSERVER",
            "serverVersion": "15.00.4073.23.v1.R1",
            "serviceName": "sqlserver-ee",
            "sessionId": 0,
            "startTime": null,
            "statementId": "0x1eb0d1808d34a94b9d3dcf5432750f02",
            "substatementId": 1,
            "transactionId": "0",
            "type": "record",
            "engineNativeAuditFields": {
                "target_database_principal_id": 0,
                "target_server_principal_id": 0,
                "target_database_principal_name": "",
                "server_principal_id": 0,
                "user_defined_information": "",
                "response_rows": 0,
                "database_principal_name": "",
                "target_server_principal_name": "",
                "schema_name": "",
                "is_column_permission": false,
                "object_id": 0,
                "server_instance_name": "EC2AMAZ-NFUJJNO",
                "target_server_principal_sid": null,
                "additional_information": "<action_info "xmlns=\"http://
schemas.microsoft.com/sqlserver/2008/sqlaudit_data\"><pooled_connection>0</
pooled_connection><error>0x00004818</error><state>8</state><address>local machine</
address><PasswordFirstNibbleHash>B</PasswordFirstNibbleHash></action_info>"-->,
                "duration_milliseconds": 0,
                "permission_bitmask": "0x00000000000000000000000000000000",
                "data_sensitivity_information": "",
                "session_server_principal_name": "",
                "connection_id": "98B4F537-0F82-49E3-AB08-B9D33B5893EF",
                "audit_schema_version": 1,
                "database_principal_id": 0,
                "server_principal_sid": null,
```

```
                "user_defined_event_id": 0,
                "host_name": "EC2AMAZ-NFUJJNO"
            }
        }
    ]
}
```


> **Note**
>
> 如果未启用数据库活动流，则 JSON 文档中的最后一个字段为 `"engineNativeAuditFields": { }`。

Example CREATE TABLE 语句的活动事件记录

以下示例显示 Oracle 数据库的 `CREATE TABLE` 事件。


```
{
    "class": "Standard",
    "clientApplication": "sqlplus@ip-12-34-5-678 (TNS V1-V3)",
    "command": "CREATE TABLE",
    "commandText": "CREATE TABLE persons(\n    person_id NUMBER GENERATED BY DEFAULT AS
 IDENTITY,\n    first_name VARCHAR2(50) NOT NULL,\n    last_name VARCHAR2(50) NOT NULL,
\n    PRIMARY KEY(person_id)\n)",
    "dbid": "0123456789",
    "databaseName": "ORCL",
    "dbProtocol": "oracle",
    "dbUserName": "TEST",
    "endTime": null,
    "errorMessage": null,
    "exitCode": 0,
    "logTime": "2021-01-15 00:22:49.535239",
    "netProtocol": "beq",
    "objectName": "PERSONS",
    "objectType": "TEST",
    "paramList": [],
    "pid": 17687,
    "remoteHost": "123.456.789.0",
    "remotePort": null,
    "rowCount": null,
    "serverHost": "987.654.321.01",
    "serverType": "oracle",
```

```
    "serverVersion": "19.0.0.0.ru-2020-01.rur-2020-01.r1.EE.3",
    "serviceName": "oracle-ee",
    "sessionId": 1234567890,
    "startTime": null,
    "statementId": 43,
    "substatementId": null,
    "transactionId": "090011007F0D0000",
    "engineNativeAuditFields": {
        "UNIFIED_AUDIT_POLICIES": "TEST_POL_EVERYTHING",
        "FGA_POLICY_NAME": null,
        "DV_OBJECT_STATUS": null,
        "SYSTEM_PRIVILEGE_USED": "CREATE SEQUENCE, CREATE TABLE",
        "OLS_LABEL_COMPONENT_TYPE": null,
        "XS_SESSIONID": null,
        "ADDITIONAL_INFO": null,
        "INSTANCE_ID": 1,
        "DV_COMMENT": null,
        "RMAN_SESSION_STAMP": null,
        "NEW_NAME": null,
        "DV_ACTION_NAME": null,
        "OLS_PROGRAM_UNIT_NAME": null,
        "OLS_STRING_LABEL": null,
        "RMAN_SESSION_RECID": null,
        "OBJECT_PRIVILEGES": null,
        "OLS_OLD_VALUE": null,
        "XS_TARGET_PRINCIPAL_NAME": null,
        "XS_NS_ATTRIBUTE": null,
        "XS_NS_NAME": null,
        "DBLINK_INFO": null,
        "AUTHENTICATION_TYPE": "(TYPE\u003d(DATABASE));(CLIENT ADDRESS\u003d((PROTOCOL
\u003dbeq)(HOST\u003d123.456.789.0)));",
        "OBJECT_EDITION": null,
        "OLS_PRIVILEGES_GRANTED": null,
        "EXCLUDED_USER": null,
        "DV_ACTION_OBJECT_NAME": null,
        "OLS_LABEL_COMPONENT_NAME": null,
        "EXCLUDED_SCHEMA": null,
        "DP_TEXT_PARAMETERS1": null,
        "XS_USER_NAME": null,
        "XS_ENABLED_ROLE": null,
        "XS_NS_ATTRIBUTE_NEW_VAL": null,
        "DIRECT_PATH_NUM_COLUMNS_LOADED": null,
        "AUDIT_OPTION": null,
        "DV_EXTENDED_ACTION_CODE": null,
```

```
        "XS_PACKAGE_NAME": null,
        "OLS_NEW_VALUE": null,
        "DV_RETURN_CODE": null,
        "XS_CALLBACK_EVENT_TYPE": null,
        "USERHOST": "ip-10-13-0-122",
        "GLOBAL_USERID": null,
        "CLIENT_IDENTIFIER": null,
        "RMAN_OPERATION": null,
        "TERMINAL": "pts/1",
        "OS_USERNAME": "rdsdb",
        "OLS_MAX_READ_LABEL": null,
        "XS_PROXY_USER_NAME": null,
        "XS_DATASEC_POLICY_NAME": null,
        "DV_FACTOR_CONTEXT": null,
        "OLS_MAX_WRITE_LABEL": null,
        "OLS_PARENT_GROUP_NAME": null,
        "EXCLUDED_OBJECT": null,
        "DV_RULE_SET_NAME": null,
        "EXTERNAL_USERID": null,
        "EXECUTION_ID": null,
        "ROLE": null,
        "PROXY_SESSIONID": 0,
        "DP_BOOLEAN_PARAMETERS1": null,
        "OLS_POLICY_NAME": null,
        "OLS_GRANTEE": null,
        "OLS_MIN_WRITE_LABEL": null,
        "APPLICATION_CONTEXTS": null,
        "XS_SCHEMA_NAME": null,
        "DV_GRANTEE": null,
        "XS_COOKIE": null,
        "DBPROXY_USERNAME": null,
        "DV_ACTION_CODE": null,
        "OLS_PRIVILEGES_USED": null,
        "RMAN_DEVICE_TYPE": null,
        "XS_NS_ATTRIBUTE_OLD_VAL": null,
        "TARGET_USER": null,
        "XS_ENTITY_TYPE": null,
        "ENTRY_ID": 12,
        "XS_PROCEDURE_NAME": null,
        "XS_INACTIVITY_TIMEOUT": null,
        "RMAN_OBJECT_TYPE": null,
        "SYSTEM_PRIVILEGE": null,
        "NEW_SCHEMA": null,
        "SCN": 5133083
```

```
    }
}
```

以下示例显示 SQL Server 数据库的 CREATE TABLE 事件。


```
{
    "type": "DatabaseActivityMonitoringRecord",
    "clusterId": "",
    "instanceId": "db-4JCWQLUZVFYP7DIWP6JVQ77O3Q",
    "databaseActivityEventList": [
        {
            "class": "SCHEMA",
            "clientApplication": "Microsoft SQL Server Management Studio - Query",
            "command": "ALTER",
            "commandText": "Create table [testDB].[dbo].[TestTable2](\r\ntextA
 varchar(6000),\r\n    textB varchar(6000)\r\n)",
            "databaseName": "testDB",
            "dbProtocol": "SQLSERVER",
            "dbUserName": "test",
            "endTime": null,
            "errorMessage": null,
            "exitCode": 1,
            "logTime": "2022-10-06 21:44:38.4120677+00",
            "netProtocol": null,
            "objectName": "dbo",
            "objectType": "SCHEMA",
            "paramList": null,
            "pid": null,
            "remoteHost": "local machine",
            "remotePort": null,
            "rowCount": 0,
            "serverHost": "172.31.30.159",
            "serverType": "SQLSERVER",
            "serverVersion": "15.00.4073.23.v1.R1",
            "serviceName": "sqlserver-ee",
            "sessionId": 84,
            "startTime": null,
            "statementId": "0x5178d33d56e95e419558b9607158a5bd",
            "substatementId": 1,
            "transactionId": "4561864",
            "type": "record",
            "engineNativeAuditFields": {
                "target_database_principal_id": 0,
```

```
                "target_server_principal_id": 0,
                "target_database_principal_name": "",
                "server_principal_id": 2,
                "user_defined_information": "",
                "response_rows": 0,
                "database_principal_name": "dbo",
                "target_server_principal_name": "",
                "schema_name": "",
                "is_column_permission": false,
                "object_id": 1,
                "server_instance_name": "EC2AMAZ-NFUJJNO",
                "target_server_principal_sid": null,
                "additional_information": "",
                "duration_milliseconds": 0,
                "permission_bitmask": "0x00000000000000000000000000000000",
                "data_sensitivity_information": "",
                "session_server_principal_name": "test",
                "connection_id": "EE1FE3FD-EF2C-41FD-AF45-9051E0CD983A",
                "audit_schema_version": 1,
                "database_principal_id": 1,
                "server_principal_sid":
 "0x010500000000000515000000bdc2795e2d0717901ba6998cf4010000",
                "user_defined_event_id": 0,
                "host_name": "EC2AMAZ-NFUJJNO"
            }
        }
    ]
}
```


Example SELECT 语句的活动事件记录

以下示例显示 Oracle 数据库的 SELECT 事件。


```
{
    "class": "Standard",
    "clientApplication": "sqlplus@ip-12-34-5-678 (TNS V1-V3)",
    "command": "SELECT",
    "commandText": "select count(*) from persons",
    "databaseName": "1234567890",
    "dbProtocol": "oracle",
    "dbUserName": "TEST",
    "endTime": null,
    "errorMessage": null,
    "exitCode": 0,
```

```
    "logTime": "2021-01-15 00:25:18.850375",
    "netProtocol": "beq",
    "objectName": "PERSONS",
    "objectType": "TEST",
    "paramList": [],
    "pid": 17687,
    "remoteHost": "123.456.789.0",
    "remotePort": null,
    "rowCount": null,
    "serverHost": "987.654.321.09",
    "serverType": "oracle",
    "serverVersion": "19.0.0.0.ru-2020-01.rur-2020-01.r1.EE.3",
    "serviceName": "oracle-ee",
    "sessionId": 1080639707,
    "startTime": null,
    "statementId": 44,
    "substatementId": null,
    "transactionId": null,
    "engineNativeAuditFields": {
        "UNIFIED_AUDIT_POLICIES": "TEST_POL_EVERYTHING",
        "FGA_POLICY_NAME": null,
        "DV_OBJECT_STATUS": null,
        "SYSTEM_PRIVILEGE_USED": null,
        "OLS_LABEL_COMPONENT_TYPE": null,
        "XS_SESSIONID": null,
        "ADDITIONAL_INFO": null,
        "INSTANCE_ID": 1,
        "DV_COMMENT": null,
        "RMAN_SESSION_STAMP": null,
        "NEW_NAME": null,
        "DV_ACTION_NAME": null,
        "OLS_PROGRAM_UNIT_NAME": null,
        "OLS_STRING_LABEL": null,
        "RMAN_SESSION_RECID": null,
        "OBJECT_PRIVILEGES": null,
        "OLS_OLD_VALUE": null,
        "XS_TARGET_PRINCIPAL_NAME": null,
        "XS_NS_ATTRIBUTE": null,
        "XS_NS_NAME": null,
        "DBLINK_INFO": null,
        "AUTHENTICATION_TYPE": "(TYPE\u003d(DATABASE));(CLIENT ADDRESS\u003d((PROTOCOL
\u003dbeq)(HOST\u003d123.456.789.0)));",
        "OBJECT_EDITION": null,
        "OLS_PRIVILEGES_GRANTED": null,
```

```
        "EXCLUDED_USER": null,
        "DV_ACTION_OBJECT_NAME": null,
        "OLS_LABEL_COMPONENT_NAME": null,
        "EXCLUDED_SCHEMA": null,
        "DP_TEXT_PARAMETERS1": null,
        "XS_USER_NAME": null,
        "XS_ENABLED_ROLE": null,
        "XS_NS_ATTRIBUTE_NEW_VAL": null,
        "DIRECT_PATH_NUM_COLUMNS_LOADED": null,
        "AUDIT_OPTION": null,
        "DV_EXTENDED_ACTION_CODE": null,
        "XS_PACKAGE_NAME": null,
        "OLS_NEW_VALUE": null,
        "DV_RETURN_CODE": null,
        "XS_CALLBACK_EVENT_TYPE": null,
        "USERHOST": "ip-12-34-5-678",
        "GLOBAL_USERID": null,
        "CLIENT_IDENTIFIER": null,
        "RMAN_OPERATION": null,
        "TERMINAL": "pts/1",
        "OS_USERNAME": "rdsdb",
        "OLS_MAX_READ_LABEL": null,
        "XS_PROXY_USER_NAME": null,
        "XS_DATASEC_POLICY_NAME": null,
        "DV_FACTOR_CONTEXT": null,
        "OLS_MAX_WRITE_LABEL": null,
        "OLS_PARENT_GROUP_NAME": null,
        "EXCLUDED_OBJECT": null,
        "DV_RULE_SET_NAME": null,
        "EXTERNAL_USERID": null,
        "EXECUTION_ID": null,
        "ROLE": null,
        "PROXY_SESSIONID": 0,
        "DP_BOOLEAN_PARAMETERS1": null,
        "OLS_POLICY_NAME": null,
        "OLS_GRANTEE": null,
        "OLS_MIN_WRITE_LABEL": null,
        "APPLICATION_CONTEXTS": null,
        "XS_SCHEMA_NAME": null,
        "DV_GRANTEE": null,
        "XS_COOKIE": null,
        "DBPROXY_USERNAME": null,
        "DV_ACTION_CODE": null,
        "OLS_PRIVILEGES_USED": null,
```

```
        "RMAN_DEVICE_TYPE": null,
        "XS_NS_ATTRIBUTE_OLD_VAL": null,
        "TARGET_USER": null,
        "XS_ENTITY_TYPE": null,
        "ENTRY_ID": 13,
        "XS_PROCEDURE_NAME": null,
        "XS_INACTIVITY_TIMEOUT": null,
        "RMAN_OBJECT_TYPE": null,
        "SYSTEM_PRIVILEGE": null,
        "NEW_SCHEMA": null,
        "SCN": 5136972
    }
}
```


以下示例显示 SQL Server 数据库的 SELECT 事件。


```
{
    "type": "DatabaseActivityMonitoringRecord",
    "clusterId": "",
    "instanceId": "db-4JCWQLUZVFYP7DIWP6JVQ77O3Q",
    "databaseActivityEventList": [
        {
            "class": "TABLE",
            "clientApplication": "Microsoft SQL Server Management Studio - Query",
            "command": "SELECT",
            "commandText": "select * from [testDB].[dbo].[TestTable]",
            "databaseName": "testDB",
            "dbProtocol": "SQLSERVER",
            "dbUserName": "test",
            "endTime": null,
            "errorMessage": null,
            "exitCode": 1,
            "logTime": "2022-10-06 21:24:59.9422268+00",
            "netProtocol": null,
            "objectName": "TestTable",
            "objectType": "TABLE",
            "paramList": null,
            "pid": null,
            "remoteHost": "local machine",
            "remotePort": null,
            "rowCount": 0,
            "serverHost": "172.31.30.159",
            "serverType": "SQLSERVER",
```

```
            "serverVersion": "15.00.4073.23.v1.R1",
            "serviceName": "sqlserver-ee",
            "sessionId": 62,
            "startTime": null,
            "statementId": "0x03baed90412f564fad640ebe51f89b99",
            "substatementId": 1,
            "transactionId": "4532935",
            "type": "record",
            "engineNativeAuditFields": {
                "target_database_principal_id": 0,
                "target_server_principal_id": 0,
                "target_database_principal_name": "",
                "server_principal_id": 2,
                "user_defined_information": "",
                "response_rows": 0,
                "database_principal_name": "dbo",
                "target_server_principal_name": "",
                "schema_name": "dbo",
                "is_column_permission": true,
                "object_id": 581577110,
                "server_instance_name": "EC2AMAZ-NFUJJNO",
                "target_server_principal_sid": null,
                "additional_information": "",
                "duration_milliseconds": 0,
                "permission_bitmask": "0x00000000000000000000000000000001",
                "data_sensitivity_information": "",
                "session_server_principal_name": "test",
                "connection_id": "AD3A5084-FB83-45C1-8334-E923459A8109",
                "audit_schema_version": 1,
                "database_principal_id": 1,
                "server_principal_sid":
 "0x010500000000000515000000bdc2795e2d0717901ba6998cf4010000",
                "user_defined_event_id": 0,
                "host_name": "EC2AMAZ-NFUJJNO"
            }
        }
    ]
}
```


DatabaseActivityMonitoringRecords JSON 对象

数据库活动事件记录位于包含以下信息的 JSON 对象中。

JSON 字段	数据类型	描述
type	字符串	JSON 记录的类型。值为 DatabaseActivityMonitoringRecords 。
version	字符串	数据库活动监控记录的版本。Oracle 数据库使用版本 1.3，SQL Server 使用版本 1.4。这些引擎版本引入了 engineNativeAuditFields JSON 对象。
databaseActivityEvents	字符串	包含活动事件的 JSON 对象。
键	字符串	用于解密 databaseActivityEventList JSON 数组 的加密密钥

databaseActivityEvents JSON 对象

databaseActivityEvents JSON 对象包含以下信息。

JSON 记录中的顶级字段

审核日志中的每个事件都包装在 JSON 格式的记录中。此记录包含以下字段。

类型

此字段始终具有值 DatabaseActivityMonitoringRecords。

版本

此字段表示数据库活动流数据协议或合同的版本。它定义了哪些字段可用。

databaseActivityEvents

表示一个或多个活动事件的加密字符串。它表示为 base64 字节数组。解密字符串时，结果是 JSON 格式的记录，其中包含字段，如本节中的示例所示。

key

用于加密 databaseActivityEvents 字符串的加密数据密钥。这与您在启动数据库活动流时提供的 Amazon KMS key 密钥相同。

以下示例显示了此记录的格式。

```
{
  "type":"DatabaseActivityMonitoringRecords",
  "version":"1.3",
  "databaseActivityEvents":"encrypted audit records",
  "key":"encrypted key"
}
```

```
         "type":"DatabaseActivityMonitoringRecords",
         "version":"1.4",
         "databaseActivityEvents":"encrypted audit records",
         "key":"encrypted key"
```

执行以下步骤来解密 `databaseActivityEvents` 字段的内容：

1. 使用您在启动数据库活动流时提供的 KMS 密钥解密 `key` JSON 字段中的值。这样做将以明文形式返回数据加密密钥。
2. 对 `databaseActivityEvents` JSON 字段中的值进行 Base64 解码，以获取审核负载的二进制格式的密文。
3. 使用您在第一步中解码的数据加密密钥解密二进制密文。
4. 解压解已解密的负载。
   - 已加密的负载在 `databaseActivityEvents` 字段中。
   - `databaseActivityEventList` 字段包含审核记录数组。此数组中的 `type` 字段可以是 `record` 或 `heartbeat`。

审核日志活动事件记录是包含以下信息的 JSON 对象。

JSON 字段	数据类型	描述
`type`	字符串	JSON 记录的类型。值为 `DatabaseActivityMonitoringRecord` 。
`instanceId`	字符串	数据库实例资源标识符。它对应于数据库实例属性 `DbiResourceId` 。

JSON 字段	数据类型	描述
databaseActivityEventList JSON 数组	字符串	活动审核记录或检测信号消息的数组。

数据库活动流的 databaseActivityEventList JSON 数组

审核日志负载是解密的 `databaseActivityEventList` JSON 数组。以下表列表按字母顺序列出了审计日志的解密 `DatabaseActivityEventList` 数组中每个活动事件的字段。

如果在 Oracle 数据库中启用了统一审计，审计记录将填充在此新的审计跟踪中。`UNIFIED_AUDIT_TRAIL` 视图通过从审计跟踪中检索审计记录以表格形式显示审计记录。启动数据库活动流时，`UNIFIED_AUDIT_TRAIL` 中的一列映射到 `databaseActivityEventList` 数组中的一个字段。

> **Important**
>
> 事件结构可能会发生变化。Amazon RDS 可能会将新字段添加到未来的活动事件中。在解析 JSON 数据的应用程序中，请确保您的代码可以忽略未知字段名称或对其采取适当操作。

Amazon RDS for Oracle 的 databaseActivityEventList 字段

以下是 Amazon RDS for Oracle 的 `databaseActivityEventList` 字段。

字段	数据类型	源	描述
`class`	字符串	`UNIFIED_AUDIT_TRAIL` 中的 `AUDIT_TYPE` 列	活动事件的类。这对应于 `UNIFIED_AUDIT_TRAIL` 视图中的 `AUDIT_TYPE` 列。Amazon RDS for Oracle 的有效值如下： • `Standard` • `FineGrainedAudit`

字段	数据类型	源	描述
			• XS • Database Vault • Label Security • RMAN_AUDIT • Datapump • Direct path API 有关更多信息，请参阅 Oracle 文档中的 [UNIFIED_AUDIT_TRAIL]()。
clientApplication	字符串	CLIENT_PROGRAM_NAME 中的UNIFIED_AUDIT_TRAIL	客户端报告的其用于连接的应用程序。由于客户端不必提供此信息，因此值可以为 null。示例值为 JDBC Thin Client。
command	字符串	UNIFIED_AUDIT_TRAIL 中的 ACTION_NAME 列	用户执行的操作名称。要了解完整操作，请同时阅读命令名称和 AUDIT_TYPE 值。示例值为 ALTER DATABASE。
commandText	字符串	UNIFIED_AUDIT_TRAIL 中的 SQL_TEXT 列	与事件关联的 SQL 语句。示例值为 ALTER DATABASE BEGIN BACKUP。

字段	数据类型	源	描述
databaseName	字符串	V$DATABASE 中的 NAME 列	数据库的名称。
dbid	numb	UNIFIED_AUDIT_TRAIL 中的 DBID 列	数据库的数字标识符。示例值为 1559204751 。
dbProtocol	字符串	不适用	数据库协议。在该测试版中，值为 oracle。
dbUserName	字符串	UNIFIED_AUDIT_TRAIL 中的 DBUSERNAME 列	已审核其操作的数据库用户的名称。示例值为 RDSADMIN。
endTime	字符串	不适用	此字段不用于 RDS for Oracle 且始终为 Null。

字段	数据类型	源	描述
engineNativeAuditFields	objec	UNIFIED_AUDIT_TRAIL	默认情况下，此对象为空。当您使用 --engine-native-audit-fields-included 选项启动活动流时，此对象包括以下列及其值： ADDITIONAL_INFO APPLICATION_CONTEXTS AUDIT_OPTION AUTHENTICATION_TYPE CLIENT_IDENTIFIER CURRENT_USER DBLINK_INFO DBPROXY_USERNAME DIRECT_PATH_NUM_COLUMNS_LOADED DP_BOOLEAN_PARAMETERS1 DP_TEXT_PARAMETERS1 DV_ACTION_CODE DV_ACTION_NAME DV_ACTION_OBJECT_NAME DV_COMMENT DV_EXTENDED_ACTION_CODE DV_FACTOR_CONTEXT DV_GRANTEE DV_OBJECT_STATUS DV_RETURN_CODE DV_RULE_SET_NAME

字段	数据类型	源	描述
			ENTRY_ID EXCLUDED_OBJECT EXCLUDED_SCHEMA EXCLUDED_USER EXECUTION_ID EXTERNAL_USERID FGA_POLICY_NAME GLOBAL_USERID INSTANCE_ID KSACL_SER VICE_NAME KSACL_SOURCE_LOCA TION KSACL_USER_NAME NEW_NAME NEW_SCHEMA OBJECT_EDITION OBJECT_PRIVILEGES OLS_GRANTEE OLS_LABEL_COM PONENT_NAME OLS_LABEL_COMPO NENT_TYPE OLS_MAX_READ_LABEL OLS_MAX_WRITE_LA BEL OLS_MIN_WRITE_ LABEL OLS_NEW_VALUE OLS_OLD_VALUE OLS_PARENT_ GROUP_NAME OLS_POLICY_NAME OLS_PRIVILEGES_GRA NTED OLS_PRIVILEGE S_USED

字段	数据类型	源	描述

			`OLS_PROGRAM_UNIT_NAME`
			`OLS_STRING_LABEL`
			`OS_USERNAME`
			`PROTOCOL_ACTION_NAME`
			`PROTOCOL_MESSAGE`
			`PROTOCOL_RETURN_CODE`
			`PROTOCOL_SESSION_ID`
			`PROTOCOL_USERHOST`
			`PROXY_SESSIONID`
			`RLS_INFO`
			`RMAN_DEVICE_TYPE`
			`RMAN_OBJECT_TYPE`
			`RMAN_OPERATION`
			`RMAN_SESSION_RECID`
			`RMAN_SESSION_STAMP`
			`ROLE`
			`SCN`
			`SYSTEM_PRIVILEGE`
			`SYSTEM_PRIVILEGE_USED`
			`TARGET_USER`
			`TERMINAL`
			`UNIFIED_AUDIT_POLICIES`
			`USERHOST`
			`XS_CALLBACK_EVENT_TYPE`
			`XS_COOKIE`
			`XS_DATASEC_POLICY_NAME`
			`XS_ENABLED_ROLE`
			`XS_ENTITY_TYPE`
			`XS_INACTIVITY_TIMEOUT`

字段	数据类型	源	描述
			XS_NS_ATTRIBUTE XS_NS_ATTRIBUTE_NEW_VAL XS_NS_ATTRIBUTE_OLD_VAL XS_NS_NAME XS_PACKAGE_NAME XS_PROCEDURE_NAME XS_PROXY_USER_NAME XS_SCHEMA_NAME XS_SESSIONID XS_TARGET_PRINCIPAL_NAME XS_USER_NAME 有关更多信息，请参阅 Oracle 数据库文档中的 UNIFIED_AUDIT_TRAIL。
`errorMessage`	字符串	不适用	此字段不用于 RDS for Oracle 且始终为 Null。
`exitCode`	numb	`UNIFIED_AUDIT_TRAIL` 中的 `RETURN_CODE` 列	操作生成的 Oracle 数据库错误代码。如果操作成功，则值为 `0`。
`logTime`	字符串	`UNIFIED_AUDIT_TRAIL` 中的 `EVENT_TIMESTAMP_UTC` 列	创建审计跟踪条目的时间戳。示例值为 `2020-11-27 06:56:14.981404` 。

字段	数据类型	源	描述
netProtocol	字符串	UNIFIED_AUDIT_TRAIL 中的 AUTHENTICATION_TYPE 列	网络通信协议。示例值为 TCP。
objectName	字符串	UNIFIED_AUDIT_TRAIL 中的 OBJECT_NAME 列	受操作影响的对象名称。示例值为 employees 。
objectType	字符串	UNIFIED_AUDIT_TRAIL 中的 OBJECT_SCHEMA 列	受操作影响的对象架构名称。示例值为 hr。
paramList	list	UNIFIED_AUDIT_TRAIL 中的 SQL_BINDS 列	与 SQL_TEXT 关联的绑定变量列表（如有）。示例值为 parameter_1,parameter_2 。
pid	numb	UNIFIED_AUDIT_TRAIL 中的 OS_PROCESS 列	Oracle 数据库进程的操作系统进程标识符。示例值为 22396。
remoteHost	字符串	UNIFIED_AUDIT_TRAIL 中的 AUTHENTICATION_TYPE 列	客户端 IP 地址或生成会话的主机的名称。示例值为 123.456.789.123 。
remotePort	字符串	UNIFIED_AUDIT_TRAIL 中的 AUTHENTICATION_TYPE 列	客户端的端口号。Oracle 数据库环境中的典型值是 1521。
rowCount	numb	不适用	此字段不用于 RDS for Oracle 且始终为 Null。

字段	数据类型	源	描述
serverHost	字符串	数据库主机	数据库服务器主机的 IP 地址。示例值为 123.456.789.123 。
serverType	字符串	不适用	数据库服务器类型。此值始终为 ORACLE。
serverVersion	字符串	数据库主机	Amazon RDS for Oracle 版本、发布更新（RU）和版本更新修订（RUR）。示例值为 19.0.0.0.ru-2020-01.rur-2020-01.r1.EE.3 。
serviceName	字符串	数据库主机	服务的名称。示例值为 oracle-ee 。
sessionId	numb	UNIFIED_AUDIT_TRAIL 中的 SESSIONID 列	审计的会话标识符。示例是 1894327130 。
startTime	字符串	不适用	此字段不用于 RDS for Oracle 且始终为 Null。
statementId	numb	UNIFIED_AUDIT_TRAIL 中的 STATEMENT_ID 列	每个语句运行的数字 ID。一个语句可能会导致多个操作。示例值为 142197。

字段	数据类型	源	描述
substatementId	不适用	不适用	此字段不用于 RDS for Oracle 且始终为 Null。
transactionId	字符串	UNIFIED_AUDIT_TRAIL 中的 TRANSACTION_ID 列	在其中修改对象事务的标识符。示例值为 02000800D5030000 。

Amazon RDS for SQL Server 的 databaseActivityEventList 字段

以下是 Amazon RDS for SQL Server 的 databaseActivityEventList 字段。

字段	数据类型	源	描述
class	字符串	sys.fn_get_audit_file.class_type 映射到 sys.dm_audit_class_type_map.class_type_desc	活动事件的类。有关更多信息，请参阅 Microsoft 文档中的 SQL Server 审计（数据库引擎）。
clientApplication	字符串	sys.fn_get_audit_file.application_name	客户端报告的客户端连接的应用程序（SQL Server 版本 14 及更高版本）。在 SQL Server 版本 13 中，此字段为空。
command	字符串	sys.fn_get_audit_file.action_id 映射到 sys.dm_audit_actions.name	SQL 语句的常规类别。此字段的值取决于类的值。
commandText	字符串	sys.fn_get_audit_file.statement	此字段指示 SQL 语句。

字段	数据类型	源	描述
databaseName	字符串	sys.fn_get_audit_file.database_name	数据库的名称。
dbProtocol	字符串	不适用	数据库协议。该值为 SQLSERVER 。
dbUserName	字符串	sys.fn_get_audit_file.server_principal_name	客户端身份验证的数据库用户。
endTime	字符串	不适用	Amazon RDS for SQL Server 未使用此字段，值为空。
engineNativeAuditFields	object	sys.fn_get_audit_file 中此列未列出的每个字段。	默认情况下，此对象为空。当您使用 --engine-native-audit-fields-included 选项启动活动流时，此对象包括其他原生引擎审计字段，此 JSON 映射不返回这些字段。
errorMessage	字符串	不适用	Amazon RDS for SQL Server 未使用此字段，值为空。
exitCode	整数	sys.fn_get_audit_file.succeeded	表示启动事件的操作是否成功。此字段不能为空。对于除登录事件以外的所有事件，此字段报告权限检查是成功还是失败，但不报告操作是成功还是失败。 值包括： • 0 – 失败 • 1 – 成功
logTime	字符串	sys.fn_get_audit_file.event_time	由 SQL Server 记录的事件时间戳。

字段	数据类型	源	描述
netProtocol	字符串	不适用	Amazon RDS for SQL Server 未使用此字段，值为空。
objectName	字符串	sys.fn_get_audit_file.object_name	数据库对象的名称（如果正在对某个对象运行 SQL 语句）。
objectType	字符串	sys.fn_get_audit_file.class_type 映射到 sys.dm_audit_class_type_map.class_type_desc	数据库对象类型（如果正在对某个对象类型运行 SQL 语句）。
paramList	字符串	不适用	Amazon RDS for SQL Server 未使用此字段，值为空。
pid	整数	不适用	Amazon RDS for SQL Server 未使用此字段，值为空。
remoteHost	字符串	sys.fn_get_audit_file.client_ip	发出 SQL 语句的客户端的 IP 地址或主机名（SQL Server 版本 14 及更高版本）。在 SQL Server 版本 13 中，此字段为空。
remotePort	整数	不适用	Amazon RDS for SQL Server 未使用此字段，值为空。
rowCount	整数	sys.fn_get_audit_file.affected_rows	SQL 语句所影响的表行的数量（SQL Server 版本 14 及更高版本）。此字段位于 SQL Server 版本 13 中。
serverHost	字符串	数据库主机	主机数据库服务器的 IP 地址。
serverType	字符串	不适用	数据库服务器类型。值为 SQLSERVER 。

字段	数据类型	源	描述
serverVersion	字符串	数据库主机	数据库服务器版本，例如，对于 SQL Server 2017，为 15.00.4073.23.v1.R1。
serviceName	字符串	数据库主机	服务的名称。示例值为 sqlserver-ee 。
sessionId	整数	sys.fn_get_audit_file.session_id	会话的唯一标识符。
startTime	字符串	不适用	Amazon RDS for SQL Server 未使用此字段，值为空。
statementId	字符串	sys.fn_get_audit_file.sequence_group_id	客户端的 SQL 语句的唯一标识符。对于生成的每个事件，标识符都不同。示例值为 0x38eaf4156267184094bb82071aaab644 。
substatementId	整数	sys.fn_get_audit_file.sequence_number	用于确定语句的序列号的标识符。当大型记录拆分为多条记录时，此标识符会有所帮助。
transactionId	整数	sys.fn_get_audit_file.transaction_id	事务的标识符。如果没有任何活动的事务，则该值为零。
type	字符串	生成的数据库活动流	事件类型。值为 record 或 heartbeat 。

使用 Amazon 开发工具包处理数据库活动流

您可以使用 Amazon 开发工具包以编程方式处理活动流。以下功能完善的 Java 和 Python 示例是关于使用数据库活动流记录实现基于实例的实施。

Java

```
import java.io.ByteArrayInputStream;
import java.io.ByteArrayOutputStream;
import java.io.IOException;
import java.net.InetAddress;
import java.nio.ByteBuffer;
import java.nio.charset.StandardCharsets;
import java.security.NoSuchAlgorithmException;
import java.security.NoSuchProviderException;
import java.security.Security;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.UUID;
import java.util.zip.GZIPInputStream;

import javax.crypto.Cipher;
import javax.crypto.NoSuchPaddingException;
import javax.crypto.spec.SecretKeySpec;

import com.amazonaws.auth.AWSStaticCredentialsProvider;
import com.amazonaws.auth.BasicAWSCredentials;
import com.amazonaws.encryptionsdk.AwsCrypto;
import com.amazonaws.encryptionsdk.CryptoInputStream;
import com.amazonaws.encryptionsdk.jce.JceMasterKey;
import
 com.amazonaws.services.kinesis.clientlibrary.exceptions.InvalidStateException;
import com.amazonaws.services.kinesis.clientlibrary.exceptions.ShutdownException;
import com.amazonaws.services.kinesis.clientlibrary.exceptions.ThrottlingException;
import com.amazonaws.services.kinesis.clientlibrary.interfaces.IRecordProcessor;
import
 com.amazonaws.services.kinesis.clientlibrary.interfaces.IRecordProcessorCheckpointer;
import
 com.amazonaws.services.kinesis.clientlibrary.interfaces.IRecordProcessorFactory;
import
 com.amazonaws.services.kinesis.clientlibrary.lib.worker.InitialPositionInStream;
import
 com.amazonaws.services.kinesis.clientlibrary.lib.worker.KinesisClientLibConfiguration;
import com.amazonaws.services.kinesis.clientlibrary.lib.worker.ShutdownReason;
import com.amazonaws.services.kinesis.clientlibrary.lib.worker.Worker;
import com.amazonaws.services.kinesis.clientlibrary.lib.worker.Worker.Builder;
import com.amazonaws.services.kinesis.model.Record;
import com.amazonaws.services.kms.AWSKMS;
```

```
import com.amazonaws.services.kms.AWSKMSClientBuilder;
import com.amazonaws.services.kms.model.DecryptRequest;
import com.amazonaws.services.kms.model.DecryptResult;
import com.amazonaws.util.Base64;
import com.amazonaws.util.IOUtils;
import com.google.gson.Gson;
import com.google.gson.GsonBuilder;
import com.google.gson.annotations.SerializedName;
import org.bouncycastle.jce.provider.BouncyCastleProvider;

public class DemoConsumer {

    private static final String STREAM_NAME = "aws-rds-das-[instance-external-
resource-id]"; // aws-rds-das-db-ABCD123456
    private static final String APPLICATION_NAME = "AnyApplication"; //unique
 application name for dynamo table generation that holds kinesis shard tracking
    private static final String AWS_ACCESS_KEY =
 "[AWS_ACCESS_KEY_TO_ACCESS_KINESIS]";
    private static final String AWS_SECRET_KEY =
 "[AWS_SECRET_KEY_TO_ACCESS_KINESIS]";
    private static final String RESOURCE_ID = "[external-resource-id]"; // db-
ABCD123456
    private static final String REGION_NAME = "[region-name]"; //us-east-1, us-
east-2...
    private static final BasicAWSCredentials CREDENTIALS = new
 BasicAWSCredentials(AWS_ACCESS_KEY, AWS_SECRET_KEY);
    private static final AWSStaticCredentialsProvider CREDENTIALS_PROVIDER = new
 AWSStaticCredentialsProvider(CREDENTIALS);

    private static final AwsCrypto CRYPTO = new AwsCrypto();
    private static final AWSKMS KMS = AWSKMSClientBuilder.standard()
            .withRegion(REGION_NAME)
            .withCredentials(CREDENTIALS_PROVIDER).build();

    class Activity {
        String type;
        String version;
        String databaseActivityEvents;
        String key;
    }

    class ActivityEvent {
        @SerializedName("class") String _class;
        String clientApplication;
```

```
        String command;
        String commandText;
        String databaseName;
        String dbProtocol;
        String dbUserName;
        String endTime;
        String errorMessage;
        String exitCode;
        String logTime;
        String netProtocol;
        String objectName;
        String objectType;
        List<String> paramList;
        String pid;
        String remoteHost;
        String remotePort;
        String rowCount;
        String serverHost;
        String serverType;
        String serverVersion;
        String serviceName;
        String sessionId;
        String startTime;
        String statementId;
        String substatementId;
        String transactionId;
        String type;
    }

    class ActivityRecords {
        String type;
        String clusterId; // note that clusterId will contain an empty string on RDS
 Oracle and RDS SQL Server
        String instanceId;
        List<ActivityEvent> databaseActivityEventList;
    }

    static class RecordProcessorFactory implements IRecordProcessorFactory {
        @Override
        public IRecordProcessor createProcessor() {
            return new RecordProcessor();
        }
    }
```

```
    static class RecordProcessor implements IRecordProcessor {

        private static final long BACKOFF_TIME_IN_MILLIS = 3000L;
        private static final int PROCESSING_RETRIES_MAX = 10;
        private static final long CHECKPOINT_INTERVAL_MILLIS = 60000L;
        private static final Gson GSON = new
 GsonBuilder().serializeNulls().create();

        private static final Cipher CIPHER;
        static {
            Security.insertProviderAt(new BouncyCastleProvider(), 1);
            try {
                CIPHER = Cipher.getInstance("AES/GCM/NoPadding", "BC");
            } catch (NoSuchAlgorithmException | NoSuchPaddingException |
 NoSuchProviderException e) {
                throw new ExceptionInInitializerError(e);
            }
        }

        private long nextCheckpointTimeInMillis;

        @Override
        public void initialize(String shardId) {
        }

        @Override
        public void processRecords(final List<Record> records, final
 IRecordProcessorCheckpointer checkpointer) {
            for (final Record record : records) {
                processSingleBlob(record.getData());
            }

            if (System.currentTimeMillis() > nextCheckpointTimeInMillis) {
                checkpoint(checkpointer);
                nextCheckpointTimeInMillis = System.currentTimeMillis() +
 CHECKPOINT_INTERVAL_MILLIS;
            }
        }

        @Override
        public void shutdown(IRecordProcessorCheckpointer checkpointer,
 ShutdownReason reason) {
            if (reason == ShutdownReason.TERMINATE) {
                checkpoint(checkpointer);
```

```
            }
        }

        private void processSingleBlob(final ByteBuffer bytes) {
            try {
                // JSON $Activity
                final Activity activity = GSON.fromJson(new String(bytes.array(), 
StandardCharsets.UTF_8), Activity.class);

                // Base64.Decode
                final byte[] decoded = 
Base64.decode(activity.databaseActivityEvents);
                final byte[] decodedDataKey = Base64.decode(activity.key);

                Map<String, String> context = new HashMap<>();
                context.put("aws:rds:db-id", RESOURCE_ID);

                // Decrypt
                final DecryptRequest decryptRequest = new DecryptRequest()
 
.withCiphertextBlob(ByteBuffer.wrap(decodedDataKey)).withEncryptionContext(context);
                final DecryptResult decryptResult = KMS.decrypt(decryptRequest);
                final byte[] decrypted = decrypt(decoded, 
getByteArray(decryptResult.getPlaintext()));

                // GZip Decompress
                final byte[] decompressed = decompress(decrypted);
                // JSON $ActivityRecords
                final ActivityRecords activityRecords = GSON.fromJson(new 
String(decompressed, StandardCharsets.UTF_8), ActivityRecords.class);

                // Iterate throught $ActivityEvents
                for (final ActivityEvent event : 
activityRecords.databaseActivityEventList) {
                    System.out.println(GSON.toJson(event));
                }
            } catch (Exception e) {
                // Handle error.
                e.printStackTrace();
            }
        }

        private static byte[] decompress(final byte[] src) throws IOException {
```

```
            ByteArrayInputStream byteArrayInputStream = new
ByteArrayInputStream(src);
            GZIPInputStream gzipInputStream = new
GZIPInputStream(byteArrayInputStream);
            return IOUtils.toByteArray(gzipInputStream);
        }

        private void checkpoint(IRecordProcessorCheckpointer checkpointer) {
            for (int i = 0; i < PROCESSING_RETRIES_MAX; i++) {
                try {
                    checkpointer.checkpoint();
                    break;
                } catch (ShutdownException se) {
                    // Ignore checkpoint if the processor instance has been shutdown
(fail over).
                    System.out.println("Caught shutdown exception, skipping
checkpoint." + se);
                    break;
                } catch (ThrottlingException e) {
                    // Backoff and re-attempt checkpoint upon transient failures
                    if (i >= (PROCESSING_RETRIES_MAX - 1)) {
                        System.out.println("Checkpoint failed after " + (i + 1) +
"attempts." + e);
                        break;
                    } else {
                        System.out.println("Transient issue when checkpointing -
attempt " + (i + 1) + " of " + PROCESSING_RETRIES_MAX + e);
                    }
                } catch (InvalidStateException e) {
                    // This indicates an issue with the DynamoDB table (check for
table, provisioned IOPS).
                    System.out.println("Cannot save checkpoint to the DynamoDB table
used by the Amazon Kinesis Client Library." + e);
                    break;
                }
                try {
                    Thread.sleep(BACKOFF_TIME_IN_MILLIS);
                } catch (InterruptedException e) {
                    System.out.println("Interrupted sleep" + e);
                }
            }
        }
    }
```

```
    private static byte[] decrypt(final byte[] decoded, final byte[] decodedDataKey)
 throws IOException {
        // Create a JCE master key provider using the random key and an AES-GCM
 encryption algorithm
        final JceMasterKey masterKey = JceMasterKey.getInstance(new
 SecretKeySpec(decodedDataKey, "AES"),
                "BC", "DataKey", "AES/GCM/NoPadding");
        try (final CryptoInputStream<JceMasterKey> decryptingStream =
 CRYPTO.createDecryptingStream(masterKey, new ByteArrayInputStream(decoded));
             final ByteArrayOutputStream out = new ByteArrayOutputStream()) {
            IOUtils.copy(decryptingStream, out);
            return out.toByteArray();
        }
    }

    public static void main(String[] args) throws Exception {
        final String workerId = InetAddress.getLocalHost().getCanonicalHostName() +
 ":" + UUID.randomUUID();
        final KinesisClientLibConfiguration kinesisClientLibConfiguration =
                new KinesisClientLibConfiguration(APPLICATION_NAME, STREAM_NAME,
 CREDENTIALS_PROVIDER, workerId);

 kinesisClientLibConfiguration.withInitialPositionInStream(InitialPositionInStream.LATEST);
        kinesisClientLibConfiguration.withRegionName(REGION_NAME);
        final Worker worker = new Builder()
                .recordProcessorFactory(new RecordProcessorFactory())
                .config(kinesisClientLibConfiguration)
                .build();

        System.out.printf("Running %s to process stream %s as worker %s...\n",
 APPLICATION_NAME, STREAM_NAME, workerId);

        try {
            worker.run();
        } catch (Throwable t) {
            System.err.println("Caught throwable while processing data.");
            t.printStackTrace();
            System.exit(1);
        }
        System.exit(0);
    }

    private static byte[] getByteArray(final ByteBuffer b) {
        byte[] byteArray = new byte[b.remaining()];
```

```
        b.get(byteArray);
        return byteArray;
    }
}
```

Python

```
import base64
import json
import zlib
import aws_encryption_sdk
from aws_encryption_sdk import CommitmentPolicy
from aws_encryption_sdk.internal.crypto import WrappingKey
from aws_encryption_sdk.key_providers.raw import RawMasterKeyProvider
from aws_encryption_sdk.identifiers import WrappingAlgorithm, EncryptionKeyType
import boto3

REGION_NAME = '<region>'                    # us-east-1
RESOURCE_ID = '<external-resource-id>'      # db-ABCD123456
STREAM_NAME = 'aws-rds-das-' + RESOURCE_ID  # aws-rds-das-db-ABCD123456

enc_client =
 aws_encryption_sdk.EncryptionSDKClient(commitment_policy=CommitmentPolicy.FORBID_ENCRYPT_AL

class MyRawMasterKeyProvider(RawMasterKeyProvider):
    provider_id = "BC"

    def __new__(cls, *args, **kwargs):
        obj = super(RawMasterKeyProvider, cls).__new__(cls)
        return obj

    def __init__(self, plain_key):
        RawMasterKeyProvider.__init__(self)
        self.wrapping_key =
 WrappingKey(wrapping_algorithm=WrappingAlgorithm.AES_256_GCM_IV12_TAG16_NO_PADDING,
                                        wrapping_key=plain_key,
 wrapping_key_type=EncryptionKeyType.SYMMETRIC)

    def _get_raw_key(self, key_id):
        return self.wrapping_key


def decrypt_payload(payload, data_key):
```

```
    my_key_provider = MyRawMasterKeyProvider(data_key)
    my_key_provider.add_master_key("DataKey")
    decrypted_plaintext, header = enc_client.decrypt(
        source=payload,

 materials_manager=aws_encryption_sdk.materials_managers.default.DefaultCryptoMaterialsManag
    return decrypted_plaintext


def decrypt_decompress(payload, key):
    decrypted = decrypt_payload(payload, key)
    return zlib.decompress(decrypted, zlib.MAX_WBITS + 16)


def main():
    session = boto3.session.Session()
    kms = session.client('kms', region_name=REGION_NAME)
    kinesis = session.client('kinesis', region_name=REGION_NAME)

    response = kinesis.describe_stream(StreamName=STREAM_NAME)
    shard_iters = []
    for shard in response['StreamDescription']['Shards']:
        shard_iter_response = kinesis.get_shard_iterator(StreamName=STREAM_NAME,
 ShardId=shard['ShardId'],

 ShardIteratorType='LATEST')
        shard_iters.append(shard_iter_response['ShardIterator'])

    while len(shard_iters) > 0:
        next_shard_iters = []
        for shard_iter in shard_iters:
            response = kinesis.get_records(ShardIterator=shard_iter, Limit=10000)
            for record in response['Records']:
                record_data = record['Data']
                record_data = json.loads(record_data)
                payload_decoded =
 base64.b64decode(record_data['databaseActivityEvents'])
                data_key_decoded = base64.b64decode(record_data['key'])
                data_key_decrypt_result =
 kms.decrypt(CiphertextBlob=data_key_decoded,

 EncryptionContext={'aws:rds:db-id': RESOURCE_ID})
                print (decrypt_decompress(payload_decoded,
 data_key_decrypt_result['Plaintext']))
```

```
            if 'NextShardIterator' in response:
                next_shard_iters.append(response['NextShardIterator'])
        shard_iters = next_shard_iters


if __name__ == '__main__':
    main()
```

数据库活动流的 IAM 策略示例

具有数据库活动流的相应 Amazon Identity and Access Management（IAM）角色权限的任何用户均可以创建、启动、停止和修改数据库实例的活动流设置。这些操作包含在流的审核日志中。对于最佳合规性实践，我们建议您不要向 DBA 提供这些权限。

您可以使用 IAM 策略设置对数据库活动流的访问权限。有关 Amazon RDS 身份验证的更多信息，请参阅Amazon RDS 的 Identity and Access Management。有关创建 IAM 策略的更多信息，请参阅 创建和使用适用于 IAM 数据库访问的 IAM 策略。

Example 允许配置数据库活动流的策略

要为用户提供精细访问权限以修改活动流，请在 IAM 策略中使用服务特定的操作上下文密钥 `rds:StartActivityStream` 和 `rds:StopActivityStream`。以下 IAM 策略示例允许用户或角色配置活动流。

```
{
   "Version":"2012-10-17",
   "Statement":[
        {
            "Sid":"ConfigureActivityStreams",
            "Effect":"Allow",
            "Action": [
                "rds:StartActivityStream",
                "rds:StopActivityStream"
            ],
            "Resource":"*",
          }
     ]
}
```

Example 允许启动数据库活动流的策略

以下 IAM 策略示例允许用户或角色启动活动流。

```
{
    "Version":"2012-10-17",
    "Statement":[
        {
            "Sid":"AllowStartActivityStreams",
            "Effect":"Allow",
            "Action":"rds:StartActivityStream",
            "Resource":"*"
        }
    ]
}
```

Example 允许停止数据库活动流的策略

以下 IAM 策略示例允许用户或角色停止活动流。

```
{
    "Version":"2012-10-17",
    "Statement":[
        {
            "Sid":"AllowStopActivityStreams",
            "Effect":"Allow",
            "Action":"rds:StopActivityStream",
            "Resource":"*"
        }
     ]
}
```

Example 拒绝启动数据库活动流的策略

以下 IAM 策略示例阻止用户或角色启动活动流。

```
{
    "Version":"2012-10-17",
    "Statement":[
        {
            "Sid":"DenyStartActivityStreams",
            "Effect":"Deny",
```

```
            "Action":"rds:StartActivityStream",
            "Resource":"*"
        }
     ]
}
```

Example 拒绝停止数据库活动流的策略

以下 IAM 策略示例阻止用户或角色停止活动流。

```
{
    "Version":"2012-10-17",
    "Statement":[
        {
            "Sid":"DenyStopActivityStreams",
            "Effect":"Deny",
            "Action":"rds:StopActivityStream",
            "Resource":"*"
        }
    ]
}
```

使用 Amazon GuardDuty RDS 保护功能监控威胁

Amazon GuardDuty 是一项威胁检测服务，可帮助保护您的账户、容器、工作负载和 Amazon 环境中的数据。GuardDuty 使用机器学习（ML）模型以及异常和威胁检测功能，持续监控不同的日志源和运行时活动，以识别环境中的潜在安全风险和恶意活动并确定其优先级。

GuardDuty RDS 保护功能会分析和剖析登录事件，找出对 Amazon RDS 数据库的潜在访问威胁。当您开启 RDS 保护功能时，GuardDuty 会监控来自 RDS 数据库的 RDS 登录事件。RDS 保护功能会监控这些事件并对其进行剖析，以了解潜在的内部威胁或外部行为者。

有关启用 GuardDuty RDS 保护功能的更多信息，请参阅《Amazon GuardDuty 用户指南》中的 GuardDuty RDS 保护。

当 RDS 保护功能检测到潜在威胁（例如成功或失败的登录尝试中的异常模式）时，GuardDuty 会生成新的调查结果，其中包含有关可能受损数据库的详细信息。您可以在 Amazon GuardDuty 控制台的调查发现摘要部分查看调查发现的详细信息。调查发现详细信息因调查发现类型而异。主要详细信息、资源类型和资源角色决定了任何调查发现中可用的信息类型。有关调查发现的常见详细信息和调查发现类型的更多信息，请分别参阅《Amazon GuardDuty 用户指南》中的调查发现详细信息和 GuardDuty RDS 保护调查发现类型。

您可以在提供 RDS 保护功能的任何 Amazon Web Services 区域，为任何 Amazon Web Services 账户开启或关闭此功能。未启用 RDS 保护功能时，GuardDuty 不会检测可能受损的 RDS 数据库，也不会提供受损的详细信息。

现有 GuardDuty 账户可以启用 RDS 保护功能，试用期为 30 天。对于新的 GuardDuty 账户，RDS 保护功能已启用，并包含在 30 天免费试用期中。有关更多信息，请参阅《Amazon GuardDuty 用户指南》中的估计 GuardDuty 成本。

有关 GuardDuty 尚不支持 RDS 保护功能的 Amazon Web Services 区域的信息，请参阅《Amazon GuardDuty User Guide》中的 Region-specific feature availability。

有关 GuardDuty RDS 保护支持的 RDS 数据库版本的信息，请参阅《Amazon GuardDuty User Guide》中的 Supported Amazon Aurora, Amazon RDS, and Aurora Limitless databases。

Amazon RDS Custom

Amazon RDS Custom 自动执行数据库管理任务和操作。Amazon RDS Custom 使您能够作为数据库管理员访问和自定义数据库环境和操作系统。使用 RDS Custom，您可以进行自定义以满足传统、自定义和打包应用程序的要求。

有关 RDS Custom 的最新网络研讨会和博客，请参阅 Amazon RDS Custom 资源。

主题

解决数据库自定义的挑战

由于第三方应用程序需要进行自定义，Amazon RDS Custom 将 Amazon RDS 的优势带到一个无法轻易转向完全托管式服务的市场。Amazon RDS Custom 可以节省管理时间、具有耐久性，并且可以随着您的业务扩缩。

如果您需要整个数据库和操作系统由 Amazon 完全托管，我们建议使用 Amazon RDS。如果您需要对数据库和底层操作系统的管理权限才能使相关应用程序可用，Amazon RDS Custom 是更好的选择。如果您想要全面的管理责任并且只需要托管式计算服务，那么最好的选择是在 Amazon EC2 上自行管理您的商业数据库。

为了提供托管式服务体验，Amazon RDS 不允许您访问底层主机。并且 Amazon RDS 还限制了对需要高级权限的某些程序和对象的访问。但是，对于某些应用程序，您可能需要以特权操作系统 (OS) 用户身份执行操作。

例如，您可能需要执行以下操作：

- 安装自定义数据库以及操作系统补丁和程序包。
- 配置特定的数据库设置。

- 将文件系统配置为直接与其应用程序共享文件。

以前，如果您需要自定义应用程序，则必须在本地或 Amazon EC2 上部署数据库。在这种情况下，您承担数据库管理的大部分或全部责任，如下表所述。

特征	本地责任	Amazon EC2 责任	Amazon RDS 责任
应用程序优化	客户	客户	客户
扩展	客户	客户	Amazon
高可用性	客户	客户	Amazon
数据库备份	客户	客户	Amazon
数据库软件修补	客户	客户	Amazon
数据库软件安装	客户	客户	Amazon
操作系统修补	客户	客户	Amazon
操作系统安装	客户	客户	Amazon
服务器维护	客户	Amazon	Amazon
硬件生命周期	客户	Amazon	Amazon
电源、网络和冷却	客户	Amazon	Amazon

当您自行管理数据库软件时，您可以获得更多的控制权，但也更容易出现用户错误。例如，当您进行手动更改时，可能会意外导致应用程序停机。您可能会花几个小时来检查每一项更改，以识别和修复问题。理想情况下，您需要一种托管式数据库服务，该服务可自动执行常见的数据管理员任务，但也支持对数据库和底层操作系统的特权访问。

Amazon RDS Custom 的管理模式和益处

Amazon RDS Custom 是一种托管式数据库服务，适用于需要访问底层操作系统和数据库环境的旧版、自定义和打包应用程序。RDS Custom 可在 Amazon Web Services 云中自动执行数据库的设置、操作和扩展，同时授予您对数据库和底层操作系统的访问权限。通过此访问权限，您可以配置设置、

安装补丁和启用本机特征以满足相关应用程序的要求。使用 RDS Custom，您可以使用 Amazon Web Services Management Console 或 Amazon CLI 运行数据库工作负载。

RDS Custom 只支持 Oracle Database 引擎和 Microsoft SQL Server 引擎。

主题

RDS Custom 中的责任共担模式

借助 RDS Custom，您可以使用 Amazon RDS 的托管式特征，但您可以像在 Amazon EC2 中一样管理主机和自定义操作系统。除了在 Amazon RDS 中执行的任务之外，您还可以承担更多的数据库管理责任。结果是，与在 Amazon RDS 中相比，您可以更好地控制数据库和数据库实例管理，同时还能从 RDS 自动化中受益。

责任共担意味着以下几点：

1. 使用 RDS Custom 特征时，您拥有该过程的一部分。

   例如，在 RDS Custom for Oracle 中，您可以控制要使用哪些 Oracle 数据库补丁以及何时将其应用于您的数据库实例。

2. 您有责任确保对 RDS Custom 特征的任何自定义都能正确工作。

   为了防止无效的自定义，RDS Custom 提供了在数据库实例之外运行的自动化软件。如果您的底层 Amazon EC2 实例受损，RDS Custom 会尝试通过重启或更换 EC2 实例来自动解决这些问题。用户可见的唯一变化是新的 IP 地址。有关更多信息，请参阅 Amazon RDS Custom 主机替换。

下表详细介绍 RDS Custom 的不同特征的责任共担模式。

特征	Amazon EC2 责任	Amazon RDS 责任	RDS Custom for Oracle 责任	RDS Custom for SQL Server 责任
应用程序优化	客户	客户	客户	客户
扩展	客户	Amazon	已共享	已共享

特征	Amazon EC2 责任	Amazon RDS 责任	RDS Custom for Oracle 责任	RDS Custom for SQL Server 责任
高可用性	客户	Amazon	客户	Amazon
数据库备份	客户	Amazon	已共享	Amazon
数据库软件修补	客户	Amazon	已共享	Amazon for RPEV、Customer for CEV[1]
数据库软件安装	客户	Amazon	已共享	Amazon for RPEV、Customer for CEV[1]
操作系统修补	客户	Amazon	Customer	Amazon for RPEV、Customer for CEV[1]
操作系统安装	客户	Amazon	已共享	Amazon
服务器维护	Amazon	Amazon	Amazon	Amazon
硬件生命周期	Amazon	Amazon	Amazon	Amazon
电源、网络和冷却	Amazon	Amazon	Amazon	Amazon

[1] 客户引擎版本（CEV）是数据库版本和 Amazon Machine Image（AMI）的二进制卷快照。RDS 提供的引擎版本（RPEV）是默认 Amazon Machine Image（AMI）和 Microsoft SQL Server 安装。

您可以使用 Microsoft SQL Server 创建 RDS Custom 数据库实例。在本例中：

- 您可以从两种许可模式中进行选择：附带许可证（LI）和自带媒体（BYOM）。
- 使用 LI，您无需单独购买 SQL Server 许可证。Amazon 持有 SQL Server 数据库软件的许可证。
- 使用 BYOM，您可以提供并安装自己的 Microsoft SQL Server 二进制文件和许可。

您可以使用 Oracle 数据库创建 RDS Custom 数据库实例。在此情况下，您可以执行以下操作：

- 管理自己的媒体。

  使用 RDS Custom 时，您可以上载自己的数据库安装文件和补丁。您可以从这些文件中创建自定义引擎版本 (CEV)。然后，您可以通过使用此 CEV 创建 RDS Custom 数据库实例。
- 管理自己的许可证。

  您可以自带 Oracle 数据库许可证并自行管理许可证。

RDS Custom 中的支持外围和不受支持的配置

RDS Custom 提供了一个名为支持外围 的监控功能。此特征可确保您的主机和数据库环境得以正确配置。如果您所做的更改导致数据库实例超出支持外围，则 RDS Custom 会将实例状态更改为 `unsupported-configuration`，直到您手动修复配置问题。有关更多信息，请参阅 RDS Custom 支持外围。

RDS Custom 的主要益处

利用 RDS Custom，您可以执行以下操作：

- 自动执行许多与 Amazon RDS 相同的管理任务，包括以下任务：
  - 数据库生命周期管理
  - 自动备份和时间点恢复 (PITR)
  - 监控 RDS Custom 数据库实例的运行状况并观察对基础设施、操作系统和数据库进程的更改。
  - 根据数据库实例的中断情况，通知或采取措施以修复问题
- 安装第三方应用程序。

  您可以安装软件来运行自定义应用程序和代理。由于您拥有对主机的特权访问权限，因此可以修改文件系统以支持旧版应用程序。
- 安装自定义补丁。

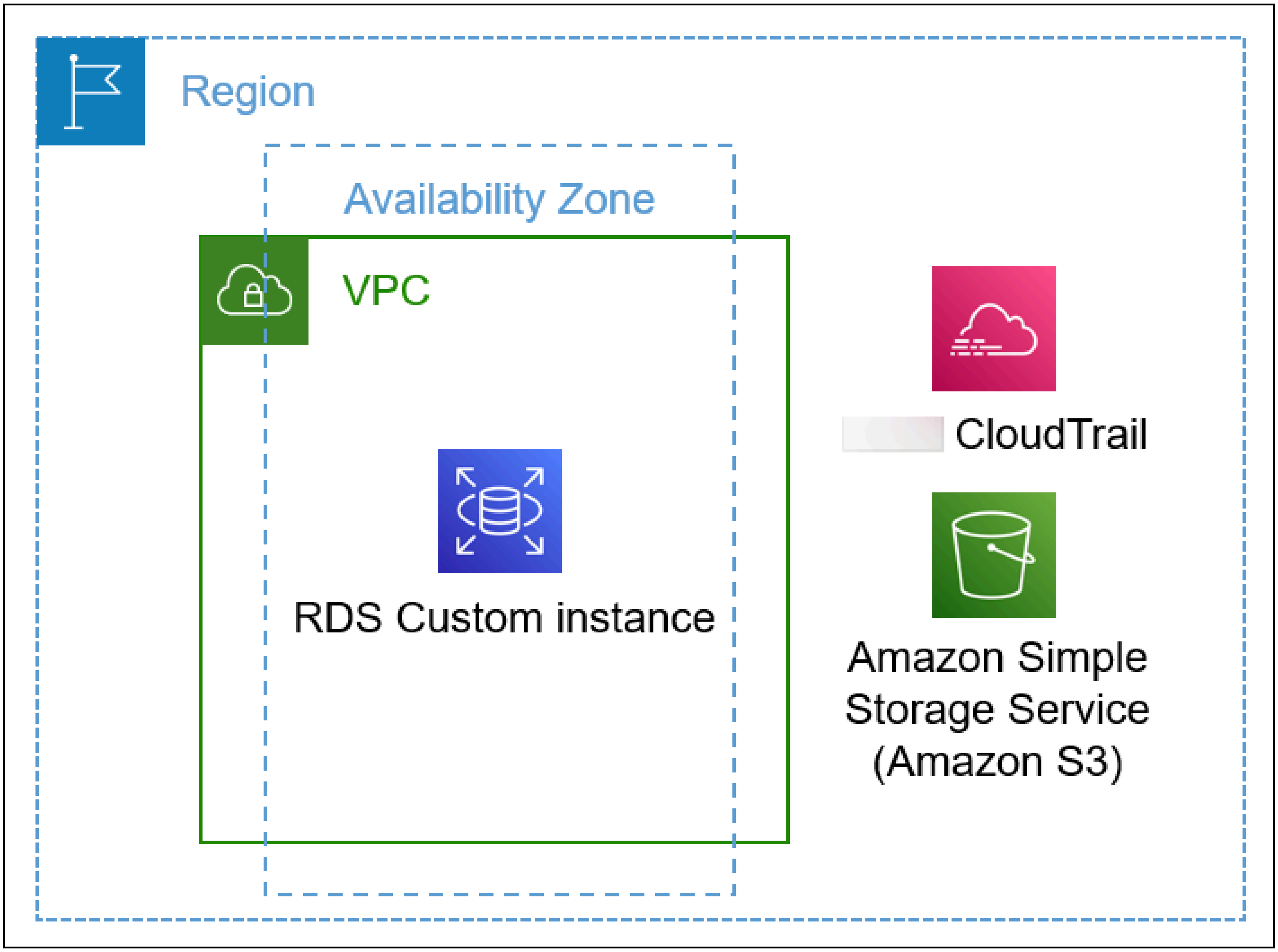
  您可以在 RDS Custom 数据库实例上应用自定义数据库补丁或修改操作系统包。
- 在将本地数据库移动到完全托管式服务之前，将其暂存。

  如果您管理自己的本地数据库，则可以按原样将数据库暂存到 RDS Custom。熟悉云环境后，您可以将数据库迁移到完全托管式 Amazon RDS 数据库实例。
- 创建您自己的自动化。

您可以为报告、管理或诊断工具创建、计划和运行自定义自动化脚本。

Amazon RDS Custom 架构

Amazon RDS Custom 架构基于 Amazon RDS，且具有重要的区别。下图显示 RDS Custom 架构的主要组件。



主题

- VPC
- RDS Custom 自动化和监控
- Amazon S3
- Amazon CloudTrail

VPC

与在 Amazon RDS 中一样，RDS Custom 数据库实例驻留在 Virtual Private Cloud（VPC）中。

RDS Custom 数据库实例包含以下主要组件：

- Amazon EC2 实例
- 实例端点
- Amazon EC2 实例上安装的操作系统
- Amazon EBS 存储，其中包含任何其他文件系统

RDS Custom 自动化和监控

RDS Custom 具有在数据库实例之外运行的自动化软件。该软件与数据库实例上的代理以及整个 RDS Custom 环境中的其他组件通信。

RDS Custom 监控和恢复特征提供与 Amazon RDS 类似的功能。预设情况下，RDS Custom 处于完全自动化模式。自动化软件负有以下主要责任：

- 收集指标并发送通知
- 执行自动实例恢复

RDS Custom 自动化的一项重要责任是响应 Amazon EC2 实例的问题。由于各种原因，主机可能会受损或无法访问。RDS Custom 可以通过重新启动或替换 Amazon EC2 实例来解决这些问题。

主题

Amazon RDS Custom 主机替换

如果 Amazon EC2 主机受损，RDS Custom 会尝试重新启动它。如果重新启动失败，RDS Custom 将使用 Amazon EC2 中包含的相同停止和开启特征。更换主机时，客户唯一可见的更改是新的公有 IP 地址。

主题

停止并启动主机。

RDS Custom 会自动执行以下步骤，无需用户干预：

1. 停止 Amazon EC2 主机。

   EC2 实例将正常关闭并停止运行。所有 Amazon EBS 卷保持连接至实例，而且其数据将保留下来。存储在实例存储卷（RDS Custom 上不支持）或主机 RAM 中的所有数据都将不复存在。

   有关更多信息，请参阅 Amazon EC2 用户指南中的停止和启动您的实例。
2. 启动 Amazon EC2 主机。

   EC2 实例将迁移到新的底层主机硬件。在某些情况下，RDS Custom 数据库实例将仍保留在原始主机上。

主机更换的影响

在 RDS Custom 中，您可以完全控制根设备卷和 Amazon EBS 存储卷。根卷可能包含您不想丢失的重要数据和配置。

RDS Custom for Oracle 会在操作后保留所有数据库和客户数据，包括根卷数据。无需用户干预。在 RDS Custom for SQL Server 上，数据库数据将会保留，但 C: 盘上的所有数据（包括操作系统和客户数据）都将丢失。

在完成主机替换过程之后，Amazon EC2 主机将拥有一个新的公有 IP 地址。新主机将保留下表中所示的数据和元数据。

元数据或数据	RDS Custom for Oracle 保留	RDS Custom for SQL Server 保留
EC2 实例 ID	是	否
EC2 实例元数据	是	是
数据存储卷数据	是	是
根卷数据	是	否
私有 IP 地址	是	是
弹性 IP 地址	是	是

Amazon EC2 主机的最佳实践

Amazon EC2 主机更换特征涵盖了大多数 Amazon EC2 受损情况。我们建议您遵循以下最佳实践：

- 在更改配置或操作系统之前，请备份数据。如果根卷或操作系统受到损坏，那么主机更换将无法修复这一问题。您唯一的选择是从数据库快照还原或采用时间点恢复。
- 不要手动停止或终止物理 Amazon EC2 主机。这两种操作都会导致实例被置于 RDS Custom 支持外围之外。
- (RDS Custom for SQL Server) 如果您将其他卷附加到 Amazon EC2 主机，请将其配置为在重新启动时重新装载。如果主机受损，RDS Custom 可能会自动停止和开启主机。

RDS Custom 支持外围

RDS Custom 提供了名为支持外围的额外监控功能。这项额外的监控功能可确保您的 RDS Custom 数据库实例使用受支持的 Amazon 基础设施、操作系统和数据库。

支持外围检查您的数据库实例是否符合修复 RDS Custom for Oracle 中不支持的配置和修复 RDS Custom for SQL Server 中不支持的配置中列出的要求。如果其中的任何要求未得到满足，则 RDS Custom 会认为数据库实例位于支持外围之外。

主题

- RDS Custom 不支持的配置
- 对不支持的配置进行故障排除

RDS Custom 不支持的配置

当您的数据库实例在支持外围之外时，RDS Custom 会将数据库实例状态更改为 `unsupported-configuration` 并发送事件通知。修复配置问题后，RDS Custom 会将数据库实例状态更改回 `available`。

当数据库实例处于 `unsupported-configuration` 状态时，会出现如下情况：

- 您的数据库可供访问。一种例外情况是当数据库实例处于 `unsupported-configuration` 时，因为数据库正在意外关闭。
- 您无法修改数据库实例。
- 您将无法生成数据库快照。
- 不会创建自动备份。
- 仅限 RDS Custom for SQL Server 数据库实例，如果底层 Amazon EC2 实例受损，RDS Custom 不会替换该实例。有关主机替换的更多信息，请参阅Amazon RDS Custom 主机替换。
- 您可以删除数据库实例，但大多数其他 RDS Custom API 操作都不可用。
- 通过归档重做日志文件并将其上传到 Amazon S3，RDS Custom 可继续支持时间点故障恢复（PITR）。`unsupported-configuration` 状态下的 PITR 在以下方面有所不同：
  - PITR 可能需要花费较长时间，才能完全还原为新的 RDS Custom 数据库实例。发生这种情况，是因为当实例处于 `unsupported-configuration` 状态时，您不能自动或手动生成快照。
  - 在实例进入 `unsupported-configuration` 状态前，PITR 必须从最近生成的快照开始重播更多重做日志。
  - 在某些情况下，数据库实例处于 `unsupported-configuration` 状态，因为您所做的更改阻止了上传归档的重做日志文件。示例包括停止 EC2 实例、停止 RDS Custom 代理和分离 EBS 卷。在此类情况下，PITR 无法将数据库实例恢复到最近的可还原时间点。

对不支持的配置进行故障排除

RDS Custom 为 `unsupported-configuration` 状态提供故障排除指南。虽然有些指南同时适用于 RDS Custom for Oracle 和 RDS Custom for SQL Server，但其他指南取决于您的数据库引擎。有关特定于引擎的故障排除信息，请参阅以下主题：

- 修复 RDS Custom for Oracle 中不支持的配置

- 修复 RDS Custom for SQL Server 中不支持的配置

Amazon S3

如果您使用 RDS Custom for Oracle，可以将安装媒体上传到用户创建的 Amazon S3 存储桶。RDS Custom for Oracle 使用此存储桶中的媒体创建自定义引擎版本 (CEV)。CEV 是数据库版本和 Amazon Machine Image (AMI) 的二进制卷快照。您可以从 CEV 中创建 RDS Custom 数据库实例。有关更多信息，请参阅 使用 Amazon RDS Custom for Oracle 的自定义引擎版本。

RDS Custom for Oracle 和 RDS Custom for SQL Server 都会自动创建一个带有字符串 `do-not-delete-rds-custom-` 作为前缀的 Amazon S3 存储桶。RDS Custom 使用 `do-not-delete-rds-custom-` S3 存储桶存储以下类型的文件：

- RDS Custom 创建的跟踪记录的 Amazon CloudTrail 日志
- 支持外围构件（请参阅RDS Custom 支持外围）
- 数据库恢复日志文件（仅 RDS Custom for Oracle）
- 事务日志（仅 RDS Custom for SQL Server）
- Custom 引擎版本构件（仅 RDS Custom for Oracle）

当您创建以下任一资源时，RDS Custom 会创建 `do-not-delete-rds-custom-` S3 存储桶：

- 您的第一个适用于 RDS Custom for Oracle 的 CEV
- 您的第一个适用于 RDS Custom for SQL Server 的数据库实例

RDS Custom 可为以下各项的每个组合创建一个存储桶：

- Amazon Web Services 账户 ID
- 引擎类型（RDS Custom for Oracle 或 RDS Custom for SQL Server）
- Amazon Web Services 区域

例如，如果您在单个 Amazon Web Services 区域中创建 RDS Custom for Oracle CEV，需要存在一个 `do-not-delete-rds-custom-` 存储桶。如果您创建多个 DS Custom for SQL Server 实例，并且它们驻留在不同的 Amazon Web Services 区域中，需要每个 Amazon Web Services 区域存在一个 `do-not-delete-rds-custom-` 存储桶。如果您在一个 Amazon Web Services 区域中创建一个 RDS

Custom for Oracle 实例和两个 RDS Custom for SQL Server 实例，需要存在两个 `do-not-delete-rds-custom-` 存储桶。

Amazon CloudTrail

RDS Custom 会自动创建名称以 `do-not-delete-rds-custom-` 开头的 Amazon CloudTrail 跟踪记录。RDS Custom 支持外围依赖于来自 CloudTrail 的事件来确定您的操作是否影响 RDS Custom 自动化。有关更多信息，请参阅 对不支持的配置进行故障排除。

RDS Custom 会在您创建第一个数据库实例时创建跟踪记录。RDS Custom 会为以下各项的每个组合创建一条跟踪记录：

- Amazon Web Services 账户 ID
- 引擎类型（RDS Custom for Oracle 或 RDS Custom for SQL Server）
- Amazon Web Services 区域

当您删除 RDS Custom 数据库实例时，不会自动删除此实例的 CloudTrail。在这种情况下，您的 Amazon Web Services 账户继续为未删除的 CloudTrail 计费。RDS Custom 不负责删除此资源。要了解如何手动删除 CloudTrail，请参阅《Amazon CloudTrail 用户指南》中的删除跟踪。

Amazon RDS Custom 中的安全性

自行熟悉 RDS Custom 的安全注意事项。

有关 RDS Custom 的安全性的更多信息，请参阅以下主题。

- 保护 Amazon S3 桶免受混淆代理人问题影响
- 轮换 RDS Custom for Oracle 凭证以遵循合规性计划

RDS Custom 如何代表您安全地管理任务

RDS Custom 使用以下工具和方法来代表您安全地运行操作：

AWSServiceRoleForRDSCustom 服务相关角色

服务相关角色是由服务预定义的，包含该服务代表您调用其他 Amazon Web Services 服务所需的所有权限。对于 RDS Custom，`AWSServiceRoleForRDSCustom` 是根据最低权限原则定义的服务相关角色。RDS Custom 使用 `AmazonRDSCustomServiceRolePolicy` 中的权限（即附加到此角色的策略）来执行大多数预置和所有脱离主机的管理任务。有关更多信息，请参阅 AmazonRDSCustomServiceRolePolicy。

在主机上执行任务时，RDS Custom 自动化通过 Amazon Systems Manager 使用来自服务相关角色的凭证来运行命令。您可以通过 Systems Manager 命令历史记录和 Amazon CloudTrail 来审计命令历史记录。Systems Manager 使用您的网络设置连接到您的 RDS Custom 数据库实例。有关更多信息，请参阅 步骤 4：为 RDS Custom for Oracle 配置 IAM。

临时 IAM 凭证

在预置或删除资源时，RDS Custom 有时会使用从发出调用的 IAM 主体的凭证中派生的临时凭证。这些 IAM 凭证受附加到该主体的 IAM policy 限制，并在操作完成后过期。要了解使用 RDS Custom 的 IAM 主体所需的权限，请参阅步骤 5：为您的 IAM 用户或角色授予所需的权限。

Amazon EC2 实例配置文件

EC2 实例配置文件是 IAM 角色的容器，可用来将角色信息传递给 EC2 实例。EC2 实例是 RDS Custom 数据库实例的基础。创建 RDS Custom 数据库实例时，您需要提供实例配置文件。RDS Custom 在执行基于主机的管理任务（例如备份）时，使用 EC2 实例配置文件凭证。有关更多信息，请参阅 手动创建您的 IAM 角色和实例配置文件。

SSH 密钥对

当 RDS Custom 创建作为数据库实例基础的 EC2 实例时，它会代表您创建 SSH 密钥对。密钥使用命名前缀 `do-not-delete-rds-custom-ssh-privatekey-db-` 或 `rds-custom!oracle-do-not-delete-`*`db_resource_id-uuid`*`-ssh-privatekey`。Amazon Secrets Manager 将此 SSH 私有密钥存储为您的 Amazon Web Services 账户中的密钥。Amazon RDS 不存储、访问或使用这些凭证。有关更多信息，请参阅 Amazon EC2 密钥对和 Linux 实例。

SSL 证书

RDS Custom 数据库实例不支持托管式 SSL 证书。如果要部署 SSL，则可以在自己的钱包中自行管理 SSL 证书，并创建 SSL 侦听器来保护客户端数据库之间的连接或进行数据库复制。有关更多信息，请参阅 Oracle 数据库文档中的配置传输层安全性协议身份验证。

保护 Amazon S3 桶免受混淆代理人问题影响

创建 Amazon RDS Custom for Oracle 自定义引擎版本（CEV）或 RDS Custom for SQL Server 数据库实例时，RDS Custom 会创建一个 Amazon S3 桶。S3 桶存储相关文件，例如 CEV 构件、重做（事务）日志、支持外围的配置项目以及 Amazon CloudTrail 日志。

您可以通过使用全局条件上下文键来防止混淆代理人问题，从而使这些 S3 桶更加安全。有关更多信息，请参阅 防范跨服务混淆代理问题。

以下 RDS Custom for Oracle 示例显示了如何在 S3 桶策略中使用 `aws:SourceArn` 和 `aws:SourceAccount` 全局条件上下文键。对于 RDS Custom for Oracle，请确保包含 CEV 和数据库实例的 Amazon 资源名称（ARN）。对于 RDS Custom for SQL Server，确请保包含数据库实例的 ARN。

```
...
{
  "Sid": "AWSRDSCustomForOracleInstancesObjectLevelAccess",
  "Effect": "Allow",
  "Principal": {
     "Service": "custom.rds.amazonaws.com"
  },
  "Action": [
     "s3:GetObject",
     "s3:GetObjectVersion",
     "s3:DeleteObject",
```

```
        "s3:DeleteObjectVersion",
        "s3:GetObjectRetention",
        "s3:BypassGovernanceRetention"
    ],
    "Resource": "arn:aws:s3:::do-not-delete-rds-custom-123456789012-us-east-2-c8a6f7/
RDSCustomForOracle/Instances/*",
    "Condition": {
        "ArnLike": {
            "aws:SourceArn": [
                "arn:aws:rds:us-east-2:123456789012:db:*",
                "arn:aws:rds:us-east-2:123456789012:cev:*/*"
            ]
        },
        "StringEquals": {
            "aws:SourceAccount": "123456789012"
        }
    }
},
...
```

轮换 RDS Custom for Oracle 凭证以遵循合规性计划

一些合规性计划要求定期更改数据库用户凭证，例如每 90 天更改一次。RDS Custom for Oracle 会自动轮换某些预定义数据库用户的凭证。

主题

- 自动轮换预定义用户的凭证
- 轮换用户凭证的指南
- 手动轮换用户凭证

自动轮换预定义用户的凭证

如果 RDS Custom for Oracle 数据库实例托管在 Amazon RDS 中，则以下预定义 Oracle 用户的凭证每 30 天自动轮换一次。之前用户的凭证位于 Amazon Secrets Manager 中。

数据库用户	创建者	支持的引擎版本	备注
SYS	Oracle	custom-oracle-ee custom-oracle-ee-cdb	

数据库用户	创建者	支持的引擎版本	备注
		custom-oracle-se2 custom-oracle-se2-cdb	
SYSTEM	Oracle	custom-oracle-ee custom-oracle-ee-cdb custom-oracle-se2 custom-oracle-se2-cdb	
RDSADMIN	RDS	custom-oracle-ee custom-oracle-se2	
C##RDSADMIN	RDS	custom-oracle-ee-cdb custom-oracle-se2-cdb	带有 C## 前缀的用户名仅存在于 CDB 中。有关 CDB 的更多信息，请参阅 Amazon RDS Custom for Oracle 架构概述。
RDS_DATAGUARD	RDS	custom-oracle-ee	此用户仅存在于只读副本、只读副本的源数据库以及您已使用 Oracle Data Guard 物理迁移到 RDS Custom 的数据库中。
C##RDS_DATAGUARD	RDS	custom-oracle-ee-cdb	此用户仅存在于只读副本、只读副本的源数据库以及您已使用 Oracle Data Guard 物理迁移到 RDS Custom 的数据库中。带有 C## 前缀的用户名仅存在于 CDB 中。有关 CDB 的更多信息，请参阅 Amazon RDS Custom for Oracle 架构概述。

自动凭证轮换的一个例外是您手动配置为备用数据库的 RDS Custom for Oracle 数据库实例。RDS 仅轮换您已使用 `create-db-instance-read-replica` CLI 命令或 `CreateDBInstanceReadReplica` API 创建的只读副本的凭证。

轮换用户凭证的指南

为确保您的凭证根据您的合规性计划进行轮换，请注意以下指南：

- 如果您的数据库实例自动轮换凭证，请不要手动更改或删除预定义 Oracle 用户中列出的用户的密钥、密码文件或密码。否则，RDS Custom 可能会将您的数据库实例置于支持边界之外，这会暂停自动轮换。
- RDS 主用户不是预定义的，因此您负责手动更改密码或在 Secrets Manager 中设置自动轮换。有关更多信息，请参阅轮换 Amazon Secrets Manager 密钥。

手动轮换用户凭证

对于以下类别的数据库，RDS 不会自动轮换在预定义 Oracle 用户中列出的用户的凭证：

- 您手动配置为用作备用数据库的数据库。
- 本地数据库。
- 位于支持边界之外或处于 RDS Custom 自动化无法运行的状态的数据库实例。在这种情况下，RDS Custom 也不会轮换密钥。

如果您的数据库属于上述任何类别，则必须手动轮换用户凭证。

手动轮换数据库实例的用户凭证

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在数据库中，确保 RDS 当前未备份您的数据库实例或执行配置高可用性等操作。
3. 在数据库详细信息页面中，选择配置并记下数据库实例的资源 ID。或者，您可以使用 Amazon CLI 命令 `describe-db-instances`。
4. 通过 https://console.aws.amazon.com/secretsmanager/ 打开 Secrets Manager 控制台。
5. 在搜索框中，输入数据库的资源 ID，然后使用以下任一命名约定搜索密钥：

```
do-not-delete-rds-custom-resource_id-uuid
```

```
rds-custom!oracle-do-not-delete-resource_id-uuid
```

此密钥存储 RDSADMIN、SYS 和 SYSTEM 的密码。以下示例密钥适用于资源 ID 为 db-ABCDEFG12HIJKLNMNOPQRS3TUVWX 和 UUID 为 123456 的数据库实例：

```
do-not-delete-rds-custom-db-ABCDEFG12HIJKLNMNOPQRS3TUVWX-123456
rds-custom!oracle-do-not-delete-db-ABCDEFG12HIJKLNMNOPQRS3TUVWX-123456
```

> **Important**
>
> 如果您的数据库实例是只读副本并使用 `custom-oracle-ee-cdb` 引擎，则存在两个带有后缀 *db-resource_id-uuid* 的密钥，一个用于主用户，另一个用于 RDSADMIN、SYS 和 SYSTEM。要找到正确的密钥，请在主机上运行以下命令：
>
> ```
> cat /opt/aws/rdscustomagent/config/database_metadata.json | python3 -c
>  "import sys,json; print(json.load(sys.stdin)['dbMonitoringUserPassword'])"
> ```
>
> `dbMonitoringUserPassword` 属性表示 RDSADMIN、SYS 和 SYSTEM 的密钥。

6. 如果您的数据库实例存在于 Oracle Data Guard 配置中，请使用以下任一命名约定搜索密钥：

```
do-not-delete-rds-custom-resource_id-uuid-dg
rds-custom!oracle-do-not-delete-resource_id-uuid-dg
```

此密钥存储了 RDS_DATAGUARD 的密码。以下示例密钥适用于数据库资源 ID 为 db-ABCDEFG12HIJKLNMNOPQRS3TUVWX 和 UUID 为 789012 的数据库实例：

```
do-not-delete-rds-custom-db-ABCDEFG12HIJKLNMNOPQRS3TUVWX-789012-dg
rds-custom!oracle-do-not-delete-db-ABCDEFG12HIJKLNMNOPQRS3TUVWX-789012-dg
```

7. 对于预定义 Oracle 用户中列出的所有数据库用户，按照修改 Amazon Secrets Manager 密钥中的说明更新密码。
8. 如果您的数据库是独立数据库或 Oracle Data Guard 配置中的源数据库：

   a. 启动 Oracle SQL 客户端并以 SYS 身份登录。
   b. 为预定义 Oracle 用户中列出的每个数据库用户运行以下形式的 SQL 语句：

```
ALTER USER user-name IDENTIFIED BY pwd-from-secrets-manager ACCOUNT UNLOCK;
```

例如，如果存储在 Secrets Manager 中的 `RDSADMIN` 的新密码为 `pwd-123`，请运行以下语句：

```
ALTER USER RDSADMIN IDENTIFIED BY pwd-123 ACCOUNT UNLOCK;
```

9. 如果您的数据库实例运行 Oracle Database 12c 版本 1（12.1）并由 Oracle Data Guard 管理，请手动将密码文件（`orapw`）从主数据库实例复制到每个备用数据库实例。

   如果您的数据库实例托管在 Amazon RDS 中，则密码文件位置为 `/rdsdbdata/config/orapw`。对于不在 Amazon RDS 中托管的数据库，在 Linux 和 UNIX 上的原定设置位置为 `$ORACLE_HOME/dbs/orapw$ORACLE_SID`，在 Windows 上的原定设置位置为 `%ORACLE_HOME%\database\PWD%ORACLE_SID%.ora`。

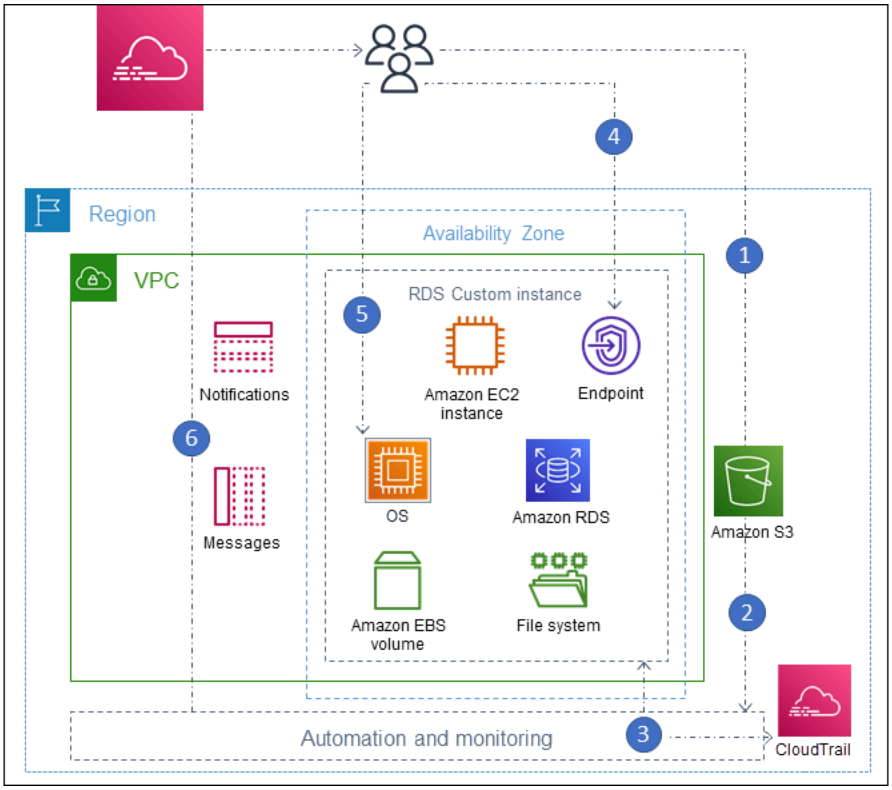
使用 RDS Custom for Oracle

您可在以下内容中找到有关管理和维护 RDS Custom for Oracle 数据库实例的说明。

主题

RDS Custom for Oracle 工作流

下图显示了适用于 RDS Custom for Oracle 的典型工作流。



步骤如下所示：

1. 将数据库软件上载到 Amazon S3 存储桶。

   有关更多信息，请参阅 步骤 3：将安装文件上传到 Amazon S3。
2. 从您的媒体创建 RDS Custom for Oracle 自定义引擎版本（CEV）。

   选择 CDB 架构或传统的非 CDB 架构。有关更多信息，请参阅 创建 CEV。
3. 从 CEV 中创建 RDS Custom for Oracle 数据库实例。

   有关更多信息，请参阅 创建 RDS Custom for Oracle 数据库实例。
4. 将应用程序连接到数据库实例端点。

   有关更多信息，请参阅 使用 SSH 连接到 RDS Custom 数据库实例 和 使用会话管理器连接到您的 RDS Custom 数据库实例。
5. （可选）访问主机以自定义您的软件。
6. 监控 RDS Custom 自动化生成的通知和消息。

数据库安装文件

您对媒体的责任是 Amazon RDS 和 RDS Custom 之间的关键区别。Amazon RDS 是一项完全托管式服务，提供亚马逊机器映像（AMI）和数据库软件。Amazon RDS 数据库软件已预安装，因此您只需选择数据库引擎和版本，然后创建数据库。

对于 RDS Custom，您需要提供自己的媒体。创建自定义引擎版本时，RDS Custom 会安装您提供的媒体。RDS Custom 媒体包含数据库安装文件和补丁。这种服务模型被称为自带媒体 (BYOM)。

RDS Custom for Oracle 的自定义引擎版本

RDS Custom for Oracle 自定义引擎版本（CEV）是数据库版本和 AMI 的二进制卷快照。原定设置情况下，RDS Custom for Oracle 使用 Amazon EC2 提供的最新 AMI。也可以选择重用现有 IAM。

CEV 清单

从 Oracle 下载 Oracle 数据库安装文件后，将这些文件上传到 Amazon S3 桶。创建 CEV 时，您可以在名为 CEV 清单的 JSON 文档中指定文件名。RDS Custom for Oracle 使用指定的文件和 AMI 来创建您的 CEV。

RDS Custom for Oracle 为每个受支持的 Oracle Database 版本提供 JSON 清单模板以及我们推荐的 .zip 文件。例如，以下模板适用于 19.17.0.0.0 RU。

```
{
    "mediaImportTemplateVersion": "2020-08-14",
    "databaseInstallationFileNames": [
        "V982063-01.zip"
    ],
    "opatchFileNames": [
        "p6880880_190000_Linux-x86-64.zip"
    ],
    "psuRuPatchFileNames": [
        "p34419443_190000_Linux-x86-64.zip",
        "p34411846_190000_Linux-x86-64.zip"
    ],
    "otherPatchFileNames": [
        "p28852325_190000_Linux-x86-64.zip",
        "p29997937_190000_Linux-x86-64.zip",
        "p31335037_190000_Linux-x86-64.zip",
        "p32327201_190000_Linux-x86-64.zip",
        "p33613829_190000_Linux-x86-64.zip",
        "p34006614_190000_Linux-x86-64.zip",
```

```
        "p34533061_190000_Linux-x86-64.zip",
        "p34533150_190000_Generic.zip",
        "p28730253_190000_Linux-x86-64.zip",
        "p29213893_1917000DBRU_Generic.zip",
        "p33125873_1917000DBRU_Linux-x86-64.zip",
        "p34446152_1917000DBRU_Linux-x86-64.zip"
    ]
}
```

您也可以在 JSON 清单中指定安装参数。例如，您可以为 Oracle 根目录、Oracle 主目录以及 UNIX/Linux 用户和组的 ID 和名称设置非原定设置值。有关更多信息，请参阅 CEV 清单中的 JSON 字段。

CEV 命名格式

使用客户指定的字符串命名您的 RDS Custom for Oracle CEV。名称格式如下，具体视您的 Oracle Database 版本而定：

- `19.`*`customized_string`*
- `18.`*`customized_string`*
- `12.2.`*`customized_string`*
- `12.1.`*`customized_string`*

您可以使用 1–50 位字母数字字符、下划线、短划线和句点。例如，您可以将 CEV 命名为 `19.my_cev1`。

RDS Custom for Oracle 中的 Oracle 多租户架构

Oracle 多租户架构使 Oracle 数据库能够作为容器数据库（CDB）使用。CDB 包括零个、一个或多个客户创建的可插拔数据库（PDB）。PDB 是模式和对象的可移植集合，在应用程序看来为传统非 CDB。从 Oracle Database 21c 开始，所有 Oracle 数据库都是 CDB。

创建 RDS Custom for Oracle CEV，您需要指定 CDB 或非 CDB 架构。只有当您用来创建 RDS Custom for Oracle CDB 的 CEV 使用 Oracle 多租户架构时，您才能创建此 CDB。有关更多信息，请参阅 使用 Amazon RDS Custom for Oracle 的自定义引擎版本。

为 RDS Custom for Oracle 创建数据库实例

当您创建 CEV 后，CEV 可供使用。您可以创建多个 CEV，并可以从任何 CEV 中创建多个 RDS Custom for Oracle 数据库实例。您还可以更改 CEV 的状态以使其可用或处于非活动状态。

您可以使用 Oracle 多租户架构（custom-oracle-ee-cdb 或 custom-oracle-se2-cdb 引擎类型）或传统的非 CDB 架构（custom-oracle-ee 或 custom-oracle-se2 引擎类型）创建 RDS Custom for Oracle 数据库实例。创建容器数据库（CDB）时，它包含一个可插拔数据库（PDB）和一个 PDB 种子。您可以使用 Oracle SQL 手动创建其他 PDB。

要创建 RDS Custom for Oracle 数据库实例，请使用 create-db-instance 命令。在此命令中，指定要使用哪个 CEV。该过程与创建 Amazon RDS 数据库实例类似。但是，有些参数是不同的。有关更多信息，请参阅 为 Amazon RDS Custom for Oracle 配置数据库实例。

数据库连接

与 Amazon RDS 数据库实例一样，RDS Custom 数据库实例驻留在虚拟私有云（VPC）中。您的应用程序使用 Oracle 侦听器连接到 Oracle 数据库。

如果您的数据库是 CDB，则可以使用侦听器 L_RDSCDB_001 连接到 CDB 根目录和 PDB。如果您将非 CDB 插入 CDB，请务必设置 USE_SID_AS_SERVICE_LISTENER = ON，以便迁移的应用程序保持相同的设置。

当您连接到非 CDB 时，主用户是非 CDB 的用户。当您连接到 CDB 时，主用户是 PDB 的用户。要连接到 CDB 根目录，请登录主机，启动 SQL 客户端，然后使用 SQL 命令创建管理用户。

RDS Custom 自定义

您可以访问 RDS Custom 主机来安装或自定义软件。为避免更改与 RDS Custom 自动化之间发生冲突，您可以在指定的时间内暂停自动化。在此期间，RDS Custom 不执行监控或实例恢复。在此时期结束时，RDS Custom 将恢复完全自动化。有关更多信息，请参阅 暂停和恢复 RDS Custom 数据库实例。

Amazon RDS Custom for Oracle 的数据库架构

RDS Custom for Oracle 同时支持 Oracle 多租户和非多租户架构。

主题

支持的 Oracle 数据库架构

Oracle 多租户架构（也称为 CDB 架构）使 Oracle 数据库能够作为容器数据库（CDB）使用。CDB 包括可插拔数据库（PDB）。PDB 是模式和对象的集合，在应用程序看来是传统的 Oracle 数据库。有关更多信息，请参阅《Oracle 多租户管理员指南》中的多租户架构简介。

CDB 架构和非 CDB 架构是互斥的。如果数据库不是 CDB，则是非 CDB，因此不能包含 PDB。在 RDS Custom for Oracle 中，只有 Oracle Database 19c 支持 CDB 架构。因此，如果您使用以前的 Oracle 数据库版本创建数据库实例，则只能创建非 CDB。有关更多信息，请参阅 多租户架构注意事项。

支持的引擎类型

当您创建 Amazon RDS Custom for Oracle CEV 或数据库实例时，请选择 CDB 引擎类型或非 CDB 引擎类型：

- `custom-oracle-ee-cdb` 和 `custom-oracle-se2-cdb`

  这些引擎类型指定了 Oracle 多租户架构。此选项仅适用于 Oracle Database 19c。当您使用多租户架构创建 RDS for Oracle 数据库实例时，您的 CDB 包含以下容器：

  - CDB 根目录（`CDB$ROOT`）
  - PDB 种子（`PDB$SEED`）
  - 初始 PDB

  您可以使用 Oracle SQL 命令 `CREATE PLUGGABLE DATABASE` 创建更多 PDB。您不能使用 RDS API 创建或删除 PDB。

- `custom-oracle-ee` 和 `custom-oracle-se2`

  这些引擎类型指定了传统的非 CDB 架构。非 CDB 不能包含可插拔数据库（PDB）。

有关更多信息，请参阅 多租户架构注意事项。

Oracle 多租户架构中支持的特征

RDS Custom for Oracle CDB 实例支持以下功能：

- 备份
- 从备份中还原和时间点还原（PITR）
- 只读副本
- 次要版本升级

RDS Custom for Oracle 的功能可用性和支持

在本主题中，您可以查找 RDS Custom for Oracle 功能可用性和支持的摘要以便快速参考。

主题

RDS Custom for Oracle 的 Amazon Web Services 区域和数据库版本支持

功能可用性和支持因每个数据库引擎的特定版本以及 Amazon Web Services 区域而异。有关 RDS Custom for Oracle 的版本和区域可用性的更多信息，请参阅支持 RDS Custom 的区域和数据库引擎。

RDS Custom for Oracle 的数据库版本支持

RDS Custom for Oracle 支持以下 Oracle 数据库版本：

- Oracle Database 19c
- Oracle Database 18c
- Oracle Database 12c 第 2 版 (12.2)
- Oracle Database 12c 第 1 版 (12.1)

RDS Custom for Oracle 的版本和许可支持

RDS Custom for Oracle 在 BYOL 模型上支持 Enterprise Edition（EE）和 Standard Edition 2（SE2）。

请注意 Standard Edition 2 的以下限制：

- 不支持 Oracle Data Guard。因此，您无法创建 Oracle 只读副本。
- 您只能使用具有 16 个或更少 vCPU 的数据库实例类（最多 4 倍大小）。
- Standard Edition 2 上的 CDB 实例最多支持 3 个租户数据库。
- 您无法在 Enterprise Edition 和 Standard Edition 2 之间迁移数据。

RDS Custom for Oracle 的数据库实例类支持

RDS Custom for Oracle 支持以下数据库实例类。如果您在 Standard Edition 2 上创建数据库实例，则只能使用具有 16 个或更少 vCPU 的实例类（最多 4 倍大小）。

类型	大小
db.r7i	db.r7i.large | db.r7i.xlarge | db.r7i.2xlarge | db.r7i.4xlarge | db.r7i.8xlarge | db.r7i.12xlarge | db.r7i.16xlarge | db.r7i.24 xlarge | db.r7i.48xlarge
db.r6i	db.r6i.large | db.r6i.xlarge | db.r6i.2xlarge | db.r6i.4xlarge | db.r6i.8xlarge | db.r6i.12xlarge | db.r6i.16xlarge | db.r6i.24 xlarge | db.r6i.32xlarge
db.r5b	db.r5b.large | db.r5b.xlarge | db.r5b.2xlarge | db.r5b.4xlarge | db.r5b.8xlarge | db.r5b.12xlarge | db.r5b.16xlarge | db.r5b.24 xlarge
db.r5	db.r5.large | db.r5.xlarge | db.r5.2xlarge | db.r5.4xlarge | db.r5.8xlarge | db.r5.12xlarge | db.r5.16xlarge | db.r5.24xlarge
db.x2iec	db.x2iedn.xlarge | db.x2iedn.2xlarge | db.x2iedn.4xlarge | db.x2iedn.8xlarge | db.x2iedn.16xlarge | db.x2iedn.24xlarge | db.x2iedn.32xlarge
db.x2iez	db.x2iezn.2xlarge | db.x2iezn.4xlarge | db.x2iezn.6xlarge | db.x2iezn.8xlarge | db.x2iezn.12xlarge
db.m7i	db.m7i.large | db.m7i.xlarge | db.m7i.2xlarge | db.m7i.4xlarge | db.m7i.8xlarge | db.m7i.12xlarge | db.m7i.16xlarge | db.m7i.24 xlarge | db.m7i.48xlarge
db.m6i	db.m6i.large | db.m6i.xlarge | db.m6i.2xlarge | db.m6i.4xlarge | db.m6i.8xlarge | db.m6i.12xlarge | db.m6i.16xlarge | db.m6i.24 xlarge | db.m6i.32xlarge
db.m5	db.m5.large | db.m5.xlarge | db.m5.2xlarge | db.m5.4xlarge | db.m5.8xlarge | db.m5.12xlarge | db.m5.16xlarge | db.m5.24xlarge

类型	大小
db.t3	db.t3.medium |db.t3.large |db.t3.xlarge |db.t3.2xlarge

RDS Custom for Oracle 的选项组支持

您可以在创建或修改 RDS Custom for Oracle 数据库实例时指定选项组。有关更多信息，请参阅 使用 RDS Custom for Oracle 中的选项组。

RDS Custom for Oracle 的要求和限制

在本主题中，您可以查找 Amazon RDS Custom for Oracle 功能可用性和要求的摘要以便快速参考。

主题

RDS Custom for Oracle 的一般要求

确保符合 Amazon RDS Custom for Oracle 的以下要求：

- 您应有权访问我的 Oracle 支持和 Oracle 软件传输云以下载 RDS Custom for Oracle 支持的安装文件和补丁列表。如果您使用未知的补丁程序，则自定义引擎版本 (CEV) 将创建失败。在这种情况下，请联系 RDS Custom 支持团队并要求添加缺少的补丁程序。有关更多信息，请参阅 步骤 2：从 Oracle 软件传输云下载数据库安装文件和补丁。
- 您对 Amazon S3 具有访问权限。您需要此服务的原因如下：
  - 您将 Oracle 安装文件上传到 S3 桶。您可以使用上传的安装文件创建 RDS Custom CEV。
  - RDS Custom for Oracle 使用从内部定义的 S3 桶下载的脚本对数据库实例执行操作。这些脚本是入门和 RDS Custom 自动化所必需的。
  - RDS Custom for Oracle 将某些文件上传到位于您的客户账户中的 S3 存储桶。这些存储桶使用以下命名格式：`do-not-delete-rds-custom-`*`account_id`*`-`*`region`*`-`*`uuid`*。例如，您可能有一个名为 `do-not-delete-rds-custom-123456789012-us-east-1-12a3b4` 的存储桶。

  有关更多信息，请参阅步骤 3：将安装文件上传到 Amazon S3和创建 CEV。
- 您可以使用RDS Custom for Oracle 的数据库实例类支持中列出的数据库实例类创建 RDS Custom for Oracle 数据库实例。
- 您的 RDS Custom for Oracle 数据库实例运行 Oracle Linux 8（建议）或 Oracle Linux 7。如果您需要 Oracle Linux 7，请联系 Amazon Web Services 支持。有关更多信息，请参阅 RDS Custom for Oracle 数据库升级注意事项。
- 您指定使用 gp2、gp3 或 io1 固态驱动器进行 Amazon EBS 存储。最大存储大小为 64TiB。

- 您拥有创建 RDS Custom for Oracle 数据库实例的 Amazon KMS 密钥。有关更多信息，请参阅 步骤 1：创建或重用对称加密 Amazon KMS 密钥。
- 您拥有创建 RDS Custom for Oracle 数据库实例所需的 Amazon Identity and Access Management（IAM）角色和实例配置文件。有关更多信息，请参阅 步骤 4：为 RDS Custom for Oracle 配置 IAM。
- 创建 CEV 或 RDS Custom 数据库实例的 Amazon Identity and Access Management (IAM) 用户具有 IAM、CloudTrail 和 Amazon S3 所需的权限。

  有关更多信息，请参阅 步骤 5：为您的 IAM 用户或角色授予所需的权限。
- 您可以提供自己的 Virtual Private Cloud (VPC) 和安全组配置。有关更多信息，请参阅 步骤 6：为 RDS Custom for Oracle 配置 VPC。
- 您提供 RDS Custom for Oracle 可用于访问其它 Amazon Web Services 服务的网络配置。有关特定要求，请参阅 步骤 4：为 RDS Custom for Oracle 配置 IAM。

RDS Custom for Oracle 的一般限制

以下限制适用于 RDS Custom for Oracle：

- 您无法修改现有 RDS Custom for Oracle 数据库实例的数据库实例标识符。
- 您无法为 Oracle Database 19c 以外的任何版本指定 Oracle 多租户架构。
- 您无法在单个 RDS Custom for Oracle 数据库实例上创建多个 Oracle 数据库。
- 您无法停止 RDS Custom for Oracle 数据库实例或其底层 Amazon EC2 实例。无法停止 RDS Custom for Oracle 数据库实例的计费。
- 您无法使用自动共享内存管理功能，因为 RDS Custom for Oracle 仅支持自动内存管理。有关更多信息，请参阅《Oracle 数据库管理员指南》中的自动内存管理。
- 确保不要更改主数据库实例的 `DB_UNIQUE_NAME`。更改名称会导致任何还原操作卡住。
- 在同一区域内，不能同时制作超过 20 个快照副本。
- 您无法将 `describe-reserved-db-instances` API 用于 RDS Custom for Oracle 数据库实例。

有关修改 RDS Custom for Oracle 数据库实例的具体限制，请参阅修改 RDS Custom for Oracle 数据库实例。有关复制限制，请参阅RDS Custom for Oracle 复制的一般限制。

RDS Custom for Oracle 的 CEV 和 AMI 限制

以下限制适用于 RDS Custom for Oracle CEV 和 AMI：

- 您不能提供自己的 AMI 以在 RDS Custom for Oracle CEV 中使用。您可以指定使用 Oracle Linux 8 的默认 AMI，也可以指定 RDS Custom for Oracle CEV 以前使用过的 AMI。

> **Note**
>
> 当发现常见漏洞时，RDS Custom for Oracle 会发布新的默认 AMI。没有固定的时间表可用或提供保证。RDS Custom for Oracle 通常每 30 天发布一次新的默认 AMI。

- 您无法修改 CEV 以使用其他 AMI。
- 您无法从使用 `custom-oracle-ee` 或 `custom-oracle-se2` 引擎类型的 CEV 创建 CDB 实例。CEV 必须使用 `custom-oracle-ee-cdb` 或 `custom-oracle-se2-cdb`。
- RDS Custom for Oracle 目前不允许您通过 RDS API 调用升级 RDS Custom for Oracle 数据库实例的操作系统。作为一种解决方法，您可以使用以下命令手动更新操作系统：`sudo yum update --security`。

创建和修改工作流程的不受支持的设置

当您创建或修改 RDS Custom for Oracle 数据库实例时，您无法执行以下操作：

- 更改数据库实例类中的 CPU 内核数和每个内核的线程数。
- 启用存储弹性伸缩。
- 创建多可用区部署。

> **Note**
>
> 有关备选高可用性解决方案，请参阅 Amazon 博客文章[使用只读副本为 Amazon RDS Custom for Oracle 构建高可用性]()。

- 将备份保留期设置为 `0`。
- 配置 Kerberos 身份验证。
- 指定自己的数据库参数组或选项组。
- 开启性能详情。
- 开启自动次要版本升级。

您的 Amazon Web Services 账户的数据库实例限额

确保 RDS Custom 和 Amazon RDS 数据库实例的总数不超过您的配额限制。例如，如果您的 Amazon RDS 配额为 40 个数据库实例，则可以有 20 个 RDS Custom for Oracle 数据库实例和 20 个 Amazon RDS 数据库实例。

为 Amazon RDS Custom for Oracle 设置您的环境

在创建 Amazon RDS Custom for Oracle 数据库实例之前，请执行以下任务。

主题

步骤 1：创建或重用对称加密 Amazon KMS 密钥

客户托管密钥是您的 Amazon 账户中由您创建、拥有和管理的 Amazon KMS keys。RDS Custom 需要客户自主管理型对称加密 KMS 密钥。创建 RDS Custom for Oracle 数据库实例时，您需要提供 KMS 密钥标识符。有关更多信息，请参阅 为 Amazon RDS Custom for Oracle 配置数据库实例。

您有以下选项：

- 如果您的 Amazon Web Services 账户中有现有的客户自主管理型 KMS 密钥，则可以将其与 RDS Custom 一起使用。无需进一步操作。
- 如果您已经为不同的 RDS Custom 引擎创建了客户自主管理型对称加密 KMS 密钥，可以重用相同的 KMS 密钥。无需进一步操作。
- 如果您的账户中没有现有的客户自主管理型对称加密 KMS 密钥，请按照《Amazon Key Management Service 开发人员指南》的创建密钥中的说明创建一个 KMS 密钥。
- 如果您正在创建 CEV 或 RDS Custom 数据库实例，而您的 KMS 密钥位于其他 Amazon Web Services 账户 中，请务必使用 Amazon CLI。您不能将 Amazon 控制台与跨账户 KMS 密钥一起使用。

> **Important**
>
> RDS Custom 不支持 Amazon 托管式 KMS 密钥。

确保您的对称加密密钥为 IAM 实例配置文件中的 `kms:Decrypt`（IAM）角色授予对 `kms:GenerateDataKey` 和 Amazon Identity and Access Management 操作的访问权限。如果您的账户中有一个新的对称加密密钥，不需要进行任何更改。否则，请确保对称加密密钥的策略授予对这些操作的访问权限。

有关更多信息，请参阅 步骤 4：为 RDS Custom for Oracle 配置 IAM。

有关为 RDS Custom for Oracle 配置 IAM 的更多信息，请参阅步骤 4：为 RDS Custom for Oracle 配置 IAM。

步骤 2：下载并安装 Amazon CLI

Amazon 为您提供了命令行界面以使用 RDS Custom 功能。您可以使用 Amazon CLI 的版本 1 或版本 2。

有关下载和安装 Amazon CLI 的信息，请参阅安装或更新 Amazon CLI 的最新版本。

如果满足以下任一条件，请跳过此步骤：

- 您计划仅从 Amazon Web Services Management Console访问 RDS Custom。
- 您已为 Amazon RDS 或其他 RDS Custom 数据库引擎下载了 Amazon CLI。

步骤 3：提取 RDS Custom for Oracle 的 CloudFormation 模板

为了简化设置，我们强烈建议您使用 Amazon CloudFormation 模板创建 CloudFormation 堆栈。如果您计划手动配置 IAM 和 VPC，请跳过此步骤。

主题

步骤 3a：下载 CloudFormation 模板文件

CloudFormation 模板是对构成堆栈的 Amazon 资源的声明。此模板存储为 JSON 文件。

下载 CloudFormation 模板文件

1. 打开链接 custom-oracle-iam.zip 的上下文（右键单击）菜单，然后选择 Save Link As（将链接另存为）。

2. 将该文件保存到您的计算机。
3. 对于链接 custom-vpc.zip 重复之前的步骤。

   如果您已经配置了适用于 RDS Custom 的 VPC，请跳过此步骤。

步骤 3b：提取 custom-oracle-iam.json

打开您下载的 `custom-oracle-iam.zip` 文件，然后解压缩文件 `custom-oracle-iam.json`。文件的开头如下所示。

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Parameters": {
    "EncryptionKey": {
      "Type": "String",
      "Default": "*",
      "Description": "KMS Key ARN for encryption of data managed by RDS Custom and by
 DB Instances."
    }
  },
  "Resources": {
    "RDSCustomInstanceServiceRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "RoleName": { "Fn::Sub": "AWSRDSCustomInstanceRole-${AWS::Region}" },
        "AssumeRolePolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Action": "sts:AssumeRole",
              "Effect": "Allow",
              "Principal": {
                "Service": "ec2.amazonaws.com"
              }
            }
          ]
        },...
```

步骤 3c：提取 custom-vpc.json

> **Note**
>
> 如果您为 RDS Custom for Oracle 配置了现有 VPC，则跳过此步骤。有关更多信息，请参阅[为 RDS Custom for Oracle 手动配置 VPC](#)。

打开您下载的 `custom-vpc.zip` 文件，然后解压缩文件 `custom-vpc.json`。文件的开头如下所示。

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Parameters": {
    "PrivateVpc": {
      "Type": "AWS::EC2::VPC::Id",
      "Description": "Private VPC Id to use for RDS Custom DB Instances"
    },
    "PrivateSubnets": {
      "Type": "List<AWS::EC2::Subnet::Id>",
      "Description": "Private Subnets to use for RDS Custom DB Instances"
    },
    "RouteTable": {
      "Type": "String",
      "Description": "Route Table that must be associated with the PrivateSubnets and
 used by S3 VPC Endpoint",
      "AllowedPattern": "rtb-[0-9a-z]+"
    }
  },
  "Resources": {
    "DBSubnetGroup": {
      "Type": "AWS::RDS::DBSubnetGroup",
      "Properties": {
        "DBSubnetGroupName": "rds-custom-private",
        "DBSubnetGroupDescription": "RDS Custom Private Network",
        "SubnetIds": {
          "Ref": "PrivateSubnets"
        }
      }
    },...
```

步骤 4：为 RDS Custom for Oracle 配置 IAM

您使用 IAM 角色或 IAM 用户（称为 IAM 实体），通过控制台或 Amazon CLI 创建 RDS Custom 数据库实例。此 IAM 实体必须具有创建实例所需的权限。

您可以使用 CloudFormation 或手动步骤配置 IAM。

> **Important**
>
> 强烈建议您使用 Amazon CloudFormation 配置您的 RDS Custom for Oracle 环境。这种技术最简单，最不容易出错。

主题

- [使用 CloudFormation 配置 IAM]
- [手动创建您的 IAM 角色和实例配置文件]

使用 CloudFormation 配置 IAM

当您对 IAM 使用 CloudFormation 模板时，该模板会创建以下必需的资源：

- 名为 `AWSRDSCustomInstanceProfile-region` 的实例配置文件
- 一个名为 `AWSRDSCustomInstanceRole-region` 的服务角色
- 附加到服务角色的名为 `AWSRDSCustomIamRolePolicy` 的访问策略

要采用 CloudFormation 配置 IAM

1. 通过以下网址打开 CloudFormation 控制台：https://console.aws.amazon.com/cloudformation。
2. 开启 Create Stack（创建堆栈）向导，然后选择 Create Stack（创建堆栈）。
3. 在 Create stack（创建堆栈）页面上，执行以下操作：
   a. 对于 Prepare template（准备模板），选择 Template is ready（模板就绪）。
   b. 对于 Template source(模板来源)，选择 Upload a template file(上载模板文件)。
   c. 对于选择文件，请导航到 custom-oracle-iam.json 并选择此文件。
   d. 选择下一步。
4. 在 Specify stack details（指定堆栈详细信息）页面中，请执行以下操作：

   a. 对于堆栈名称，输入 **`custom-oracle-iam`**。
   b. 选择 Next(下一步)。
5. 在 配置堆栈选项 页面上，请选择 下一步。
6. 在 Review custom-oracle-iam（查看 custom-oracle-iam）页面上，请执行以下操作：
   a. 选中我确认 Amazon CloudFormation 可能使用自定义名称创建 IAM 资源复选框。
   b. 选择提交。

   CloudFormation 创建 RDS Custom for Oracle 所需的 IAM 角色。在左侧面板中，当 custom-oracle-iam 显示 CREATE_COMPLETE 时，继续执行下一步。
7. 在左侧面板中，选择 custom-oracle-iam。在右侧面板中，执行以下操作：
   a. 选择堆栈信息。堆栈的 ID 格式为：arn:aws:cloudformation:***`region`***:***`account-no`***:stack/custom-oracle-iam/***`identifier`***。
   b. 选择资源。您将看到以下内容：
      - 名为 AWSRDSCustomInstanceProfile-***`region`*** 的实例配置文件
      - 名为 AWSRDSCustomInstanceRole-***`region`*** 的服务角色

      创建 RDS Custom 数据库实例时，您需要提供实例配置文件 ID。

手动创建您的 IAM 角色和实例配置文件

使用 CloudFormation 时，配置是最简单的。但是，您还可以手动配置 IAM。对于手动设置，请执行以下操作：

- 步骤 1：创建 IAM 角色 AWSRDSCustomInstanceRoleForRdsCustomInstance.
- 步骤 2：将访问策略添加到 AWSRDSCustomInstanceRoleForRdsCustomInstance 中.
- 步骤 2：将访问策略添加到 AWSRDSCustomInstanceRoleForRdsCustomInstance 中.
- 步骤 4：将 AWSRDSCustomInstanceRoleForRdsCustomInstance 添加到 AWSRDSCustomInstanceProfile.

步骤 1：创建 IAM 角色 AWSRDSCustomInstanceRoleForRdsCustomInstance

在此步骤中，您将使用命名格式 `AWSRDSCustomInstanceRole-region` 创建角色。使用信任策略，以便 Amazon EC2 可以代入该角色。以下示例假设您已将环境变量 `$REGION` 设置为要在其中创建数据库实例的 Amazon Web Services 区域。

```
aws iam create-role \
  --role-name AWSRDSCustomInstanceRole-$REGION \
  --assume-role-policy-document '{
    "Version": "2012-10-17",
      "Statement": [
        {
          "Action": "sts:AssumeRole",
          "Effect": "Allow",
          "Principal": {
              "Service": "ec2.amazonaws.com"
          }
        }
      ]
    }'
```

步骤 2：将访问策略添加到 AWSRDSCustomInstanceRoleForRdsCustomInstance 中

在将内联策略嵌入 IAM 角色中时，内联策略将用作角色的访问（权限）策略的一部分。您将创建 `AWSRDSCustomIamRolePolicy` 策略，以允许 Amazon EC2 发送和接收消息，以及执行各种操作。

以下示例创建 `AWSRDSCustomIamRolePolicy` 名为的访问策略，然后将其添加到 IAM 角色 `AWSRDSCustomInstanceRole-region`。此示例假定您已设置以下环境变量：

`$REGION`

将此变量设置为您计划在其中创建数据库实例的 Amazon Web Services 区域。

`$ACCOUNT_ID`

将此变量设置为您的 Amazon Web Services 账户号码。

`$KMS_KEY`

将此变量设置为您要用于 RDS Custom 数据库实例的 Amazon KMS key 的 Amazon 资源名称（ARN）。要指定多个 KMS 密钥，请将其添加到语句 ID (Sid) 11 的 `Resources` 部分。

```
aws iam put-role-policy \
```

```
  --role-name AWSRDSCustomInstanceRole-$REGION \
  --policy-name AWSRDSCustomIamRolePolicy \
  --policy-document '{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "1",
            "Effect": "Allow",
            "Action": [
                "ssm:DescribeAssociation",
                "ssm:GetDeployablePatchSnapshotForInstance",
                "ssm:GetDocument",
                "ssm:DescribeDocument",
                "ssm:GetManifest",
                "ssm:GetParameter",
                "ssm:GetParameters",
                "ssm:ListAssociations",
                "ssm:ListInstanceAssociations",
                "ssm:PutInventory",
                "ssm:PutComplianceItems",
                "ssm:PutConfigurePackageResult",
                "ssm:UpdateAssociationStatus",
                "ssm:UpdateInstanceAssociationStatus",
                "ssm:UpdateInstanceInformation",
                "ssm:GetConnectionStatus",
                "ssm:DescribeInstanceInformation",
                "ssmmessages:CreateControlChannel",
                "ssmmessages:CreateDataChannel",
                "ssmmessages:OpenControlChannel",
                "ssmmessages:OpenDataChannel"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Sid": "2",
            "Effect": "Allow",
            "Action": [
                "ec2messages:AcknowledgeMessage",
                "ec2messages:DeleteMessage",
                "ec2messages:FailMessage",
                "ec2messages:GetEndpoint",
                "ec2messages:GetMessages",
```

```
                "ec2messages:SendReply"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Sid": "3",
            "Effect": "Allow",
            "Action": [
                "logs:PutRetentionPolicy",
                "logs:PutLogEvents",
                "logs:DescribeLogStreams",
                "logs:DescribeLogGroups",
                "logs:CreateLogStream",
                "logs:CreateLogGroup"
            ],
            "Resource": [
                "arn:aws:logs:'$REGION':'$ACCOUNT_ID':log-group:rds-custom-instance*"
            ]
        },
        {
            "Sid": "4",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": [
                "arn:aws:s3:::do-not-delete-rds-custom-*/*"
            ]
        },
        {
            "Sid": "5",
            "Effect": "Allow",
            "Action": [
                "cloudwatch:PutMetricData"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringEquals": {
```

```
                    "cloudwatch:namespace": [
                        "RDSCustomForOracle/Agent"
                    ]
                }
            }
        },
        {
            "Sid": "6",
            "Effect": "Allow",
            "Action": [
                "events:PutEvents"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Sid": "7",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue",
                "secretsmanager:DescribeSecret"
            ],
            "Resource": [
                "arn:aws:secretsmanager:'$REGION':'$ACCOUNT_ID':secret:do-not-delete-
rds-custom-*",
                "arn:aws:secretsmanager:'$REGION':'$ACCOUNT_ID':secret:rds-custom!
oracle-do-not-delete-*"
            ]
        },
        {
           "Sid": "8",
           "Effect": "Allow",
           "Action": [
             "s3:ListBucketVersions"
           ],
           "Resource": [
             "arn:aws:s3:::do-not-delete-rds-custom-*"
           ]
         },
         {
            "Sid": "9",
            "Effect": "Allow",
            "Action": "ec2:CreateSnapshots",
```

```
            "Resource": [
                "arn:aws:ec2:*:*:instance/*",
                "arn:aws:ec2:*:*:volume/*"
            ],
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/AWSRDSCustom": "custom-oracle"
                }
            }
          },
          {
            "Sid": "10",
            "Effect": "Allow",
            "Action": "ec2:CreateSnapshots",
            "Resource": [
                "arn:aws:ec2:*::snapshot/*"
            ]
          },
          {
            "Sid": "11",
            "Effect": "Allow",
            "Action": [
              "kms:Decrypt",
              "kms:GenerateDataKey"
            ],
            "Resource": [
              "arn:aws:kms:'$REGION':'$ACCOUNT_ID':key/'$KMS_KEY'"
            ]
          },
          {
            "Sid": "12",
            "Effect": "Allow",
            "Action": "ec2:CreateTags",
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "ec2:CreateAction": [
                        "CreateSnapshots"
                    ]
                }
            }
        }
    ]
```

```
}'
```

步骤 3：创建 RDS Custom 实例配置文件 AWSRDSCustomInstanceProfile

实例配置文件是包含单个 IAM 角色的容器。RDS Custom 使用实例配置文件将角色传递给实例。

如果您使用 CLI 创建角色，则是以单独操作的形式创建角色和实例配置文件，可以为它们提供不同的名称。按如下方式创建您的 IAM 实例配置文件，并使用格式 `AWSRDSCustomInstanceProfile-region` 对该配置文件命名。以下示例假设您已将环境变量 `$REGION` 设置为要在其中创建数据库实例的 Amazon Web Services 区域。

```
aws iam create-instance-profile \
    --instance-profile-name AWSRDSCustomInstanceProfile-$REGION
```

步骤 4：将 AWSRDSCustomInstanceRoleForRdsCustomInstance 添加到 AWSRDSCustomInstanceProfile

将您的 IAM 角色添加到您之前创建的实例配置文件。以下示例假设您已将环境变量 `$REGION` 设置为要在其中创建数据库实例的 Amazon Web Services 区域。

```
aws iam add-role-to-instance-profile \
    --instance-profile-name AWSRDSCustomInstanceProfile-$REGION \
    --role-name AWSRDSCustomInstanceRole-$REGION
```

步骤 5：为您的 IAM 用户或角色授予所需的权限

确保创建 CEV 或 RDS Custom 数据库实例的 IAM 主体（用户或角色）具有以下任一策略：

- `AdministratorAccess` 策略
- 具有 Amazon S3 和 Amazon KMS、CEV 创建和数据库实例创建所需权限的 `AmazonRDSFullAccess` 策略

主题

- Amazon S3 和 Amazon KMS 所需的 IAM 权限
- 创建 CEV 所需的 IAM 权限
- 从 CEV 创建数据库实例所需的 IAM 权限

Amazon S3 和 Amazon KMS 所需的 IAM 权限

要创建 CEV 或 RDS Custom for Oracle 数据库实例，IAM 主体需要访问 Amazon S3 和 Amazon KMS。以下示例 JSON 策略授予所需的权限。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "CreateS3Bucket",
            "Effect": "Allow",
            "Action": [
                "s3:CreateBucket",
                "s3:PutBucketPolicy",
                "s3:PutBucketObjectLockConfiguration",
                "s3:PutBucketVersioning"
            ],
            "Resource": "arn:aws:s3:::do-not-delete-rds-custom-*"
        },
        {
            "Sid": "CreateKmsGrant",
            "Effect": "Allow",
            "Action": [
                "kms:CreateGrant",
                "kms:DescribeKey"
            ],
            "Resource": "*"
        }
    ]
}
```

有关 `kms:CreateGrant` 权限的更多信息，请参阅 Amazon KMS key 管理。

创建 CEV 所需的 IAM 权限

要创建 CEV，IAM 主体需要以下额外权限：

```
s3:GetObjectAcl
s3:GetObject
s3:GetObjectTagging
s3:ListBucket
mediaimport:CreateDatabaseBinarySnapshot
```

以下示例 JSON 策略授予访问桶 *my-custom-installation-files* 及其内容所需的额外权限。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AccessToS3MediaBucket",
            "Effect": "Allow",
            "Action": [
                "s3:GetObjectAcl",
                "s3:GetObject",
                "s3:GetObjectTagging",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::my-custom-installation-files",
                "arn:aws:s3:::my-custom-installation-files/*"
            ]
        },
        {
            "Sid": "PermissionForByom",
            "Effect": "Allow",
            "Action": [
                "mediaimport:CreateDatabaseBinarySnapshot"
            ],
            "Resource": "*"
        }
    ]
}
```

您可以使用 S3 存储桶策略向调用者的账户授予 Amazon S3 的类似权限。

从 CEV 创建数据库实例所需的 IAM 权限

要从现有的 CEV 创建 RDS Custom for Oracle 数据库实例，IAM 主体需要以下额外权限。

```
iam:SimulatePrincipalPolicy
cloudtrail:CreateTrail
cloudtrail:StartLogging
```

以下示例 JSON 策略授予验证 IAM 角色和将信息记录到 Amazon CloudTrail 所需的权限。

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ValidateIamRole",
            "Effect": "Allow",
            "Action": "iam:SimulatePrincipalPolicy",
            "Resource": "*"
        },
        {
            "Sid": "CreateCloudTrail",
            "Effect": "Allow",
            "Action": [
                "cloudtrail:CreateTrail",
                "cloudtrail:StartLogging"
            ],
            "Resource": "arn:aws:cloudtrail:*:*:trail/do-not-delete-rds-custom-*"
        }
    ]
}
```

步骤 6：为 RDS Custom for Oracle 配置 VPC

就像 Amazon EC2 实例或 Amazon RDS 实例一样，您的 RDS Custom 数据库实例位于基于 Amazon VPC 服务的虚拟私有云 (VPC) 中。您将提供并配置自己的 VPC。与 RDS Custom for SQL Server 不同，RDS Custom for Oracle 不创建访问控制列表或安全组。您必须附加自己的安全组、子网和路由表。

您可以使用 CloudFormation 或手动流程配置您的虚拟私有云（VPC）。

> **Important**
>
> 强烈建议您使用 Amazon CloudFormation 配置您的 RDS Custom for Oracle 环境。这种技术最简单，最不容易出错。

主题

- [使用 CloudFormation 配置 VPC（建议）]
- [为 RDS Custom for Oracle 手动配置 VPC]

使用 CloudFormation 配置 VPC（建议）

如果您已经为其他 RDS Custom 引擎配置了 VPC 并且想要重用现有的 VPC，请跳过此步骤。此部分假设如下：

- 您已使用 CloudFormation 创建 IAM 实例配置文件和角色。
- 您知道路由表 ID。

  要使数据库实例成为私有实例，该实例必须位于私有子网中。要使子网成为私有子网，该子网不得与具有原定设置互联网网关的路由表关联。有关更多信息，请参阅《Amazon VPC 用户指南》中的配置路由表。

当您对 VPC 使用 CloudFormation 模板时，该模板会创建以下资源：

- 私有 VPC
- 名为 `rds-custom-private` 的子网组
- 以下 VPC 端点，您的数据库实例使用这些端点与从属 Amazon Web Services 服务进行通信：
  - `com.amazonaws.`*`region`*`.ec2messages`
  - `com.amazonaws.`*`region`*`.events`
  - `com.amazonaws.`*`region`*`.logs`
  - `com.amazonaws.`*`region`*`.monitoring`
  - `com.amazonaws.`*`region`*`.s3`
  - `com.amazonaws.`*`region`*`.secretsmanager`
  - `com.amazonaws.`*`region`*`.ssm`
  - `com.amazonaws.`*`region`*`.ssmmessages`

> **Note**
>
> 对于使用现有账户的复杂网络设置，如果访问权限尚不存在，建议您手动配置对从属服务的访问权限。有关更多信息，请参阅 确保您的 VPC 可以访问从属 Amazon Web Services 服务。

要采用 CloudFormation 配置 VPC

1. 通过以下网址打开 CloudFormation 控制台：https://console.aws.amazon.com/cloudformation。

2. 启动“创建堆栈”向导，选择创建堆栈，然后选择使用新资源（标准）。
3. 在 Create stack（创建堆栈）页面上，执行以下操作：

    a. 对于 Prepare template（准备模板），选择 Template is ready（模板就绪）。
    b. 在 Template source（模板来源）项下，选择 Upload a template file（上载模板文件）。
    c. 对于 Choose file（选择文件），请导航到 `custom-vpc.json` 并选择。
    d. 选择下一步。
4. 在 Specify stack details（指定堆栈详细信息）页面中，请执行以下操作：

    a. 对于堆栈名称，输入 **`custom-vpc`**。
    b. 对于 Parameters（参数），请选择要用于 RDS Custom 数据库实例的私有子网。
    c. 选择要用于 RDS Custom 数据库实例的私有 VPC ID。
    d. 输入与私有子网关联的路由表。
    e. 选择 Next(下一步)。
5. 在 配置堆栈选项 页面上，请选择 下一步。
6. 在审核 custom-vpc 页面上，选择提交。

    CloudFormation 会配置您的私有 VPC。在左侧面板中，当 custom-vpc 显示 CREATE_COMPLETE 时，继续执行下一步。
7. （可选）查看您的 VPC 的详细信息。在堆栈窗格中，选择 custom-vpc。在右窗格中，执行以下操作：

    a. 选择堆栈信息。堆栈的 ID 格式为：arn:aws:cloudformation:***`region`***:***`account-no`***:stack/custom-vpc/***`identifier`***。
    b. 选择资源。您应该看到一个名为 rds-custom-private 的子网组和若干使用命名格式 vpce-***`string`*** 的 VPC 端点。每个端点对应于 RDS Custom 需要与之通信的一项 Amazon Web Services 服务。有关更多信息，请参阅 确保您的 VPC 可以访问从属 Amazon Web Services 服务。
    c. 选择参数。您应该看到私有子网、私有 VPC 和您在创建堆栈时指定的路由表。在创建数据库实例时，您需要提供 VPC ID 和子网组。

为 RDS Custom for Oracle 手动配置 VPC

除了使用 Amazon CloudFormation 自动创建 VPC 之外，您还可以手动配置 VPC。当您具有使用现有资源的复杂网络设置时，此选项可能是最好的。

主题

- [确保您的 VPC 可以访问从属 Amazon Web Services 服务]
- [配置实例元数据服务]

确保您的 VPC 可以访问从属 Amazon Web Services 服务

RDS Custom 将通信从数据库实例发送到其他 Amazon Web Services 服务。确保可以从您在其中创建 RDS Custom 数据库实例的子网访问以下服务：

- Amazon CloudWatch (`com.amazonaws.region.monitoring`)
- Amazon CloudWatch Logs (`com.amazonaws.region.logs`)
- Amazon CloudWatch Events (`com.amazonaws.region.events`)
- Amazon EC2（`com.amazonaws.region.ec2` 和 `com.amazonaws.region.ec2messages`）
- Amazon EventBridge (`com.amazonaws.region.events`)
- Amazon S3 (`com.amazonaws.region.s3`)
- Amazon Secrets Manager (`com.amazonaws.region.secretsmanager`)
- Amazon Systems Manager (`com.amazonaws.region.ssm` 和 `com.amazonaws.region.ssmmessages`)

如果创建多可用区部署

- Amazon Simple Queue Service (`com.amazonaws.region.sqs`)

如果 RDS Custom 无法与必需的服务通信，它会发布以下事件：

```
Database instance in incompatible-network. SSM Agent connection not available. Amazon
 RDS can't connect to the dependent Amazon services.
```

```
Database instance in incompatible-network. Amazon RDS can't connect to dependent Amazon
 services. Make sure port 443 (HTTPS) allows outbound connections, and try again.
 "Failed to connect to the following services: s3 events"
```

为避免 `incompatible-network` 错误，请确保您的 RDS Custom 数据库实例与 Amazon Web Services 服务之间通信所涉及的 VPC 组件满足以下要求：

- 数据库实例可以在端口 443 上建立到其他 Amazon Web Services 服务的出站连接。

- VPC 允许对源自 RDS Custom 数据库实例的请求进行传入响应。
- RDS Custom 可以正确解析每个 Amazon Web Services 服务的端点的域名。

如果您已经为其他 RDS Custom 数据库引擎配置了 VPC，可以重用该 VPC 并跳过此过程。

配置实例元数据服务

请确保您的实例可以执行以下操作：

- 可使用 Instance Metadata Service Version 2 (IMDSv2) 来访问实例元数据服务。
- 允许通过端口 80 (HTTP) 到 IMDS 链接 IP 地址的出站通信。
- 从 `http://169.254.169.254` IMDSv2 链接请求实例元数据。

有关更多信息，请参阅《Amazon EC2 用户指南》中的使用 IMDSv2。

RDS Custom for Oracle 自动化程序预设情况下使用 IMDSv2，方法是在底层 Amazon EC2 实例上设置 `HttpTokens=enabled`。但是，您也可以根据需要使用 IMDSv1。有关更多信息，请参阅《Amazon EC2 用户指南》https://docs.amazonaws.cn/AWSEC2/latest/UserGuide/configuring-instance-metadata-options.html中的配置实例元数据选项。

使用 Amazon RDS Custom for Oracle 的自定义引擎版本

Amazon RDS Custom for Oracle 的一个自定义引擎版本 (CEV)是数据库引擎和特定 Amazon Machine Image (AMI) 的二进制卷快照。默认情况下，RDS Custom for Oracle 使用 RDS Custom 管理的最新可用 AMI，但您可以指定在先前 CEV 中使用的 AMI。您将数据库安装文件存储在 Amazon S3 中。RDS Custom 将使用安装文件和 AMI 为您创建 CEV。

主题

为创建 CEV 做准备

要创建 CEV，请访问存储在 Amazon S3 桶中的以下任意版本的安装文件和补丁：

- Oracle Database 19c
- Oracle Database 18c
- Oracle Database 12c 第 2 版 (12.2)
- Oracle Database 12c 第 1 版 (12.1)

例如，您可以将 2021 年 4 月的 RU/RUR 用于 Oracle Database 19c，也可以使用安装文件和补丁的任何有效组合。有关 RDS Custom for Oracle 支持的版本和区域的更多信息，请参阅适用于 RDS for Oracle 的 RDS Custom。

主题

- [步骤 6（可选）：验证 CEV 清单]
- [步骤 7：添加必要的 IAM 权限]

步骤 1（可选）：下载清单模板

CEV 清单是一个 JSON 文档，其中包含 CEV 的数据库安装 .zip 文件的列表。要创建 CEV，请执行以下操作：

1. 确定要包含在 CEV 中的 Oracle 数据库安装文件。
2. 下载安装文件。
3. 创建列出安装文件的 JSON 清单。

RDS Custom for Oracle 为每个受支持的 Oracle Database 版本提供 JSON 清单模板以及我们推荐的 .zip 文件。例如，以下模板适用于 19.17.0.0.0 RU。

```
{
    "mediaImportTemplateVersion": "2020-08-14",
    "databaseInstallationFileNames": [
        "V982063-01.zip"
    ],
    "opatchFileNames": [
        "p6880880_190000_Linux-x86-64.zip"
    ],
    "psuRuPatchFileNames": [
        "p34419443_190000_Linux-x86-64.zip",
        "p34411846_190000_Linux-x86-64.zip"
    ],
    "otherPatchFileNames": [
        "p28852325_190000_Linux-x86-64.zip",
        "p29997937_190000_Linux-x86-64.zip",
        "p31335037_190000_Linux-x86-64.zip",
        "p32327201_190000_Linux-x86-64.zip",
        "p33613829_190000_Linux-x86-64.zip",
        "p34006614_190000_Linux-x86-64.zip",
        "p34533061_190000_Linux-x86-64.zip",
        "p34533150_190000_Generic.zip",
        "p28730253_190000_Linux-x86-64.zip",
        "p29213893_1917000DBRU_Generic.zip",
        "p33125873_1917000DBRU_Linux-x86-64.zip",
        "p34446152_1917000DBRU_Linux-x86-64.zip"
```

```
    ]
}
```

每个模板都有关联的自述文件，其中包括有关下载补丁、.zip 文件的 URL 和文件校验和的说明。您可以按原样使用这些模板，也可以使用自己的补丁对这些模板进行修改。要查看模板，请将 custom-oracle-manifest.zip 下载到本地磁盘，然后使用文件存档应用程序将其打开。有关更多信息，请参阅 步骤 5：准备 CEV 清单。

步骤 2：从 Oracle 软件传输云下载数据库安装文件和补丁

确定了要用于 CEV 的安装文件后，将这些文件下载到本地系统。Oracle 数据库安装文件和补丁托管在 Oracle 软件传输云上。每个 CEV 都需要一个基础版本 [例如 Oracle Database 19c 或 Oracle Database 12c 第 2 版（12.2）] 以及可选补丁列表。

下载 Oracle Database 的数据库安装文件

1. 请转到 https://edelivery.oracle.com/ 然后登录。
2. 在搜索框中，输入 **`Oracle Database Enterprise Edition`** 或 **`Oracle Database Standard Edition 2`**，然后选择搜索。
3. 请选择以下基础版本之一：

数据库版本	Enterprise Edition	Standard Edition 2
Oracle Database 19c	DLP：Oracle Database 19c Enterprise Edition 19.3.0.0.0（Oracle Database Enterprise Edition）	DLP：Oracle Database 19c Standard Edition 2 19.3.0.0.0（Oracle Database Standard Edition 2）
Oracle Database 18c	DLP：Oracle Database 18c Enterprise Edition 18.0.0.0.0（Oracle Database Enterprise Edition）	DLP：Oracle Database Standard Edition 2 18.0.0.0.0（Oracle Database Standard Edition 2）
Oracle Database 12c 第 2 版 (12.2.0.1)	DLP：Oracle Database 12c Enterprise Edition 12.2.0.1.0（Oracle Database Enterprise Edition）	DLP：Oracle Database Standard Edition 2 12.2.0.1.0（Oracle Database Standard Edition 2）

数据库版本	Enterprise Edition	Standard Edition 2
Oracle Database 12c 第 1 版 (12.1.0.2)	DLP：Oracle Database 12c Enterprise Edition 12.1.0.2.0（Oracle Database Enterprise Edition）	DLP：Oracle Database Standard Edition 2 12.1.0.2.0（Oracle Database Standard Edition 2）

4. 选择继续。
5. 清除 Download Queue（下载队列）复选框。
6. 选择与基础版本相对应的选项：
   - Oracle Database 19.3.0.0.0 - 长期版本。
   - Oracle Database 18.0.0.0.0
   - Oracle Database 12.2.0.1.0。
   - Oracle Database 12.1.0.2.0。
7. 在 Platform/Languages（平台/语言）中选择 Linux x86-64。
8. 选择继续，然后签署 Oracle 许可协议。
9. 选择与您的数据库版本对应的 .zip 文件：

数据库发行版和版本	zip 文件	SHA-256 哈希
19c EE 和 SE2	V982063-01.zip	BA8329C757133DA313ED3B6D7F86C5AC42CD9970A28BF2E6233F3235233AA8D8
18c EE 和 SE2	V978967-01.zip	C96A4FD768787AF98272008833FE10B172691CF84E42816B138C12D4DE63AB96
12.2.0.1 EE 和 SE2	V839960-01.zip	96ED97D21F15C1AC0CCE3749DA6C3DAC7059BB60672D76B008103FC754D22DDE
12.1.0.2 EE	V46095-01_1of2.zip V46095-01_2of2.zip	31FDC2AF41687B4E547A3A18F796424D8C1AF36406D2160F65B0AF6A9CD47355 对应于 V46095-01_1of2.zip

数据库发行版和版本	zip 文件	SHA-256 哈希
		03DA14F5E875304B28F0F3BB02AF0EC33227885B99C9865DF70749D1E220ACCD 对应于 V46095-01_2of2.zip
12.1.0.2 SE2	V77388-01_1of2.zip V77388-01_2of2.zip	73873369753230F5A0921F95ACEADB591388CB06ED72A7F3AEA7BCBCEA2403BC 对应于 V77388-01_1of2.zip 2492E1BE1E3E3531DA83D0843C09C08E435AC8CEFD9A00C0DF56BE4F15CEEBF3 对应于 V77388-01_2of2.zip

10. 将所需的 Oracle 补丁从 `updates.oracle.com` 或 `support.oracle.com` 下载到本地系统。您可以在以下位置找到补丁的 URL：

    - 您在步骤 1（可选）：下载清单模板中下载的 .zip 文件中的自述文件
    - 适用于 Oracle 的 Amazon Relational Database Service（Amazon RDS）发布说明的每个版本更新（RU）中列出的补丁

步骤 3：将安装文件上传到 Amazon S3

使用 Amazon CLI 将您的 Oracle 安装和补丁文件上载到 Amazon S3。包含安装文件的 S3 存储桶必须与您的 CEV 在同一个 Amazon 区域。

本节中的示例使用以下占位符：

- *install-or-patch-file.zip* – Oracle 安装媒体文件。例如，p32126828_190000_Linux-x86-64.zip 是一个补丁。
- *amzn-s3-demo-destination-bucket* – 您为上载的安装文件指定的 Amazon S3 存储桶。
- *123456789012/cev1* – Amazon S3 存储桶中的可选前缀。
- *amzn-s3-demo-source-bucket* – 您可以选择将文件暂存到的 Amazon S3 存储桶。

主题

- 步骤 3a：验证您的 S3 桶是否位于正确的 Amazon Web Services 区域

步骤 3a：验证您的 S3 桶是否位于正确的 Amazon Web Services 区域

验证 S3 存储桶是否位于您计划运行 `create-custom-db-engine-version` 命令所在的 Amazon 区域。

```
aws s3api get-bucket-location --bucket amzn-s3-demo-destination-bucket
```

步骤 3b：确保您的 S3 桶策略具有正确的权限

您可以从头开始或者从源 CEV 创建 CEV。如果您计划从源 CEV 创建新的 CEV，请确保您的 S3 桶策略具有正确的权限：

1. 标识 RDS Custom 预留的 S3 桶。桶名称的格式为 `do-not-delete-rds-custom-account-region-string`。例如，桶名称可能为 `do-not-delete-rds-custom-123456789012-us-east-1-abc123EXAMPLE`。
2. 确保将以下权限附加到您的 S3 桶策略中。将 `do-not-delete-rds-custom-123456789012-us-east-1-abc123EXAMPLE` 替换为您的存储桶的名称。

   ```
   {
       "Sid": "AWSRDSCustomForOracleCustomEngineVersionGetObject",
       "Effect": "Allow",
       "Principal": {
           "Service": "custom.rds.amazonaws.com"
       },
       "Action": [
           "s3:GetObject",
           "s3:GetObjectTagging"
       ],
       "Resource": "arn:aws:s3:::do-not-delete-rds-custom-123456789012-us-
   east-1-abc123EXAMPLE/CustomEngineVersions/*"
   }, ...
   ```

步骤 3c：使用 cp 或同步命令上传文件

请选择以下任一选项：

- 使用 `aws s3 cp` 上载单个 .zip 文件。

  分别上载每个安装 .zip 文件。不要将 .zip 文件合并为单个 .zip 文件。
- 使用 `aws s3 sync` 上传目录。

Example

以下示例将 *install-or-patch-file.zip* 上传到 RDS Custom Amazon S3 存储桶中的 *123456789012/cev1* 文件夹。为您想要上传的每个 .zip 运行单独的 `aws s3` 命令。

对于 Linux、macOS 或 Unix：

```
aws s3 cp install-or-patch-file.zip \
    s3://amzn-s3-demo-destination-bucket/123456789012/cev1/
```

对于 Windows：

```
aws s3 cp install-or-patch-file.zip ^
    s3://amzn-s3-demo-destination-bucket/123456789012/cev1/
```

Example

以下示例将您本地 *cev1* 文件中的文件上传到您的 Amazon S3 存储桶中的 *123456789012/cev1* 文件夹。

对于 Linux、macOS 或 Unix：

```
aws s3 sync cev1 \
    s3://amzn-s3-demo-destination-bucket/123456789012/cev1/
```

对于 Windows：

```
aws s3 sync cev1 ^
    s3://amzn-s3-demo-destination-bucket/123456789012/cev1/
```

Example

以下示例将 *amzn-s3-demo-source-bucket* 中的所有文件上传到您的 Amazon S3 存储桶中的 *123456789012/cev1* 文件夹。

对于 Linux、macOS 或 Unix：

```
aws s3 sync s3://amzn-s3-demo-source-bucket/ \
    s3://amzn-s3-demo-destination-bucket/123456789012/cev1/
```

对于 Windows：

```
aws s3 sync s3://amzn-s3-demo-source-bucket/ ^
    s3://amzn-s3-demo-destination-bucket/123456789012/cev1/
```

步骤 3d：列出 S3 桶中的文件

以下示例使用 `s3 ls` 命令列出 RDS Custom Amazon S3 桶中的文件。

```
aws s3 ls \
    s3://amzn-s3-demo-destination-bucket/123456789012/cev1/
```

步骤 4（可选）：在 S3 中跨 Amazon Web Services 账户共享您的安装媒体

就本部分的目的而言，包含您上载的 Oracle 安装文件的 Amazon S3 桶是您的媒体桶。您的组织可能会在 Amazon Web Services 区域中使用多个 Amazon Web Services 账户。如果是这样，您可能需要使用一个 Amazon Web Services 账户来填充媒体桶，并使用另一个 Amazon Web Services 账户来创建 CEV。如果您不打算共享媒体桶，请跳至下一部分。

此部分假设如下：

- 您可以访问创建媒体桶的账户和您打算在其中创建 CEV 的另一个账户。
- 您打算只在一个 Amazon Web Services 区域中创建 CEV。如果您打算使用多个区域，请在每个区域中创建一个媒体桶。
- 您使用的是 CLI。如果您使用 Amazon S3 控制台，请调整以下步骤。

将您的媒体桶配置为跨 Amazon Web Services 账户进行共享

1. 登录到 Amazon Web Services 账户，其中包含您上载了安装媒体的 S3 桶。
2. 从空白 JSON 策略模板或您可以调整的现有策略开始。

   以下命令检索现有策略并将其另存为 *my-policy.json*。在此示例中，包含安装文件的 S3 存储桶命名为 *amzn-s3-demo-bucket*。

```
aws s3api get-bucket-policy \
    --bucket amzn-s3-demo-bucket \
    --query Policy \
    --output text > my-policy.json
```

3. 按如下方式编辑媒体桶权限：

   - 在模板的 `Resource` 元素中，指定您将 Oracle 数据库安装文件上载到的 S3 桶。
   - 在 `Principal` 元素中，指定您打算用于创建 CEV 的所有 Amazon Web Services 账户的 ARN。您可以将根、用户或角色添加到 S3 桶允许列表中。有关更多信息，请参阅 Amazon Identity and Access Management 用户指南 中的 [IAM 标识符](#)。

```
{
    "Version": "2008-10-17",
    "Statement": [
        {
            "Sid": "GrantAccountsAccess",
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::account-1:root",
                    "arn:aws:iam::account-2:user/user-name-with-path",
                    "arn:aws:iam::account-3:role/role-name-with-path",
                    ...
                ]
            },
            "Action": [
                "s3:GetObject",
                "s3:GetObjectAcl",
                "s3:GetObjectTagging",
                "s3:ListBucket",
                "s3:GetBucketLocation"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        }
    ]
}
```

4. 将此策略附加到您的媒体桶。

   在以下示例中，*amzn-s3-demo-bucket* 是包含安装文件的 S3 存储桶的名称，而 *my-policy.json* 是 JSON 文件的名称。

   ```
   aws s3api put-bucket-policy \
       --bucket amzn-s3-demo-bucket \
       --policy file://my-policy.json
   ```

5. 登录到您打算在其中创建 CEV 的 Amazon Web Services 账户。
6. 验证此账户是否可以访问 Amazon Web Services 账户（此账户创建了媒体桶）中的媒体桶。

   ```
   aws s3 ls --query "Buckets[].Name"
   ```

   有关更多信息，请参阅《Amazon CLI 命令参考》中的 aws s3 ls。
7. 请按照创建 CEV中的步骤创建 CEV。

步骤 5：准备 CEV 清单

CEV 清单是包含以下内容的 JSON 文档：

- （必需）您上载到 Amazon S3 的安装 .zip 文件的列表。RDS Custom 将按它们在清单中列出的顺序应用补丁。
- （可选）为 Oracle 根目录、Oracle 主目录以及 UNIX/Linux 用户和组的 ID 和名称设置非原定设置值的安装参数。请注意，您无法修改现有 CEV 或现有数据库实例的安装参数。当安装参数具有不同的设置时，您也无法从一个 CEV 升级到另一个 CEV。

有关示例 CEV 清单，请参阅您在步骤 1（可选）：下载清单模板中下载的 JSON 模板。此外，您还可以在CEV 清单示例中查看示例。

主题

- CEV 清单中的 JSON 字段
- 创建 CEV 清单
- CEV 清单示例

CEV 清单中的 JSON 字段

下表介绍了清单中的 JSON 字段。

JSON 字段	描述
`MediaImportTemplateVersion`	CEV 清单的版本。日期的格式为 `YYYY-MM-DD` 。
`databaseInstallationFileNames`	数据库安装文件的有序列表。
`opatchFileNames`	用于 Oracle 数据库引擎的 OPatch 安装程序的有序列表。只有一个值有效。`opatchFileNames` 的值必须以 `p6880880_` 开头。
`psuRuPatchFileNames`	此数据库的 PSU 和 RU 补丁程序。 **Important** 如果包含 `psuRuPatchFileNames` ，则 `opatchFileNames` 为必需的。`opatchFileNames` 的值必须以 `p6880880_` 开头。
`OtherPatchFileNames`	不在 PSU 和 RU 补丁列表中的补丁。RDS Custom 在应用 PSU 和 RU 补丁后应用这些补丁。 **Important** 如果包含 `OtherPatchFileNames` ，则 `opatchFileNames` 为必需的。`opatchFileNames` 的值必须以 `p6880880_` 开头。
`installationParameters`	Oracle 根目录、Oracle 主目录以及 UNIX/Linux 用户和组的 ID 和名称的非原定设置。您可以设置以下参数：

JSON 字段	描述
	oracleBase 在其中安装 Oracle 二进制文件的目录。它是存储文件的二进制卷的挂载点。Oracle 根目录可以包含多个 Oracle 主目录。例如，如果 `/home/oracle/oracle.19.0.0.0.ru-2020-04.rur-2020-04.r1.EE.1` 是您的 Oracle 主目录之一，则 `/home/oracle` 是 Oracle 根目录。用户指定的 Oracle 根目录不是符号链接。 如果您未指定 Oracle 根目录，则原定设置目录为 `/rdsdbbin`。 oracleHome 在其中安装 Oracle 数据库二进制文件的目录。例如，如果您将 `/home/oracle/` 指定为 Oracle 根目录，则可以将 `/home/oracle/oracle.19.0.0.0.ru-2020-04.rur-2020-04.r1.EE.1/` 指定为 Oracle 主目录。用户指定的 Oracle 主目录不是符号链接。Oracle 主目录值由 `$ORACLE_HOME` 环境变量引用。 如果未指定 Oracle 主目录，则原定设置命名格式为 `/rdsdbbin/oracle.`*major-engine-version*`.custom.r1.`*engine-edition*`.1`。 unixUname 拥有 Oracle 软件的 UNIX 用户的名称。RDS Custom 在运行本地数据库命令时使用此用户。如果您同时指定 `unixUid` 和 `unixUname`，则 RDS Custom 会在用户不存在时创建该用户，然后在该 UID 与初始 UID 不同时将其分配给该用户。 原定设置用户名为 `rdsdb`。 unixUid 拥有 Oracle 软件的 UNIX 用户的 ID（UID）。如果您同时指定 `unixUid` 和 `unixUname`，则 RDS Custom 会在用户不存在

JSON 字段	描述
	时创建该用户，然后在该 UID 与初始 UID 不同时将其分配给该用户。 原定设置 UID 为 `61001`。这是用户 `rdsdb` 的 UID。 unixGroupName UNIX 组的名称。拥有 Oracle 软件的 UNIX 用户属于该组。 原定设置组名称为 `rdsdb`。 unixGroupId UNIX 用户所属的 UNIX 组的 ID。 原定设置组 ID 为 `1000`。这是组 `rdsdb` 的 ID。

每个 Oracle 数据库版本都有不同的受支持安装文件的列表。在创建 CEV 清单时，请确保仅指定 RDS Custom for Oracle 支持的文件。否则，CEV 创建将失败并出现错误。支持适用于 Oracle 的 Amazon Relational Database Service（Amazon RDS）发布说明中列出的所有补丁。

创建 CEV 清单

创建 CEV 清单

1. 按您计划应用安装文件的顺序，列出您计划应用的所有安装文件。
2. 将安装文件与CEV 清单中的 JSON 字段中描述的 JSON 字段关联起来。
3. 请执行以下任一操作：
   - 将 CEV 清单创建为 JSON 文本文件。
   - 在控制台中创建 CEV 时编辑 CEV 清单模板。有关更多信息，请参阅 创建 CEV。

CEV 清单示例

以下示例显示了不同 Oracle Database 版本的 CEV 清单文件。如果您在清单中包含 JSON 字段，请确保该字段不为空。例如，以下 CEV 清单无效，因为 `otherPatchFileNames` 为空。

```
{
    "mediaImportTemplateVersion": "2020-08-14",
```

```
    "databaseInstallationFileNames": [
        "V982063-01.zip"
    ],
    "opatchFileNames": [
        "p6880880_190000_Linux-x86-64.zip"
    ],
    "psuRuPatchFileNames": [
        "p32126828_190000_Linux-x86-64.zip"
    ],
    "otherPatchFileNames": [
    ]
}
```

主题

- Sample CEV manifest for Oracle Database 12c Release 1 (12.1)
- Sample CEV manifest for Oracle Database 12c Release 2 (12.2)
- Sample CEV manifest for Oracle Database 18c
- Sample CEV manifest for Oracle Database 19c

Example Oracle Database 12c 第 1 版（12.1）的 CEV 清单示例

在以下示例中，对用于 Oracle Database 12c 第 1 版（12.1）的 2021 年 7 月 PSU，RDS Custom 将按指定的顺序应用补丁。因此，RDS Custom 依次应用 p32768233、p32876425、p18759211，以此类推。该示例为 UNIX 用户和组以及 Oracle 主目录和 Oracle 根目录设置新值。

```
{
    "mediaImportTemplateVersion":"2020-08-14",
    "databaseInstallationFileNames":[
        "V46095-01_1of2.zip",
        "V46095-01_2of2.zip"
    ],
    "opatchFileNames":[
        "p6880880_121010_Linux-x86-64.zip"
    ],
    "psuRuPatchFileNames":[
        "p32768233_121020_Linux-x86-64.zip"
    ],
    "otherPatchFileNames":[
        "p32876425_121020_Linux-x86-64.zip",
        "p18759211_121020_Linux-x86-64.zip",
```

```
        "p19396455_121020_Linux-x86-64.zip",
        "p20875898_121020_Linux-x86-64.zip",
        "p22037014_121020_Linux-x86-64.zip",
        "p22873635_121020_Linux-x86-64.zip",
        "p23614158_121020_Linux-x86-64.zip",
        "p24701840_121020_Linux-x86-64.zip",
        "p25881255_121020_Linux-x86-64.zip",
        "p27015449_121020_Linux-x86-64.zip",
        "p28125601_121020_Linux-x86-64.zip",
        "p28852325_121020_Linux-x86-64.zip",
        "p29997937_121020_Linux-x86-64.zip",
        "p31335037_121020_Linux-x86-64.zip",
        "p32327201_121020_Linux-x86-64.zip",
        "p32327208_121020_Generic.zip",
        "p17969866_12102210119_Linux-x86-64.zip",
        "p20394750_12102210119_Linux-x86-64.zip",
        "p24835919_121020_Linux-x86-64.zip",
        "p23262847_12102201020_Linux-x86-64.zip",
        "p21171382_12102201020_Generic.zip",
        "p21091901_12102210720_Linux-x86-64.zip",
        "p33013352_12102210720_Linux-x86-64.zip",
        "p25031502_12102210720_Linux-x86-64.zip",
        "p23711335_12102191015_Generic.zip",
        "p19504946_121020_Linux-x86-64.zip"
    ],
    "installationParameters": {
        "unixGroupName": "dba",
        "unixGroupId": 12345,
        "unixUname": "oracle",
        "unixUid": 12345,
        "oracleHome": "/home/oracle/oracle.12.1.0.2",
        "oracleBase": "/home/oracle"
    }
}
```

Example Oracle Database 12c 第 2 版（12.2）的 CEV 清单示例

在以下示例中，对用于 Oracle Database 12c 第 2 版（12.2）的 2021 年 10 月 PSU，RDS Custom 将依次应用 p33261817、p33192662、p29213893，以此类推。该示例为 UNIX 用户和组以及 Oracle 主目录和 Oracle 根目录设置新值。

```
{
    "mediaImportTemplateVersion":"2020-08-14",
```

```
    "databaseInstallationFileNames":[
        "V839960-01.zip"
    ],
    "opatchFileNames":[
        "p6880880_122010_Linux-x86-64.zip"
    ],
    "psuRuPatchFileNames":[
        "p33261817_122010_Linux-x86-64.zip"
    ],
    "otherPatchFileNames":[
        "p33192662_122010_Linux-x86-64.zip",
        "p29213893_122010_Generic.zip",
        "p28730253_122010_Linux-x86-64.zip",
        "p26352615_12201211019DBOCT2021RU_Linux-x86-64.zip",
        "p23614158_122010_Linux-x86-64.zip",
        "p24701840_122010_Linux-x86-64.zip",
        "p25173124_122010_Linux-x86-64.zip",
        "p25881255_122010_Linux-x86-64.zip",
        "p27015449_122010_Linux-x86-64.zip",
        "p28125601_122010_Linux-x86-64.zip",
        "p28852325_122010_Linux-x86-64.zip",
        "p29997937_122010_Linux-x86-64.zip",
        "p31335037_122010_Linux-x86-64.zip",
        "p32327201_122010_Linux-x86-64.zip",
        "p32327208_122010_Generic.zip"
    ],
    "installationParameters": {
        "unixGroupName": "dba",
        "unixGroupId": 12345,
        "unixUname": "oracle",
        "unixUid": 12345,
        "oracleHome": "/home/oracle/oracle.12.2.0.1",
        "oracleBase": "/home/oracle"
    }
}
```

Example Oracle Database 18c 的 CEV 清单示例

在以下示例中，对用于 Oracle Database 18c 的 2021 年 10 月 PSU，RDS Custom 将依次应用 p32126855、p28730253、p27539475，以此类推。该示例为 UNIX 用户和组以及 Oracle 主目录和 Oracle 根目录设置新值。

```
{
```

```
    "mediaImportTemplateVersion":"2020-08-14",
    "databaseInstallationFileNames":[
        "V978967-01.zip"
    ],
    "opatchFileNames":[
        "p6880880_180000_Linux-x86-64.zip"
    ],
    "psuRuPatchFileNames":[
        "p32126855_180000_Linux-x86-64.zip"
    ],
    "otherPatchFileNames":[
        "p28730253_180000_Linux-x86-64.zip",
        "p27539475_1813000DBRU_Linux-x86-64.zip",
        "p29213893_180000_Generic.zip",
        "p29374604_1813000DBRU_Linux-x86-64.zip",
        "p29782284_180000_Generic.zip",
        "p28125601_180000_Linux-x86-64.zip",
        "p28852325_180000_Linux-x86-64.zip",
        "p29997937_180000_Linux-x86-64.zip",
        "p31335037_180000_Linux-x86-64.zip",
        "p31335142_180000_Generic.zip"
    ]
    "installationParameters": {
        "unixGroupName": "dba",
        "unixGroupId": 12345,
        "unixUname": "oracle",
        "unixUid": 12345,
        "oracleHome": "/home/oracle/18.0.0.0.ru-2020-10.rur-2020-10.r1",
        "oracleBase": "/home/oracle/"
    }
}
```

Example Oracle Database 19c 的 CEV 清单示例

在以下示例中，对 Oracle Database 19c，RDS Custom 依次应用 p32126828、p29213893、p29782284，以此类推。该示例为 UNIX 用户和组以及 Oracle 主目录和 Oracle 根目录设置新值。

```
{
    "mediaImportTemplateVersion": "2020-08-14",
    "databaseInstallationFileNames": [
        "V982063-01.zip"
    ],
```

```
    "opatchFileNames": [
        "p6880880_190000_Linux-x86-64.zip"
    ],
    "psuRuPatchFileNames": [
        "p32126828_190000_Linux-x86-64.zip"
    ],
    "otherPatchFileNames": [
        "p29213893_1910000DBRU_Generic.zip",
        "p29782284_1910000DBRU_Generic.zip",
        "p28730253_190000_Linux-x86-64.zip",
        "p29374604_1910000DBRU_Linux-x86-64.zip",
        "p28852325_190000_Linux-x86-64.zip",
        "p29997937_190000_Linux-x86-64.zip",
        "p31335037_190000_Linux-x86-64.zip",
        "p31335142_190000_Generic.zip"
    ],
    "installationParameters": {
        "unixGroupName": "dba",
        "unixGroupId": 12345,
        "unixUname": "oracle",
        "unixUid": 12345,
        "oracleHome": "/home/oracle/oracle.19.0.0.0.ru-2020-04.rur-2020-04.r1.EE.1",
        "oracleBase": "/home/oracle"
    }
}
```

步骤 6（可选）：验证 CEV 清单

或者，通过运行 `json.tool` Python 脚本验证清单是有效的 JSON 文件。例如，如果您更改到包含名为 `manifest.json` 的 CEV 清单的目录，请运行以下命令。

```
python -m json.tool < manifest.json
```

步骤 7：添加必要的 IAM 权限

确保创建 CEV 的 IAM 主体具有中所述的必要策略 [步骤 5：为您的 IAM 用户或角色授予所需的权限](#)。

创建 CEV

您可以使用 Amazon Web Services Management Console 或 Amazon CLI 创建 CEV。指定多租户或非多租户架构。有关更多信息，请参阅 [多租户架构注意事项](#)。

通常，创建 CEV 需要大约两个小时。创建 CEV 后，您可以使用它来创建或更新 RDS Custom 数据库实例。有关更多信息，请参阅创建 RDS Custom for Oracle 数据库实例 和升级 RDS Custom for Oracle 数据库实例。

> **Note**
> 如果您的数据库实例当前使用 Oracle Linux 7.9，请创建一个使用最新 AMI（使用 Oracle Linux 8）的新 CEV。然后修改该实例以使用新的 CEV。

请注意创建 CEV 的以下要求和限制：

- 包含安装文件的 Amazon S3 存储桶必须与您的 CEV 在同一个 Amazon Web Services 区域。否则，创建过程将失败。
- CEV 名称的格式必须为 `major-engine-version.customized_string`，如 `19.cdb_cev1` 所示。
- CEV 名称必须包含 1–50 个字母数字字符、下划线、短划线或句点。
- CEV 名称不能包含连续的句点，如 `19..cdb_cev1` 所示。

控制台

要创建 CEV

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Custom engine versions（自定义引擎版本）。

   Custom engine versions（自定义引擎版本）页面显示了当前存在的所有 CEV。如果您尚未创建任何 CEV，则该页面为空。
3. 选择 Create custom engine version（创建自定义引擎版本）。
4. 在引擎选项中，执行以下操作：

   a. 对于 Engine type（引擎类型），选择 Oracle。
   b. 对于架构设置，（可选）选择多租户架构来创建使用引擎 `custom-oracle-ee-cdb` 或 `custom-oracle-se2-cdb` 的 Oracle 多租户 CEV。您只能使用多租户 CEV 创建 RDS Custom for Oracle CDB。如果您不选择此选项，CEV 将为非 CDB，它使用引擎 `custom-oracle-ee` 或 `custom-oracle-se2`。

> **Note**
>
> 您选择的架构是 CEV 的永久特征。您以后无法修改您的 CEV 以使用其他架构。

c. 请选择以下任一选项：

- 创建新的 CEV – 从头开始创建 CEV。在这种情况下，必须指定 JSON 清单，其中指定了数据库二进制文件。
- 从源创建 CEV – 在指定要复制的 CEV 中，选择要用作源 CEV 的现有 CEV。在这种情况下，您可以指定新的亚马逊机器映像（AMI），但不能指定不同的数据库二进制文件。

d. 对于引擎版本，选择主要引擎版本。

5. 在版本详细信息中，执行以下操作：

a. 在自定义引擎版本名称中输入有效的名称。例如，您可以输入名称 **`19.cdb_cev1`**。

b. （可选）为 CEV 输入描述。

6. 在安装媒体中，执行以下操作：

a. （可选）对于 AMI ID，将该字段留空，以使用提供最新服务的 AMI，或者输入您之前用于创建 CEV 的 AMI。要获取有效的 AMI ID，请使用下面的任一种方法：

- 在控制台中，选择左侧导航窗格中的自定义引擎版本，然后选择 CEV 的名称。CEV 使用的 AMI ID 显示在配置选项卡中。
- 在 Amazon CLI 中，使用 `describe-db-engine-versions` 命令。在输出中搜索 `ImageID`。

b. 对于 S3 location of manifest files（清单文件的 S3 位置），输入您在 [步骤 3：将安装文件上传到 Amazon S3]() 中指定的 Amazon S3 存储桶的位置。例如，输入 **`s3://my-custom-installation-files/123456789012/cev1/`**。

> **Note**
>
> 您创建 CEV 的 Amazon Web Services 区域必须与 S3 桶位于相同的区域。

c. （仅限创建新 CEV）对于 CEV 清单，请输入您在[创建 CEV 清单]()中创建的 JSON 清单。

7. 在 KMS 密钥部分，选择输入密钥 ARN以列出可用的 Amazon KMS 密钥。然后从列表中选择您的 KMS 密钥。

RDS Custom 需要 Amazon KMS 密钥。有关更多信息，请参阅 步骤 1：创建或重用对称加密 Amazon KMS 密钥。

8. （可选）选择添加新标签以便为您的 CEV 创建键值对。
9. 选择 Create custom engine version（创建自定义引擎版本）。

   如果 JSON 清单格式无效，则控制台将显示验证 CEV 清单时出错。修复问题并重试。

Custom engine versions（自定义引擎版本）页面显示。您的 CEV 显示状态 Creating（正在创建）。创建 CEV 的过程需要大约两个小时。

Amazon CLI

要使用 Amazon CLI 创建 CEV，请运行 create-custom-db-engine-version 命令。

以下选项为必填：

- `--engine` – 指定引擎类型。对于 CDB，请指定 `custom-oracle-ee-cdb` 或 `custom-oracle-se2-cdb`。对于非 CDB，请指定 `custom-oracle-ee` 或 `custom-oracle-se2`。您只能从使用 `custom-oracle-ee-cdb` 或 `custom-oracle-se2-cdb` 创建的 CEV 创建 CDB。您只能从使用 `custom-oracle-ee` 或 `custom-oracle-se2` 创建的 CEV 创建非 CDB。
- `--engine-version` – 指定引擎版本。格式为 *major-engine-version.customized_string*。CEV 名称必须包含 1–50 个字母数字字符、下划线、短划线或句点。CEV 名称不能包含连续的句点，如 `19..cdb_cev1` 所示。
- `--kms-key-id` – 指定一个 Amazon KMS key。
- `--manifest` – 指定 *manifest_json_string* 或 `--manifest file:`*file_name*。*manifest_json_string* 中不允许使用换行符。确保在 JSON 代码中转义双引号 (")，方法是使用反斜杠 (\) 作为其前缀。

  以下示例展示来自 步骤 5：准备 CEV 清单 的 19c 的 *manifest_json_string*。该示例为 Oracle 根目录、Oracle 主目录以及 UNIX/Linux 用户和组的 ID 和名称设置了新值。如果您复制此字符串，请先移除所有换行符，然后再将其粘贴到命令中。

```
"{\"mediaImportTemplateVersion\": \"2020-08-14\",
\"databaseInstallationFileNames\": [\"V982063-01.zip\"],
\"opatchFileNames\": [\"p6880880_190000_Linux-x86-64.zip\"],
\"psuRuPatchFileNames\": [\"p32126828_190000_Linux-x86-64.zip\"],
\"otherPatchFileNames\": [\"p29213893_1910000DBRU_Generic.zip\",
```

\"p29782284_1910000DBRU_Generic.zip\",\"p28730253_190000_Linux-x86-64.zip\",\"p29374604_1910000DBRU_Linux-x86-64.zip\",\"p28852325_190000_Linux-x86-64.zip\",\"p29997937_190000_Linux-x86-64.zip\",\"p31335037_190000_Linux-x86-64.zip\",\"p31335142_190000_Generic.zip\"]\"installationParameters\":{ \"unixGroupName\":\"dba\", \ \"unixUname\":\"oracle\", \ \"oracleHome\":\"/home/oracle/oracle.19.0.0.0.ru-2020-04.rur-2020-04.r1.EE.1\", \ \"oracleBase\":\"/home/oracle/\"}}"

- `--database-installation-files-s3-bucket-name` – 指定您在[步骤 3：将安装文件上传到 Amazon S3](#)中指定的桶名称。您运行 `create-custom-db-engine-version` 所在的 Amazon Web Services 区域必须与 Amazon S3 桶位于相同的区域。

您还可以指定以下选项：

- `--description` – 指定您的 CEV 的描述。
- `--database-installation-files-s3-prefix` – 指定您在[步骤 3：将安装文件上传到 Amazon S3](#)中指定的文件夹名称。
- `--image-id` – 指定要重复使用的 AMI ID。要查找有效的 ID，请运行 `describe-db-engine-versions` 命令，然后在输出中搜索 `ImageID`。默认情况下，RDS Custom for Oracle 使用最新的可用 AMI。

下面的示例创建一个名为 `19.cdb_cev1` 的 Oracle 多租户 CEV。该示例重用现有 AMI，而不是使用最新的可用 AMI。确保 CEV 的名称以主要引擎版本号开头。

Example

对于 Linux、macOS 或 Unix：

```
aws rds create-custom-db-engine-version \
    --engine custom-oracle-se2-cdb \
    --engine-version 19.cdb_cev1 \
    --database-installation-files-s3-bucket-name us-east-1-123456789012-custom-installation-files \
    --database-installation-files-s3-prefix 123456789012/cev1 \
    --kms-key-id my-kms-key \
    --description "test cev" \
    --manifest manifest_string \
    --image-id ami-012a345678901bcde
```

对于 Windows：

```
aws rds create-custom-db-engine-version ^
    --engine custom-oracle-se2-cdb ^
    --engine-version 19.cdb_cev1 ^
    --database-installation-files-s3-bucket-name us-east-1-123456789012-custom-installation-files ^
    --database-installation-files-s3-prefix 123456789012/cev1 ^
    --kms-key-id my-kms-key ^
    --description "test cev" ^
    --manifest manifest_string ^
    --image-id ami-012a345678901bcde
```

Example

使用 `describe-db-engine-versions` 命令以获取有关您的 CEV 的详细信息。

```
aws rds describe-db-engine-versions \
    --engine custom-oracle-se2-cdb \
    --include-all
```

以下不完整的示例输出显示了引擎、参数组、清单和其他信息。

```
{
    "DBEngineVersions": [
        {
            "Engine": "custom-oracle-se2-cdb",
            "EngineVersion": "19.cdb_cev1",
            "DBParameterGroupFamily": "custom-oracle-se2-cdb-19",
            "DBEngineDescription": "Containerized Database for Oracle Custom SE2",
            "DBEngineVersionDescription": "test cev",
            "Image": {
                "ImageId": "ami-012a345678901bcde",
                "Status": "active"
            },
            "ValidUpgradeTarget": [],
            "SupportsLogExportsToCloudwatchLogs": false,
            "SupportsReadReplica": true,
            "SupportedFeatureNames": [],
            "Status": "available",
            "SupportsParallelQuery": false,
            "SupportsGlobalDatabases": false,
            "MajorEngineVersion": "19",
```

```
            "DatabaseInstallationFilesS3BucketName": "us-east-1-123456789012-custom-installation-files",
            "DatabaseInstallationFilesS3Prefix": "123456789012/cev1",
            "DBEngineVersionArn": "arn:aws:rds:us-east-1:123456789012:cev:custom-oracle-se2-cdb/19.cdb_cev1/abcd12e3-4f5g-67h8-i9j0-k1234l56m789",
            "KMSKeyId": "arn:aws:kms:us-east-1:732027699161:key/1ab2345c-6d78-9ef0-1gh2-3456i7j89k01",
            "CreateTime": "2023-03-07T19:47:58.131000+00:00",
            "TagList": [],
            "SupportsBabelfish": false,
...
```

创建 CEV 失败

如果创建 CEV 的过程失败，RDS Custom 将发布带有消息 `Creation failed for custom engine version` *`major-engine-version.cev_name`* 的 RDS-EVENT-0198，并包含有关失败的详细信息。例如，事件会打印丢失的文件。

您无法修改失败的 CEV。您只能删除它，然后在修复失败的原因之后再次尝试创建 CEV。有关排查 CEV 创建失败原因的信息，请参阅为 RDS Custom for Oracle 排除自定义引擎版本创建的故障。

修改 CEV 状态

您可以使用 Amazon Web Services Management Console 或 Amazon CLI 修改 CEV。您可以修改 CEV 描述或其可用性状态。您的 CEV 具有以下状态值之一：

- `available` – 您可以使用此 CEV 创建新的 RDS Custom 数据库实例或升级数据库实例。这是新创建的 CEV 的原定设置状态。
- `inactive` – 您无法使用此 CEV 创建或升级 RDS Custom 实例。使用此 CEV，您无法还原数据库快照以创建新的 RDS Custom 数据库实例。

您可以将 CEV 从任何受支持状态更改为任何其他受支持状态。您可以更改状态以防止意外使用 CEV 或使已停止的 CEV 有资格再次使用。例如，您可以将 CEV 的状态从 `available` 更改为 `inactive`，并从 `inactive` 更改回 `available`。

控制台

要修改 CEV

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。

2. 在导航窗格中，选择 Custom engine versions（自定义引擎版本）。
3. 选择要修改其描述或状态的 CEV。
4. 对于 Actions（操作），选择 Modify（修改）。
5. 进行以下任何更改：
   - 对于 CEV status settings（CEV 状态设置），选择新的可用状态。
   - 对于 Version description（版本描述），输入新的描述。
6. 选择 Modify CEV（修改 CEV）。

   如果 CEV 正在使用中，控制台将显示 You can't modify the CEV status（您不能修改 CEV 状态）。修复问题并重试。

Custom engine versions（自定义引擎版本）页面显示。

Amazon CLI

要通过使用 Amazon CLI 修改 CEV，请运行 modify-custom-db-engine-version 命令。您可以通过运行 describe-db-engine-versions 命令查找要修改的 CEV。

以下选项为必填：

- `--engine` *`engine-type`*，其中 *`engine-type`* 为 `custom-oracle-ee`、`custom-oracle-se2`、`custom-oracle-ee-cdb` 或 `custom-oracle-se2-cdb`
- `--engine-version` *`cev`*，其中 *`cev`* 是要修改的自定义引擎版本的名称
- `--status` *`status`*，其中 *`status`* 是您希望分配给 CEV 的可用状态。

以下示例将名为 `19.my_cev1` 的 CEV 从当前状态更改为 `inactive`。

Example

对于 Linux、macOS 或 Unix：

```
aws rds modify-custom-db-engine-version \
    --engine custom-oracle-se2 \
    --engine-version 19.my_cev1 \
    --status inactive
```

对于 Windows：

```
aws rds modify-custom-db-engine-version ^
    --engine custom-oracle-se2 ^
    --engine-version 19.my_cev1 ^
    --status inactive
```

查看 Amazon RDS Custom for Oracle 的 CEV 详细信息

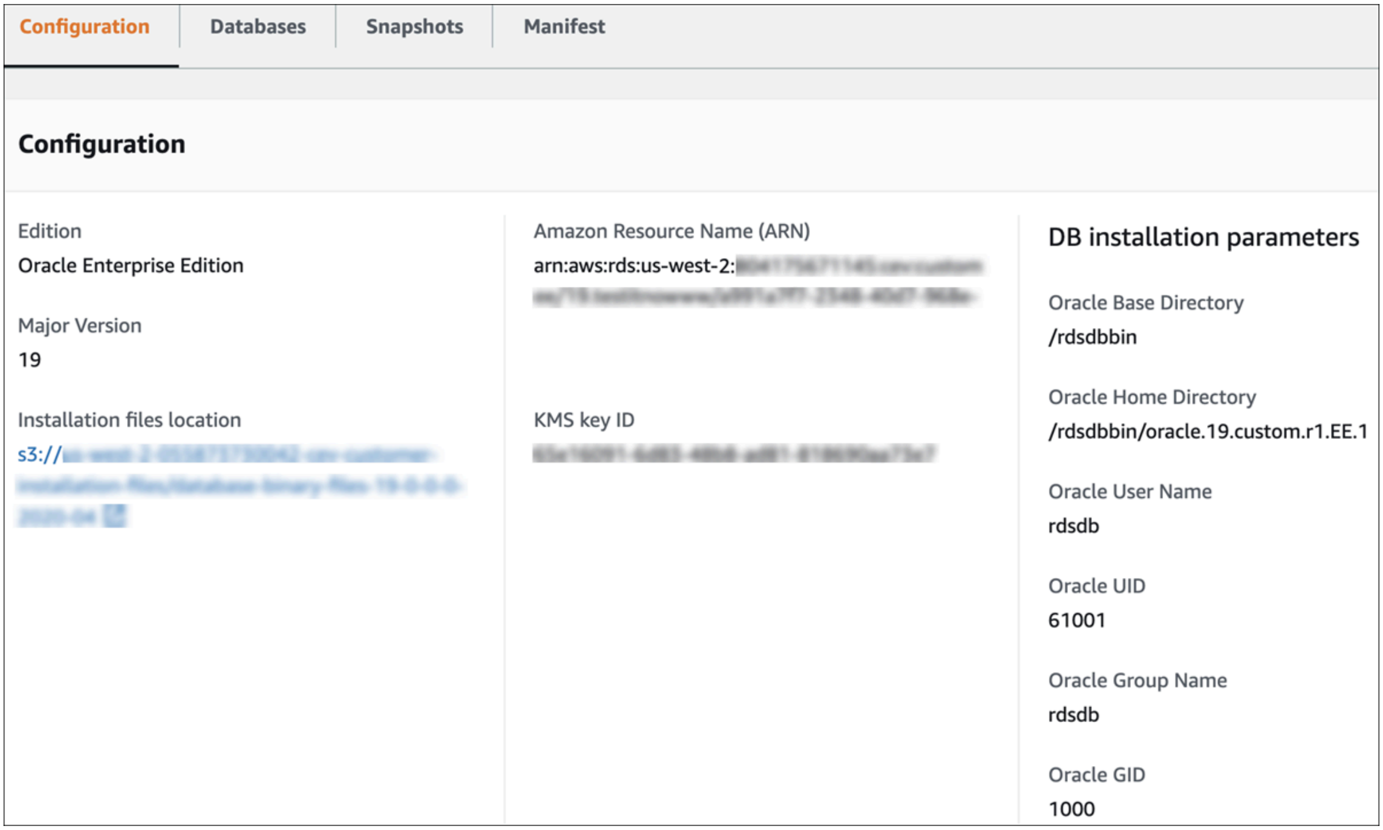
您可以使用 Amazon Web Services Management Console或 Amazon CLI 查看有关 CEV 清单和用于创建 CEV 的命令的详细信息。

控制台

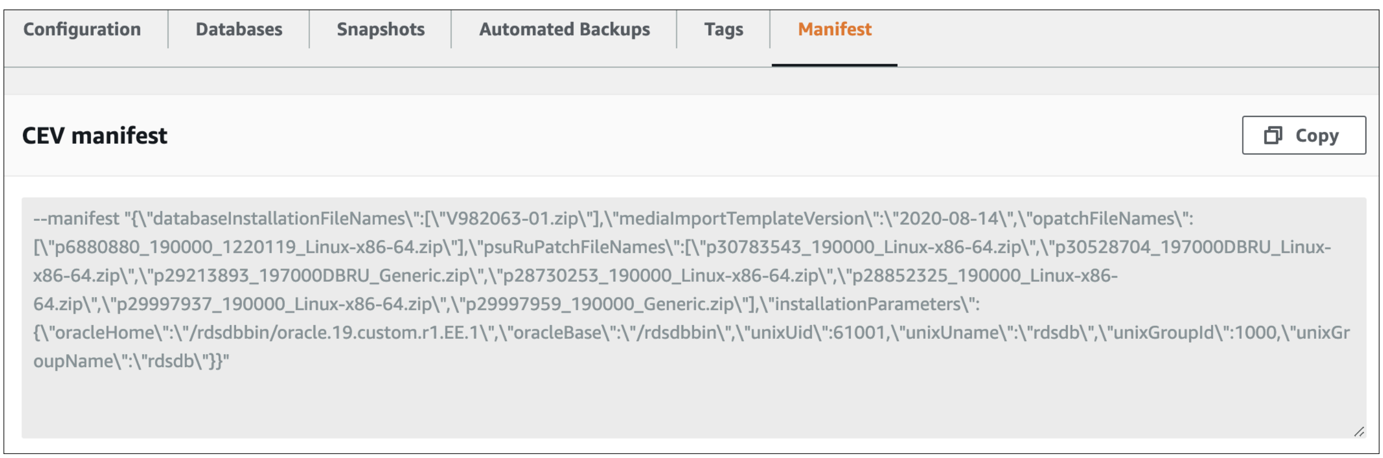
查看 CEV 详细信息

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Custom engine versions（自定义引擎版本）。

   Custom engine versions（自定义引擎版本）页面显示了当前存在的所有 CEV。如果您尚未创建任何 CEV，则该页面为空。
3. 选择要查看的 CEV 的名称。
4. 选择 Configuration（配置）以查看在清单中指定的安装参数。



5. 选择 Manifest（清单）以查看在 `create-custom-db-engine-version` 命令的 `--manifest` 选项中指定的安装参数。您可以复制此文本，根据需要替换值，然后在新命令中使用它们。



Amazon CLI

要通过使用 Amazon CLI 查看有关 CEV 的详细信息，请运行 [describe-db-engine-versions](#) 命令。

以下选项为必填：

- `--engine` *`engine-type`*，其中 *`engine-type`* 为 `custom-oracle-ee`、`custom-oracle-se2`、`custom-oracle-ee-cdb` 或 `custom-oracle-se2-cdb`
- `--engine-version` *`major-engine-version`*.*`customized_string`*

以下示例创建一个使用 Enterprise Edition 的非 CDB CEV。CEV 名称 `19.my_cev1` 以主要引擎版本号开头，这是必需项。

Example

对于 Linux、macOS 或 Unix：

```
aws rds describe-db-engine-versions \
    --engine custom-oracle-ee \
    --engine-version 19.my_cev1
```

对于 Windows：

```
aws rds describe-db-engine-versions ^
    --engine custom-oracle-ee ^
    --engine-version 19.my_cev1
```

以下不完整的示例输出显示了引擎、参数组、清单和其他信息。

```
"DBEngineVersions": [
    {
        "Engine": "custom-oracle-ee",
        "MajorEngineVersion": "19",
        "EngineVersion": "19.my_cev1",
        "DatabaseInstallationFilesS3BucketName": "us-east-1-123456789012-cev-customer-
installation-files",
        "DatabaseInstallationFilesS3Prefix": "123456789012/cev1",
        "CustomDBEngineVersionManifest": "{\n\"mediaImportTemplateVersion\":
 \"2020-08-14\",\n\"databaseInstallationFileNames\": [\n\"V982063-01.zip\"\n],
\n\"installationParameters\": {\n\"oracleBase\":\"/tmp\",\n\"oracleHome\":\"/
tmp/Oracle\"\n},\n\"opatchFileNames\": [\n\"p6880880_190000_Linux-x86-64.zip
\"\n],\n\"psuRuPatchFileNames\": [\n\"p32126828_190000_Linux-x86-64.zip
\"\n],\n\"otherPatchFileNames\": [\n\"p29213893_1910000DBRU_Generic.zip\",\n
\"p29782284_1910000DBRU_Generic.zip\",\n\"p28730253_190000_Linux-x86-64.zip\",\n
\"p29374604_1910000DBRU_Linux-x86-64.zip\",\n\"p28852325_190000_Linux-x86-64.zip\",
\n\"p29997937_190000_Linux-x86-64.zip\",\n\"p31335037_190000_Linux-x86-64.zip\",\n
\"p31335142_190000_Generic.zip\"\n]\n}\n",
```

```
        "DBParameterGroupFamily": "custom-oracle-ee-19",
        "DBEngineDescription": "Oracle Database server EE for RDS Custom",
        "DBEngineVersionArn": "arn:aws:rds:us-west-2:123456789012:cev:custom-oracle-
ee/19.my_cev1/0a123b45-6c78-901d-23e4-5678f901fg23",
        "DBEngineVersionDescription": "test",
        "KMSKeyId": "arn:aws:kms:us-east-1:123456789012:key/ab1c2de3-f4g5-6789-h012-
h3ijk4567l89",
        "CreateTime": "2022-11-18T09:17:07.693000+00:00",
        "ValidUpgradeTarget": [
        {
            "Engine": "custom-oracle-ee",
            "EngineVersion": "19.cev.2021-01.09",
            "Description": "test",
            "AutoUpgrade": false,
            "IsMajorVersionUpgrade": false
        }
]
```

删除 CEV

您可以使用 Amazon Web Services Management Console 或 Amazon CLI 删除 CEV。删除通常需要花费几分钟的时间。

要删除 CEV，它不能被以下任何一项使用：

- RDS Custom 数据库实例
- RDS Custom 数据库实例的快照
- RDS Custom 数据库实例的自动备份

控制台

要删除 CEV

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Custom engine versions（自定义引擎版本）。
3. 选择要删除其描述或状态的 CEV。
4. 对于操作，选择删除。

   Delete ***cev_name***?（删除 cev_name？）对话框显示。

5. 输入 **delete me**，然后选择删除。

   在 Custom engine versions（自定义引擎版本）页面中，横幅显示您的 CEV 正在被删除。

Amazon CLI

要通过使用 Amazon CLI 删除 CEV，请运行 delete-custom-db-engine-version 命令。

以下选项为必填：

- `--engine` *`engine-type`*，其中 *`engine-type`* 为 `custom-oracle-ee`、`custom-oracle-se2`、`custom-oracle-ee-cdb` 或 `custom-oracle-se2-cdb`
- `--engine-version` *`cev`* ，其中 *`cev`* 是要删除的自定义引擎版本的名称

以下示例会删除名为 `19.my_cev1` 的 CEV。

Example

对于 Linux、macOS 或 Unix：

```
aws rds delete-custom-db-engine-version \
    --engine custom-oracle-ee \
    --engine-version 19.my_cev1
```

对于 Windows：

```
aws rds delete-custom-db-engine-version ^
    --engine custom-oracle-ee ^
    --engine-version 19.my_cev1
```

为 Amazon RDS Custom for Oracle 配置数据库实例

您可以创建 RDS Custom 数据库实例，然后使用安全外壳 (SSH) 或 Amazon Systems Manager 连接到该实例。

有关连接和登录 RDS Custom for Oracle 数据库实例的更多信息，请参阅以下主题。

- 使用会话管理器连接到您的 RDS Custom 数据库实例
- 使用 SSH 连接到 RDS Custom 数据库实例
- 以 SYS 身份登录您的 RDS Custom for Oracle 数据库

创建 RDS Custom for Oracle 数据库实例

使用 Amazon Web Services Management Console 或 Amazon CLI 创建 Amazon RDS Custom for Oracle 数据库实例。该过程与创建 Amazon RDS 数据库实例的过程相似。有关更多信息，请参阅 创建 Amazon RDS 数据库实例。

如果您在 CEV 清单中包含了安装参数，则您的数据库实例将使用您指定的 Oracle 根目录、Oracle 主目录以及 UNIX/Linux 用户和组的 ID 和名称。`oratab` 文件由 Oracle Database 在安装期间创建，指向实际安装位置，而不是符号链接。当 RDS Custom for Oracle 运行命令时，它以配置的操作系统用户而不是原定设置用户 `rdsdb` 的身份运行。有关更多信息，请参阅 步骤 5：准备 CEV 清单。

在尝试创建或连接到 RDS Custom 数据库实例之前，请先完成为 Amazon RDS Custom for Oracle 设置您的环境中的任务。

控制台

要创建 RDS Custom for Oracle 数据库实例

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。
3. 选择创建数据库。
4. 在 Choose a database creation method（选择数据库创建方法）中，选择 Standard create（标准创建）。
5. 在引擎选项部分中，执行以下操作：

    a. 对于 Engine type（引擎类型），选择 Oracle。

b. 对于 Database management type（数据库管理类型），选择 Amazon RDS Custom。

c. 对于架构设置，执行以下操作之一：

- 选择多租户架构以创建容器数据库（CDB）。创建时，您的 CDB 包含一个 PDB 种子和一个初始 PDB。

> **Note**
>
> 只有 Oracle Database 19c 才支持多租户架构设置。

- 清除多租户架构以创建非 CDB。非 CDB 不能包含 PDB。

d. 对于版本，请选择 Oracle Enterprise Edition 或 Oracle Standard Edition 2。

e. 对于自定义引擎版本，选择现有 RDS Custom 自定义引擎版本（CEV）。CEV 采用以下格式：*major-engine-version*.*customized_string*。一个示例标识符是 `19.cdb_cev1`。

如果您在上一步中选择了多租户架构，则只能指定使用 `custom-oracle-ee-cdb` 或 `custom-oracle-se2-cdb` 引擎类型的 CEV。控制台会筛选掉使用不同引擎类型创建的 CEV。

6. 在 Templates（模板）中，选择 Production（生产）。

7. 在设置部分，执行以下操作：

a. 对于数据库实例标识符，请输入数据库实例的唯一名称。

b. 对于主用户名，输入用户名。您可以稍后从控制台检索此值。

当您连接到非 CDB 时，主用户是非 CDB 的用户。当您连接到 CDB 时，主用户是 PDB 的用户。要连接到 CDB 根目录，请登录主机，启动 SQL 客户端，然后使用 SQL 命令创建管理用户。

c. 清除自动生成密码。

8. 选择数据库实例类。

有关支持的类，请参阅 RDS Custom for Oracle 的数据库实例类支持。

9. 在 Storage（存储）部分中，执行以下操作：

a. 对于存储类型，选择 SSD 类型：io1、gp2 或 gp3。您还有以下附加选项：

  - 对于 io1 或 gp3，为预调配 IOPS 选择费率。io1 的原定设置值为 1000，gp3 的原定设置值为 12000。
  - 对于 gp3，为存储吞吐量选择速率。原定设置值为 500MiBps。

   b. 对于分配的存储空间，选择存储大小。原定设置值为 40GiB。

10. 对于连接，请指定您的虚拟私有云（VPC）、数据库子网组和 VPC 安全组（防火墙）。
11. 对于 RDS Custom 安全性，执行以下操作：

    a. 对于 IAM instance profile（IAM 实例配置文件），为 RDS Custom for Oracle 数据库实例选择实例配置文件。

       IAM 实例配置文件必须以 `AWSRDSCustom` 开头，例如，*`AWSRDSCustomInstanceProfileForRdsCustomInstance`*。

    b. 对于 Encryption（加密），选择 Enter a key ARN（输入一个密钥 ARN）以列出可用的 Amazon KMS 密钥。然后从列表中选择您的密钥。

       RDS Custom 需要 Amazon KMS 密钥。有关更多信息，请参阅 步骤 1：创建或重用对称加密 Amazon KMS 密钥。

12. 对于数据库选项，执行以下操作：

    a. （可选）对于系统 ID（SID），输入 Oracle SID 的值，该值也是您 CDB 的名称。SID 是管理您数据库文件的 Oracle 数据库实例的名称。在这种情况下，“Oracle 数据库实例”一词仅指系统全局区域（SGA）和 Oracle 后台进程。如果未指定 SID，则该值原定设置为 **`RDSCDB`**。

    b. （可选）对于初始数据库名称，输入名称。默认值为 **`ORCL`**。在多租户架构中，初始数据库名称是 PDB 名称。

       > **Note**
       >
       > SID 和 PDB 的名称必须不同。

    c. 对于选项组，选择一个选项组或接受默认值。

       > **Note**
       >
       > RDS Custom for Oracle 唯一支持的选项是 `Timezone`。有关更多信息，请参阅 Oracle 时区。

    d. 对于备份保留期，选择一个值。您不能选择 0 天。

e. 对于其余部分，请指定首选的 RDS Custom 数据库实例设置。有关每项设置的信息，请参阅数据库实例的设置。以下设置不会显示在控制台中，也不受支持：

- 处理器功能
- Storage autoscaling (存储自动扩展)
- Database authentication（数据库身份验证）中的 Password and Kerberos authentication（密码和 Kerberos 身份验证）选项（仅支持 Password authentication（密码身份验证））
- Performance Insights
- 日志导出
- 启用自动次要版本升级
- 删除保护

13. 选择创建数据库。

> **Important**
>
> 当您创建 RDS Custom for Oracle 数据库实例时，您可能会收到以下错误：The service-linked role is in the process of being created.（服务相关角色正在创建过程中）。请稍后重试。如果您这样做，请等待几分钟，然后重新尝试创建数据库实例。

View credential details（查看凭证详细信息）按钮显示在 Databases（数据库）页面中。

要查看 RDS Custom 数据库实例的主用户名和密码，请选择 View credential details（查看凭证详细信息）。

要以主用户身份连接到数据库实例，请使用显示的用户名和密码。

> **Important**
>
> 您无法再次在控制台中查看主用户密码。如果您不记录它，您可能需要更改它。要在 RDS Custom 数据库实例可用后更改主用户密码，请登录数据库并运行 `ALTER USER` 命令。您可以使用控制台中的修改选项重置密码。

14. 选择 Databases（数据库）以查看 RDS Custom 数据库实例列表。
15. 选择您刚创建的 RDS Custom 数据库实例。

在 RDS 控制台上，将显示新 RDS Custom 数据库实例的详细信息。

- 数据库实例具有 creating（正在创建）状态，直到 RDS Custom 数据库实例完成创建并可供使用。当状态变为 Available (可用) 时，您可以连接到该数据库实例。根据所分配的实例类和存储的不同，新数据库实例可能需要数分钟时间才能变得可用。
- Role（角色）具有值 Instance (RDS Custom)（实例 (RDS Custom)）。
- RDS Custom automation mode（RDS Custom 自动化模式）具有值 Full automation（完全自动化）。此设置意味着数据库实例可以提供自动监控和实例恢复。

Amazon CLI

您可以通过使用 create-db-instance Amazon CLI 命令创建 RDS Custom 数据库实例。

以下选项为必填：

- `--db-instance-identifier`
- `--db-instance-class`（有关支持的实例类的列表，请参阅 RDS Custom for Oracle 的数据库实例类支持）
- `--engine` *`engine-type`*，其中 *`engine-type`* 为 `custom-oracle-ee`、`custom-oracle-se2`、`custom-oracle-ee-cdb` 或 `custom-oracle-se2-cdb`
- `--engine-version` *`cev`*（其中 *`cev`* 是您在 创建 CEV 中指定的自定义引擎版本的名称）
- `--kms-key-id` *`my-kms-key`*
- `--backup-retention-period` *`days`*（其中，*`days`* 值大于 `0`）
- `--no-auto-minor-version-upgrade`
- `--custom-iam-instance-profile AWSRDSCustomInstanceProfile-`*`us-east-1`*（其中，*`region`* 是您在其中创建数据库实例的 Amazon Web Services 区域）

以下示例将创建名为 `my-cfo-cdb-instance` 的 RDS Custom 数据库实例。该数据库是一个 CDB，其非原定设置名称为 *`MYCDB`*。非原定设置 PDB 名称是 *`MYPDB`*。备份保留期为三天。

Example

对于 Linux、macOS 或 Unix：

```
aws rds create-db-instance \
    --engine custom-oracle-ee-cdb \
```

```
    --db-instance-identifier my-cfo-cdb-instance \
    --engine-version 19.cdb_cev1 \
    --db-name MYPDB \
    --db-system-id MYCDB \
    --allocated-storage 250 \
    --db-instance-class db.m5.xlarge \
    --db-subnet-group mydbsubnetgroup \
    --master-username myuser \
    --master-user-password mypassword \
    --backup-retention-period 3 \
    --port 8200 \
    --kms-key-id my-kms-key \
    --no-auto-minor-version-upgrade \
    --custom-iam-instance-profile AWSRDSCustomInstanceProfile-us-east-1
```

对于 Windows：

```
aws rds create-db-instance ^
    --engine custom-oracle-ee-cdb ^
    --db-instance-identifier my-cfo-cdb-instance ^
    --engine-version 19.cdb_cev1 ^
    --db-name MYPDB ^
    --db-system-id MYCDB ^
    --allocated-storage 250 ^
    --db-instance-class db.m5.xlarge ^
    --db-subnet-group mydbsubnetgroup ^
    --master-username myuser ^
    --master-user-password mypassword ^
    --backup-retention-period 3 ^
    --port 8200 ^
    --kms-key-id my-kms-key ^
    --no-auto-minor-version-upgrade ^
    --custom-iam-instance-profile AWSRDSCustomInstanceProfile-us-east-1
```

> **Note**
>
> 作为安全最佳实践，请指定除此处所示提示以外的密码。

使用 `describe-db-instances` 命令以获取有关您的实例的详细信息。

Example

```
aws rds describe-db-instances --db-instance-identifier my-cfo-cdb-instance
```

以下部分输出显示了引擎、参数组和其他信息。

```
        {
            "DBInstanceIdentifier": "my-cfo-cdb-instance",
            "DBInstanceClass": "db.m5.xlarge",
            "Engine": "custom-oracle-ee-cdb",
            "DBInstanceStatus": "available",
            "MasterUsername": "admin",
            "DBName": "MYPDB",
            "DBSystemID": "MYCDB",
            "Endpoint": {
                "Address": "my-cfo-cdb-instance.abcdefghijkl.us-
east-1.rds.amazonaws.com",
                "Port": 1521,
                "HostedZoneId": "A1B2CDEFGH34IJ"
            },
            "AllocatedStorage": 100,
            "InstanceCreateTime": "2023-04-12T18:52:16.353000+00:00",
            "PreferredBackupWindow": "08:46-09:16",
            "BackupRetentionPeriod": 7,
            "DBSecurityGroups": [],
            "VpcSecurityGroups": [
                {
                    "VpcSecurityGroupId": "sg-0a1bcd2e",
                    "Status": "active"
                }
            ],
            "DBParameterGroups": [
                {
                    "DBParameterGroupName": "default.custom-oracle-ee-cdb-19",
                    "ParameterApplyStatus": "in-sync"
                }
            ],
...
```

多租户架构注意事项

如果您使用 Oracle 多租户架构（`custom-oracle-ee-cdb` 或 `custom-oracle-se2-cdb` 引擎类型）创建 Amazon RDS Custom for Oracle 数据库实例，则您的数据库是容器数据库（CDB）。如果您未指定 Oracle 多租户架构，则您的数据库是使用 `custom-oracle-ee` 或 `custom-oracle-se2` 引擎类型的传统非 CDB。非 CDB 不能包含可插拔数据库（PDB）。有关更多信息，请参阅 Amazon RDS Custom for Oracle 的数据库架构。

创建 RDS Custom for Oracle CDB 实例时，请考虑以下几点：

- 您只能从 Oracle Database 19c CEV 创建多租户数据库。
- 仅当 CEV 使用 `custom-oracle-ee-cdb` 或 `custom-oracle-se2-cdb` 引擎类型时，您才能创建 CDB 实例。
- 如果您使用 Standard Edition 2 创建 CDB 实例，则 CDB 最多可以包含 3 个 PDB。
- 原定设置情况下，您的 CDB 命名为 `RDSCDB`，这也是 Oracle 系统 ID（Oracle SID）的名称。您可以选择其他名称。
- 您的 CDB 仅包含一个初始 PDB。PDB 名称原定设置为 `ORCL`。您可以为初始 PDB 选择不同的名称，但是 Oracle SID 和 PDB 名称不能相同。
- RDS Custom for Oracle 不为 PDB 提供 API。要创建其他 PDB，请使用 Oracle SQL 命令 `CREATE PLUGGABLE DATABASE`。RDS Custom for Oracle 不限制您可以创建的 PDB 数量。通常，您负责创建和管理 PDB，就像在本地部署中一样。
- 您不能使用 RDS API 来创建、修改和删除 PDB：您必须使用 Oracle SQL 语句来执行这些操作。如果您使用 Oracle SQL 创建 PDB，建议您之后拍摄手动快照，以防需要执行时间点故障恢复（PITR）。
- 您无法使用 Amazon RDS API 重命名现有 PDB。您也无法使用 `modify-db-instance` 命令重命名 CDB。
- CDB 根目录的打开模式在主数据库上为 `READ WRITE`，在已挂载的备用数据库上为 `MOUNTED`。RDS Custom for Oracle 会在打开 CDB 时尝试打开所有 PDB。如果 RDS Custom for Oracle 无法打开所有 PDB，它会发出事件 `tenant database shutdown`。

RDS Custom 服务相关角色

服务相关角色 为 Amazon RDS Custom 提供针对您的 Amazon Web Services 账户中的资源的访问权限。这样可以让您更轻松地使用 RDS Custom，因为您不必手动添加必要的权限。RDS Custom 定义其服务相关角色的权限，除非另外定义，否则只有 RDS Custom 可以代入该角色。定义的权限包括信任策略和权限策略，以及不能附加到任何其他 IAM 实体的权限策略。

创建 RDS Custom 数据库实例时，将创建并使用 Amazon RDS 和 RDS Custom 服务相关角色（如果它们尚未存在）。有关更多信息，请参阅 将服务相关角色用于 Amazon RDS。

首次创建 RDS Custom for Oracle 数据库实例时，您可能会收到以下错误：The service-linked role is in the process of being created.（服务相关角色正在创建过程中）。请稍后重试。如果您这样做，请等待几分钟，然后重新尝试创建数据库实例。

在 RDS Custom for Oracle 数据库实例上安装其他软件组件

在新创建的数据库实例中，您的数据库环境包括 Oracle 二进制文件、数据库和数据库侦听器。您可能需要在数据库实例的主机操作系统上安装其他软件。例如，您可能需要安装 Oracle Application Express（APEX）、Oracle Enterprise Manager（OEM）代理或 Guardium S-TAP 代理。有关指南和总体说明，请参阅详细的 Amazon 博客文章在 Amazon RDS Custom for Oracle 上安装其他软件组件。

使用会话管理器连接到您的 RDS Custom 数据库实例

创建 RDS Custom 数据库实例后，您可以使用 Amazon Systems Manager Session Manager 连接到该实例。当数据库实例无法公开访问时，这是首选技术。

会话管理器可让您通过基于浏览器的 Shell 或通过 Amazon CLI 访问 Amazon EC2 实例。有关更多信息，请参阅 Amazon Systems Manager 会话管理器。

控制台

使用会话管理器连接到您的数据库实例

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，请选择 Databases（数据库），然后选择要连接到的 RDS Custom 数据库实例。
3. 选择配置。
4. 请注意您的数据库实例的 Resource ID（资源 ID）。例如，资源 ID 可能为 `db-ABCDEFGHIJKLMNOPQRS0123456`。
5. 通过以下网址打开 Amazon EC2 控制台：https://console.aws.amazon.com/ec2/。
6. 在导航窗格中，选择实例。
7. 查找 EC2 实例的名称，然后单击与其关联的实例 ID。例如，实例 ID 可能为 `i-abcdefghijklm01234`。
8. 选择连接。

9. 选择 Session Manager（会话管理器）。
10. 选择连接。

    系统将为您的会话打开一个窗口。

Amazon CLI

您可以使用 Amazon CLI 连接到 RDS Custom 数据库实例。此方法需要 Amazon CLI 的会话管理器插件。要了解如何安装插件，请参阅安装 Amazon CLI 的会话管理器插件。

要查找 RDS Custom 数据库实例的数据库资源 ID，请使用 `aws rds describe-db-instances`。

```
aws rds describe-db-instances \
    --query 'DBInstances[*].[DBInstanceIdentifier,DbiResourceId]' \
    --output text
```

以下示例输出显示了 RDS Custom 实例的资源 ID。前缀为 `db-`。

```
db-ABCDEFGHIJKLMNOPQRS0123456
```

要查找数据库实例的 EC2 实例 ID，请使用 `aws ec2 describe-instances`。以下示例将 `db-ABCDEFGHIJKLMNOPQRS0123456` 用于资源 ID。

```
aws ec2 describe-instances \
    --filters "Name=tag:Name,Values=db-ABCDEFGHIJKLMNOPQRS0123456" \
    --output text \
    --query 'Reservations[*].Instances[*].InstanceId'
```

以下示例输出显示了 EC2 实例 ID。

```
i-abcdefghijklm01234
```

使用 `aws ssm start-session` 命令，以在 `--target` 参数中提供 EC2 实例 ID。

```
aws ssm start-session --target "i-abcdefghijklm01234"
```

成功的连接如下所示。

```
Starting session with SessionId: yourid-abcdefghijklm1234
```

```
[ssm-user@ip-123-45-67-89 bin]$
```

使用 SSH 连接到 RDS Custom 数据库实例

Secure Shell 协议（SSH）是一种支持通过不安全网络进行加密通信的网络协议。创建 RDS Custom 数据库实例后，您可以使用 ssh 客户端连接到此实例。有关更多信息，请参阅使用 SSH 连接到 Linux 实例。

您的 SSH 连接技术取决于您的数据库实例是否为私有实例，这意味着它不接受来自公共互联网的连接。在这种情况下，您必须使用 SSH 隧道将 ssh 实用程序连接到您的实例。该技术在现有 SSH 会话内通过专用数据流（隧道）传输数据。您可以使用 Amazon Systems Manager 配置 SSH 隧道。

> **Note**
>
> 支持各种策略以访问私有实例。要了解如何使用堡垒主机将 ssh 客户端连接到私有实例，请参阅 Amazon 上的 Linux 堡垒机主机。要了解如何配置端口转发，请参阅使用 Amazon Systems Manager Session Manager 进行端口转发。

如果您的数据库实例位于公有子网中并且具有公共可用设置，则不需要 SSH 隧道。您可以像连接到公有 Amazon EC2 实例一样使用 SSH 进行连接。

要将 ssh 客户端连接到数据库实例，请完成以下步骤：

1. 步骤 1：配置您的数据库实例以允许 SSH 连接
2. 步骤 2：检索您的 SSH 私密密钥和 EC2 实例 ID
3. 步骤 3：使用 ssh 实用程序连接到 EC2 实例

步骤 1：配置您的数据库实例以允许 SSH 连接

要确保您的数据库实例可接受 SSH 连接，请执行以下操作：

- 确保数据库实例安全组允许端口 22 上的 TCP 入站连接。

  要了解如何为数据库实例配置安全组，请参阅使用安全组控制访问权限。
- 如果您不打算使用 SSH 隧道，请确保数据库实例位于公有子网中并且可供公开访问。

  在控制台中，相关字段是数据库详细信息页面的连接和安全性选项卡上的可公开访问。要在 CLI 中检查设置，请运行以下命令：

```
aws rds describe-db-instances \
--query 'DBInstances[*].
{DBInstanceIdentifier:DBInstanceIdentifier,PubliclyAccessible:PubliclyAccessible}' \
--output table
```

要更改数据库实例的可访问性设置，请参阅修改 Amazon RDS 数据库实例。

步骤 2：检索您的 SSH 私密密钥和 EC2 实例 ID

要使用 SSH 连接到数据库实例，您需要与该实例关联的 SSH 密钥对。RDS Custom 代表您使用命名约定 `do-not-delete-rds-custom-ssh-privatekey-`*`resource_id`*`-`*`uuid`* 或 `rds-custom!oracle-do-not-delete-`*`resource_id`*`-`*`uuid`*`-ssh-privatekey` 创建 SSH 密钥对。Amazon Secrets Manager 将您的 SSH 私有密钥存储为密钥。

使用 Amazon Web Services Management Console或 Amazon CLI 检索您的 SSH 私密密钥。如果您的实例有公有 DNS，并且您不打算使用 SSH 隧道，则还会检索 DNS 名称。您指定公有连接的 DNS 名称。

控制台

检索私密 SSH 密钥

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，请选择 Databases（数据库），然后选择要连接到的 RDS Custom 数据库实例。
3. 选择配置。
4. 请注意 Resource ID（资源 ID）值。例如，数据库实例资源 ID 可能为 `db-ABCDEFGHIJKLMNOPQRS0123456`。
5. 通过以下网址打开 Amazon EC2 控制台：https://console.aws.amazon.com/ec2/。
6. 在导航窗格中，选择 Instances (实例)。
7. 查找 EC2 实例的名称，然后选择与其关联的实例 ID。例如，EC2 实例 ID 可能为 `i-abcdefghijklm01234`。
8. 在 Details（详细信息）中，查找 Key pair name（密钥对名称）。密钥对名称包括数据库实例资源 ID。例如，密钥对名称可能为 `do-not-delete-rds-custom-ssh-privatekey-db-ABCDEFGHIJKLMNOPQRS0123456-0d726c` 或 `rds-custom!oracle-do-not-delete-db-ABCDEFGHIJKLMNOPQRS0123456-0d726c-ssh-privatekey`。

9. 如果您的 EC2 实例是公有的，请记下公有 IPv4 DNS。例如，公有域名系统 (DNS) 地址可能为 `ec2-12-345-678-901.us-east-2.compute.amazonaws.com`。
10. 打开位于 https://console.aws.amazon.com/secretsmanager/ 的 Amazon Secrets Manager 控制台。
11. 选择与密钥对同名的密钥。
12. 选择 Retrieve secret value (检索密钥值)。
13. 将 SSH 私有密钥复制到文本文件中，然后使用 `.pem` 扩展名保存该文件。例如，将文件另存为 `/tmp/do-not-delete-rds-custom-ssh-privatekey-db-ABCDEFGHIJKLMNOPQRS0123456-0d726c.pem` 或 `/tmp/rds-custom!oracle-do-not-delete-db-ABCDEFGHIJKLMNOPQRS0123456-0d726c-ssh-privatekey.pem`。

Amazon CLI

要检索 SSH 私有密钥并将其保存在 .pem 文件中，可以使用 Amazon CLI。

1. 使用 `aws rds describe-db-instances` 查找 RDS Custom 数据库实例的数据库资源 ID。

```
aws rds describe-db-instances \
    --query 'DBInstances[*].[DBInstanceIdentifier,DbiResourceId]' \
    --output text
```

   以下示例输出显示了 RDS Custom 实例的资源 ID。前缀为 `db-`。

```
db-ABCDEFGHIJKLMNOPQRS0123456
```

2. 使用 `aws ec2 describe-instances` 查找数据库实例的 EC2 实例 ID。以下示例将 `db-ABCDEFGHIJKLMNOPQRS0123456` 用于资源 ID。

```
aws ec2 describe-instances \
    --filters "Name=tag:Name,Values=db-ABCDEFGHIJKLMNOPQRS0123456" \
    --output text \
    --query 'Reservations[*].Instances[*].InstanceId'
```

   以下示例输出显示了 EC2 实例 ID。

```
i-abcdefghijklm01234
```

3. 要查找密钥名称，请指定 EC2 实例 ID。以下示例描述了 EC2 实例 `i-0bdc4219e66944afa`。

```
aws ec2 describe-instances \
    --instance-ids i-0bdc4219e66944afa \
    --output text \
    --query 'Reservations[*].Instances[*].KeyName'
```

以下示例输出显示了使用命名格式 `do-not-delete-rds-custom-ssh-privatekey-`*resource_id*`-`*uuid* 或 `rds-custom!oracle-do-not-delete-`*resource_id*`-`*uuid*`-ssh-privatekey` 的密钥名称。

```
do-not-delete-rds-custom-ssh-privatekey-db-ABCDEFGHIJKLMNOPQRS0123456-0d726c
rds-custom!oracle-do-not-delete-db-ABCDEFGHIJKLMNOPQRS0123456-0d726c-ssh-privatekey
```

4. 使用 `aws secretsmanager` 将私有密钥保存在以密钥命名的 .pem 文件中。

   以下示例将密钥 `do-not-delete-rds-custom-ssh-privatekey-db-ABCDEFGHIJKLMNOPQRS0123456-0d726c` 保存到您的 `/tmp` 目录中的文件。

```
aws secretsmanager get-secret-value \
    --secret-id do-not-delete-rds-custom-ssh-privatekey-db-
ABCDEFGHIJKLMNOPQRS0123456-0d726c \
    --query SecretString \
    --output text >/tmp/do-not-delete-rds-custom-ssh-privatekey-db-
ABCDEFGHIJKLMNOPQRS0123456-0d726c.pem
```

   以下示例将密钥 `rds-custom!oracle-do-not-delete-db-ABCDEFGHIJKLMNOPQRS0123456-0d726c-ssh-privatekey` 保存到您的 `/tmp` 目录中的文件。

```
aws secretsmanager get-secret-value \
    --secret-id rds-custom!oracle-do-not-delete-db-
ABCDEFGHIJKLMNOPQRS0123456-0d726c-ssh-privatekey \
    --query SecretString \
    --output text >/tmp/rds-custom!oracle-do-not-delete-db-
ABCDEFGHIJKLMNOPQRS0123456-0d726c-ssh-privatekey.pem
```

步骤 3：使用 ssh 实用程序连接到 EC2 实例

您的连接技术取决于您是连接到私有数据库实例，还是连接到公有实例。私有连接要求您通过 Amazon Systems Manager 配置 SSH 隧道。

使用 ssh 实用程序连接到 EC2 实例

1. 对于私有连接，请将您的 SSH 配置文件修改为代理对 Amazon Systems Manager Session Manager 发出的命令。对于公有连接，请跳至步骤 2。

   将以下行添加到 `~/.ssh/config`。这些行为名称以 `i-` 或 `mi-` 开头的主机代理 SSH 命令。

   ```
   Host i-* mi-*
       ProxyCommand sh -c "aws ssm start-session --target %h --document-name AWS-
   StartSSHSession --parameters 'portNumber=%p'"
   ```

2. 包含您的 .pem 文件的目录的更改。使用 `chmod`，将权限设置为 `400`。

   以下示例更改为 `/tmp` 目录并设置 .pem 文件 `do-not-delete-rds-custom-ssh-privatekey-db-ABCDEFGHIJKLMNOPQRS0123456-0d726c.pem` 的权限。

   ```
   cd /tmp
   chmod 400 do-not-delete-rds-custom-ssh-privatekey-db-
   ABCDEFGHIJKLMNOPQRS0123456-0d726c.pem
   ```

   以下示例更改为 `/tmp` 目录并设置 .pem 文件 `rds-custom!oracle-do-not-delete-db-ABCDEFGHIJKLMNOPQRS0123456-0d726c-ssh-privatekey.pem` 的权限。

   ```
   cd /tmp
   chmod 400 rds-custom!oracle-do-not-delete-db-ABCDEFGHIJKLMNOPQRS0123456-0d726c-ssh-
   privatekey.pem
   ```

3. 运行 ssh 实用程序，指定 .pem 文件和公有 DNS 名称（用于公有连接）或 EC2 实例 ID（用于私有连接）。以用户 `ec2-user` 身份登录。

   以下示例使用 DNS 名称 `ec2-12-345-678-901.us-east-2.compute.amazonaws.com` 连接到公有实例。

   ```
   # .pem file using naming prefix do-not-delete
   ssh -i \
   ```

```
  "do-not-delete-rds-custom-ssh-privatekey-db-
ABCDEFGHIJKLMNOPQRS0123456-0d726c.pem" \
  ec2-user@ec2-12-345-678-901.us-east-2.compute.amazonaws.com

# .pem file using naming prefix rds-custom!oracle-do-not-delete
ssh -i \
  "rds-custom!oracle-do-not-delete-db-ABCDEFGHIJKLMNOPQRS0123456-0d726c-ssh-
privatekey.pem" \
  ec2-user@ec2-12-345-678-901.us-east-2.compute.amazonaws.com
```

以下示例使用 EC2 实例 ID `i-0bdc4219e66944afa` 连接到私有实例。

```
# .pem file using naming prefix do-not-delete
ssh -i \
  "do-not-delete-rds-custom-ssh-privatekey-db-
ABCDEFGHIJKLMNOPQRS0123456-0d726c.pem" \
  ec2-user@i-0bdc4219e66944afa

# .pem file using naming prefix rds-custom!oracle-do-not-delete
ssh -i \
  "rds-custom!oracle-do-not-delete-db-ABCDEFGHIJKLMNOPQRS0123456-0d726c-ssh-
privatekey.pem" \
  ec2-user@i-0bdc4219e66944afa
```

以 SYS 身份登录您的 RDS Custom for Oracle 数据库

创建 RDS Custom 数据库实例后，您能够以用户 `SYS` 身份登录 Oracle 数据库，这会为您提供 `SYSDBA` 权限。您有以下登录选项：

- 从 Secrets Manager 获取 `SYS` 密码，然后在 SQL 客户端中指定此密码。
- 使用操作系统身份验证登录您的数据库。在这种情况下，不需要密码。

查找您的 RDS Custom for Oracle 数据库的 SYS 密码

您能够以 `SYS` 或 `SYSTEM` 身份或通过在 API 调用中指定主用户名来登录 Oracle 数据库。`SYS` 和 `SYSTEM` 的密码存储在 Secrets Manager 中。

此密钥使用命名格式 `do-not-delete-rds-custom-`*`resource_id`*`-`*`uuid`* 或 `rds-custom!oracle-do-not-delete-`*`resource_id`*`-`*`uuid`*。您可以使用 Amazon Web Services Management Console找到密码。

控制台

在 Secrets Manager 中查找数据库的 SYS 密码

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在 RDS 控制台中，完成以下步骤：
   a. 在导航窗格中，选择数据库。
   b. 选择您的 RDS Custom for Oracle 数据库实例的名称。
   c. 选择配置。
   d. 复制资源 ID 下方的值。例如，资源 ID 可能是 db-ABC12CDE3FGH4I5JKLMNO6PQR7。
3. 通过 https://console.aws.amazon.com/secretsmanager/ 打开 Secrets Manager 控制台。
4. 在 Secrets Manager 控制台中，完成以下步骤：
   a. 在左侧导航窗格中，选择密钥。
   b. 按您在步骤 2.d 中复制的资源 ID 筛选密钥。
   c. 选择使用以下命名格式的密钥：do-not-delete-rds-custom-***`resource_id-uuid`*** 或 rds-custom!oracle-do-not-delete-***`resource_id-uuid`***。*`resource_id`* 是您在步骤 2.d 中复制的资源 ID。

      例如，如果资源 ID 为 db-ABC12CDE3FGH4I5JKLMNO6PQR7 并且 UUID 为 1234ab，则您的密钥命名为 do-not-delete-rds-custom-db-ABC12CDE3FGH4I5JKLMNO6PQR7-1234ab 或 rds-custom!oracle-do-not-delete-db-ABC12CDE3FGH4I5JKLMNO6PQR7-1234ab。
   d. 在密钥值中，选择检索密钥值。
   e. 在键/值中，复制密码的值。
5. 在数据库实例上安装 SQL*Plus 并以 SYS 身份登录数据库。有关更多信息，请参阅 步骤 3：将 SQL 客户端连接到 Oracle 数据库实例。

使用操作系统身份验证登录您的 RDS Custom for Oracle 数据库

操作系统用户 `rdsdb` 拥有 Oracle 数据库二进制文件。您可以切换到 `rdsdb` 用户，无需密码即可登录 RDS Custom for Oracle 数据库。

1. 使用 Amazon Systems Manager 连接到您的数据库实例。有关更多信息，请参阅 使用会话管理器连接到您的 RDS Custom 数据库实例。

2. 在 Web 浏览器中，转到 https://www.oracle.com/database/technologies/instant-client/linux-x86-64-downloads.html。
3. 对于网页上显示的最新数据库版本，请复制 Instant Client 基本软件包和 SQL*Plus 软件包的 .rpm 链接（不是 .zip 链接）。例如，以下链接适用于 Oracle Database 21.9 版本：

   - https://download.oracle.com/otn_software/linux/instantclient/219000/oracle-instantclient-basic-21.9.0.0.0-1.el8.x86_64.rpm
   - https://download.oracle.com/otn_software/linux/instantclient/219000/oracle-instantclient-sqlplus-21.9.0.0.0-1.el8.x86_64.rpm
4. 在 SSH 会话中，运行 `wget` 命令以从您在上一步中获得的链接下载 .rpm 文件。以下示例下载 Oracle Database 21.9 版本的 .rpm 文件：

```
wget https://download.oracle.com/otn_software/linux/instantclient/219000/oracle-instantclient-basic-21.9.0.0.0-1.el8.x86_64.rpm
wget https://download.oracle.com/otn_software/linux/instantclient/219000/oracle-instantclient-sqlplus-21.9.0.0.0-1.el8.x86_64.rpm
```

5. 通过运行 yum 命令安装软件包，如下所示：

```
sudo yum install oracle-instantclient-*.rpm
```

6. 切换到 `rdsdb` 用户。

```
sudo su - rdsdb
```

7. 使用操作系统身份验证登录数据库。

```
$ sqlplus / as sysdba

SQL*Plus: Release 21.0.0.0.0 - Production on Wed Apr 12 20:11:08 2023
Version 21.9.0.0.0

Copyright (c) 1982, 2020, Oracle.  All rights reserved.


Connected to:
Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production
Version 19.10.0.0.0
```

管理 Amazon RDS Custom for Oracle 数据库实例

Amazon RDS Custom 支持 Amazon RDS 数据库实例的常见管理任务子集。在下文中，您可以找到使用 Amazon Web Services Management Console和 Amazon CLI 的受支持的 RDS Custom for Oracle 管理任务的说明。

主题

在 RDS Custom for Oracle 中使用容器数据库（CDB）

您可以使用 Oracle 多租户架构（`custom-oracle-ee-cdb` 或 `custom-oracle-se2-cdb` 引擎类型）或传统的非 CDB 架构（`custom-oracle-ee` 或 `custom-oracle-se2` 引擎类型）创建 RDS Custom for Oracle 数据库实例。创建容器数据库（CDB）时，它包含一个可插拔数据库（PDB）和一个 PDB 种子。您可以使用 Oracle SQL 手动创建其他 PDB。

PDB 和 CDB 名称

创建 RDS Custom for Oracle CDB 实例时，您需要为初始 PDB 指定名称。默认情况下，初始 PDB 命名为 `ORCL`。您可以选择其他名称。

默认情况下，您的 CDB 命名为 `RDSCDB`。您可以选择其他名称。CDB 名称也是您的 Oracle 系统标识符（SID）的名称，它唯一标识管理 CDB 的内存和进程。有关 Oracle SID 的更多信息，请参阅《Oracle 数据库概念》中的 Oracle 系统标识符（SID）。

您无法使用 Amazon RDS API 重命名现有 PDB。您也无法使用 `modify-db-instance` 命令重命名 CDB。

PDB 管理

在 RDS Custom for Oracle 责任共担模式下，您负责管理 PDB 和创建任何其他 PDB。RDS Custom 不限制 PDB 的数量。您可以通过连接到 CDB 根目录并运行 SQL 语句来手动创建、修改和删除 PDB。在 Amazon EBS 数据卷上创建 PDB，以防止数据库实例超出支持边界。

要修改 CDB 或 PDB，请完成以下步骤：

1. 暂停自动化以防止干扰 RDS Custom 操作。
2. 修改您的 CDB 或 PDB。
3. 备份任何修改过的 PDB。
4. 恢复 RDS Custom 自动化。

CDB 根目录的自动恢复

RDS Custom 保持 CDB 根目录处于打开状态的方式与保持非 CDB 打开的方式相同。如果 CDB 根目录的状态发生变化，监控和恢复自动化会尝试将 CDB 根目录恢复到所需状态。与非 CDB 架构类似，当根 CDB 关闭（`RDS-EVENT-0004`）或重新启动（`RDS-EVENT-0006`）时，您会收到 RDS 事件通知。RDS Custom 尝试在数据库实例启动时以 `READ WRITE` 模式打开所有 PDB。如果某些 PDB 无法打开，RDS Custom 会发布以下事件：`tenant database shutdown`。

使用 RDS Custom for Oracle 的高可用性特征

要支持 RDS Custom for Oracle 数据库实例之间的复制，您可以使用 Oracle Data Guard 配置高可用性（HA）。主数据库实例会自动将数据同步到备用实例。此特征仅在 Enterprise Edition 中受支持。

您可以通过以下方式配置高可用性环境：

- 在不同可用区 (AZ) 中配置备用实例，以应对可用区故障。
- 将备用数据库置于挂载模式或只读模式。
- 故障转移或从主数据库切换到备用数据库，而不会丢失数据。
- 通过为本地实例配置高可用性，然后故障转移或切换到 RDS Custom 备用数据库来迁移数据。

要了解如何配置高可用性，请参阅白皮书使用只读副本为 Amazon RDS Custom for Oracle 构建高可用性。您可以执行以下任务：

- 使用虚拟专用网络 (VPN) 隧道为您的高可用性实例进行传输中的数据加密。RDS Custom 不会自动配置传输中的加密。

- 配置 Oracle 快速故障转移观察器 (FSFO) 以监控您的高可用性实例。
- 在满足必要条件时，允许观察期执行自动故障转移。

自定义 RDS Custom 环境

RDS Custom for Oracle 包含内置特征，允许您在不暂停自动化的情况下自定义数据库实例环境。例如，您可以使用 RDS API 自定义您的环境，如下所示：

- 创建和还原数据库快照以创建克隆环境。
- 创建只读副本。
- 修改存储设置。
- 更改 CEV 以应用版本更新

对于某些自定义，例如更改字符集，您不能使用 RDS API。在这些情况下，您需要通过以根用户身份访问 Amazon EC2 实例或以 `SYSDBA` 身份登录 Oracle 数据库来手动更改环境。

要手动自定义您的实例，您必须暂停并恢复 RDS Custom 自动化。这一暂停可确保您的自定义项不会干扰 RDS Custom 自动化。通过这种方式，您可以避免突破支持边界，这会将实例置于 `unsupported-configuration` 状态，直到您修复底层问题为止。修改 RDS Custom for Oracle 数据库实例时，暂停和恢复是唯一受支持的自动化任务。

自定义 RDS Custom 环境的一般步骤

要自定义 RDS Custom 数据库实例，请完成以下步骤：

1. 使用控制台或 CLI 在指定的时间内暂停 RDS Custom 自动化
2. 识别您的底层 Amazon EC2 实例。
3. 通过使用 SSH 密钥或 Amazon Systems Manager 连接到底层 Amazon EC2 实例。
4. 在数据库或操作系统层验证您的当前配置设置。

   您可以通过将初始配置与更改的配置进行比较来验证您的更改。根据自定义类型，使用操作系统工具或数据库查询。
5. 根据需要自定义 RDS Custom for Oracle 数据库实例。
6. 如果需要，重启您的实例或数据库。

> **Note**
>
> 在本地 Oracle CDB 中，您可以使用内置命令或在启动触发器之后为 PDB 保留指定的打开模式。当 CDB 重新启动时，此机制使 PDB 进入指定状态。打开 CDB 时，RDS Custom 自动化会丢弃任何用户指定的保留状态，并尝试打开所有 PDB。如果 RDS Custom 无法打开所有 PDB，则会发出以下事件：`The following PDBs failed to open: list-of-PDBs`。

7. 通过将您的新配置设置与以前的设置进行比较，对其进行验证。
8. 通过以下任一方式恢复 RDS Custom 自动化：
   - 手动恢复自动化。
   - 等待暂停期结束。在这种情况下，RDS Custom 会自动恢复监控和实例恢复。
9. 验证 RDS Custom 自动化框架

   如果您正确执行了上述步骤，RDS Custom 将启动自动备份。控制台中实例的状态显示为可用。

有关最佳实践和分步说明，请参阅 Amazon 博客文章[对 Amazon RDS Custom for Oracle 实例进行配置更改：第 1 部分]和[重新创建 Amazon RDS Custom for Oracle 数据库：第 2 部分]。

暂停和恢复 RDS Custom 数据库实例

您可以使用控制台或 CLI 暂停和恢复数据库实例的自动化。

控制台

要暂停或恢复 RDS Custom 自动化

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，请选择 Databases（数据库），然后选择要修改的 RDS Custom 数据库实例。
3. 选择修改。将显示修改数据库实例页面。
4. 对于 RDS Custom automation mode（RDS Custom 自动化模式），请选择以下选项之一：
   - Paused（已暂停）将暂停 RDS Custom 数据库实例的监控和实例恢复。为 Automation mode duration（自动化模式持续时间）输入您想要的暂停持续时间。最小值为 60 分钟（默认值）。最大值为 1440 分钟。

  - Full automation（完全自动化）将恢复自动化。
5. 请选择 Continue（继续），以查看修改摘要。

   将显示一条消息，表示 RDS Custom 将立即应用更改。
6. 如果更改正确无误，请选择 Modify DB Instance（修改数据库实例）。也可以选择 Back (返回) 编辑您的更改，或选择 Cancel (取消) 取消更改。

   在 RDS 控制台上，将显示修改的详细信息。如果您暂停了自动化，RDS Custom 数据库实例的 Status（状态）将指示 Automation paused（自动化已暂停）。
7. （可选）在导航窗格中，选择 Databases (数据库)，然后选择 RDS Custom 数据库实例。

   在 Summary（摘要）窗格中，RDS Custom automation mode（RDS Custom 自动化模式）表示自动化状态。如果自动化已暂停，则值为 Paused（已暂停）。自动化将在 ***num*** 分钟内恢复。

Amazon CLI

要暂停或恢复 RDS Custom 自动化，请使用 `modify-db-instance` Amazon CLI 命令。使用必需的参数 `--db-instance-identifier` 识别数据库实例。使用以下参数控制自动化模式：

- `--automation-mode` 指定数据库实例的暂停状态。有效值为 `all-paused`，它将暂停自动化，以及 `full`，它将恢复自动化。
- `--resume-full-automation-mode-minutes` 指定暂停的持续时间。默认值为 60 分钟。

> **Note**
>
> 无论您是指定 `--no-apply-immediately` 还是 `--apply-immediately`，RDS Custom 都会尽快地异步应用修改。

在命令响应中，`ResumeFullAutomationModeTime` 以 UTC 时间戳表示恢复时间。当自动化模式为 `all-paused` 时，您可以使用 `modify-db-instance` 恢复自动化模式或延长暂停时间。不支持其他 `modify-db-instance` 选项。

以下示例将 `my-custom-instance` 的自动化暂停 90 分钟。

Example

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier my-custom-instance \
    --automation-mode all-paused \
    --resume-full-automation-mode-minutes 90
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier my-custom-instance ^
    --automation-mode all-paused ^
    --resume-full-automation-mode-minutes 90
```

以下示例将暂停持续时间再延长 30 分钟。在 `ResumeFullAutomationModeTime` 中显示的原始时间中增加 30 分钟。

Example

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier my-custom-instance \
    --automation-mode all-paused \
    --resume-full-automation-mode-minutes 30
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier my-custom-instance ^
    --automation-mode all-paused ^
    --resume-full-automation-mode-minutes 30
```

以下示例恢复 `my-custom-instance` 的完全自动化。

Example

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier my-custom-instance \
```

```
    --automation-mode full \
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier my-custom-instance ^
    --automation-mode full
```

在以下部分示例输出中，待处理 `AutomationMode` 值为 `full`。

```
{
    "DBInstance": {
        "PubliclyAccessible": true,
        "MasterUsername": "admin",
        "MonitoringInterval": 0,
        "LicenseModel": "bring-your-own-license",
        "VpcSecurityGroups": [
            {
                "Status": "active",
                "VpcSecurityGroupId": "0123456789abcdefg"
            }
        ],
        "InstanceCreateTime": "2020-11-07T19:50:06.193Z",
        "CopyTagsToSnapshot": false,
        "OptionGroupMemberships": [
            {
                "Status": "in-sync",
                "OptionGroupName": "default:custom-oracle-ee-19"
            }
        ],
        "PendingModifiedValues": {
            "AutomationMode": "full"
        },
        "Engine": "custom-oracle-ee",
        "MultiAZ": false,
        "DBSecurityGroups": [],
        "DBParameterGroups": [
            {
                "DBParameterGroupName": "default.custom-oracle-ee-19",
                "ParameterApplyStatus": "in-sync"
            }
        ],
        ...
```

```
        "ReadReplicaDBInstanceIdentifiers": [],
        "AllocatedStorage": 250,
        "DBInstanceArn": "arn:aws:rds:us-west-2:012345678912:db:my-custom-instance",
        "BackupRetentionPeriod": 3,
        "DBName": "ORCL",
        "PreferredMaintenanceWindow": "fri:10:56-fri:11:26",
        "Endpoint": {
            "HostedZoneId": "ABCDEFGHIJKLMNO",
            "Port": 8200,
            "Address": "my-custom-instance.abcdefghijk.us-west-2.rds.amazonaws.com"
        },
        "DBInstanceStatus": "automation-paused",
        "IAMDatabaseAuthenticationEnabled": false,
        "AutomationMode": "all-paused",
        "EngineVersion": "19.my_cev1",
        "DeletionProtection": false,
        "AvailabilityZone": "us-west-2a",
        "DomainMemberships": [],
        "StorageType": "gp2",
        "DbiResourceId": "db-ABCDEFGHIJKLMNOPQRSTUVW",
        "ResumeFullAutomationModeTime": "2020-11-07T20:56:50.565Z",
        "KmsKeyId": "arn:aws:kms:us-west-2:012345678912:key/
aa111a11-111a-11a1-1a11-1111a11a1a1a",
        "StorageEncrypted": false,
        "AssociatedRoles": [],
        "DBInstanceClass": "db.m5.xlarge",
        "DbInstancePort": 0,
        "DBInstanceIdentifier": "my-custom-instance",
        "TagList": []
    }
```

修改 RDS Custom for Oracle 数据库实例

修改 RDS Custom for Oracle 数据库实例与修改 Amazon RDS 数据库实例类似。您可以更改如下设置：

- 数据库实例类
- 存储分配和类型
- 备份保留期
- 删除保护
- 选项组

- CEV（请参阅升级 RDS Custom for Oracle 数据库实例）
- 端口

主题

修改数据库实例存储时的要求和限制

当您修改 RDS Custom for Oracle 数据库实例的存储时，应考虑以下要求和限制：

- RDS Custom for Oracle 的最低分配存储空间为 40GiB，最大值为 64TiB。
- 与 Amazon RDS 一样，您无法减少分配的存储。这是对 Amazon EBS 卷的限制。
- RDS Custom 数据库实例不支持存储弹性伸缩。
- 手动附加到 RDS Custom 数据库实例的任何存储卷都在支持外围之外。

  有关更多信息，请参阅 RDS Custom 支持外围。
- RDS Custom 不支持磁性（标准）Amazon EBS 存储。只能选择 io1、gp2 或 gp3 SSD 存储类型。

有关 Amazon EBS 存储的更多信息，请参阅Amazon RDS 数据库实例存储。有关存储修改的一般信息，请参阅使用 Amazon RDS 数据库实例的存储。

修改数据库实例类时的要求和限制

当您修改 RDS Custom for Oracle 数据库实例的实例类时，应考虑以下要求和限制：

- 数据库实例必须处于 `available` 状态。
- 您的数据库实例在根卷、数据卷和二进制卷上必须至少有 100 MiB 的可用空间。
- 使用默认弹性网络接口（ENI）时，您只能为 RDS Custom for Oracle 数据库实例分配单个弹性 IP（EIP）。如果您将多个 ENI 连接到数据库实例，修改操作将失败。
- 所有 RDS Custom for Oracle 标签都必须存在。
- 如果您使用 RDS Custom for Oracle 复制，请注意以下要求和限制：

- 对于主数据库实例和只读副本，您一次只能更改一个数据库实例的实例类。
- 如果您的 RDS Custom for Oracle 数据库实例具有本地主数据库或副本数据库，请确保在修改完成后手动更新本地数据库实例上的私有 IP 地址。此操作是保留 Oracle DataGuard 功能所必需的。修改成功后，RDS Custom for Oracle 会发布事件。
- 当主数据库实例或只读副本数据库实例配置了 FSFO（快速启动失效转移）时，您无法修改 RDS Custom for Oracle 数据库实例类。

修改实例类时 RDS Custom 如何创建数据库实例

当您修改实例类时，RDS Custom 将按如下方式创建数据库实例：

- 创建 Amazon EC2 实例。
- 从最新的数据库快照创建根卷。RDS Custom for Oracle 不保留在最新数据库快照之后添加到根卷的信息。
- 创建 Amazon CloudWatch 告警。
- 如果删除了原始密钥对，则创建 Amazon EC2 SSH 密钥对。否则，RDS Custom for Oracle 保留原始密钥对。
- 使用启动修改时附加到数据库实例的标签创建新资源。当新资源直接附加到底层资源时，RDS Custom 不会将标签传输到新资源。
- 将包含最新修改的二进制和数据卷传输到新数据库实例。
- 传输弹性 IP 地址（EIP）。如果数据库实例可公开访问，则在传输 EIP 之前，RDS Custom 会临时将公有 IP 地址附加到新的数据库实例。如果数据库实例不可供公开访问，则 RDS Custom 不创建公有 IP 地址。

修改 RDS Custom for Oracle 数据库实例

您可以使用控制台、Amazon CLI 或 RDS API 修改数据库实例类或存储。

控制台

修改 RDS Custom for Oracle 数据库实例

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。
3. 选择要修改的数据库实例。

4. 选择修改。
5. （可选）在实例配置中，为数据库实例类选择一个值。有关支持的类，请参阅 RDS Custom for Oracle 的数据库实例类支持。
6. （可选）在存储中，根据需要进行以下更改：
   a. 为 Allocated Storage (分配的存储) 输入新值。它必须大于当前值，且范围从 40GiB 到 64TiB。
   b. 将存储类型的值更改为通用型 SSD（gp2）、通用型 SSD（gp3）或预调配 IOPS（io1）。
   c. 如果您使用预调配 IOPS（io1）或通用型 SSD（gp3），则可以更改预调配 IOPS 值。
7. （可选）对于其它配置，根据需要执行以下更改：
   - 对于选项组，选择新的选项组。有关更多信息，请参阅 使用 RDS Custom for Oracle 中的选项组。
8. 选择 Continue (继续)。
9. 选择 Apply immediately（立即应用）或 Apply during the next scheduled maintenance window（在下一个计划的维护时段内应用）。
10. 选择修改数据库实例。

Amazon CLI

要修改 RDS Custom for Oracle 数据库实例的存储，请使用 modify-db-instance Amazon CLI 命令。根据需要设置以下参数：

- `--db-instance-class` – 一个新的实例类。有关支持的类，请参阅 RDS Custom for Oracle 的数据库实例类支持。
- `--allocated-storage` – 要为数据库实例分配的存储量（以 GiB 为单位）。它必须大于当前值，并且范围从 40 到 65536GiB。
- `--storage-type` – 存储类型，gp2、gp3 或 io1。
- `--iops` – 如果使用 io1 或 gp3 存储类型，则为数据库实例的预调配 IOPS。
- `--apply-immediately` – 使用 `--apply-immediately` 可立即应用存储更改。

  或使用 `--no-apply-immediately`（默认设置）以在下一个维护时段内应用更改。

以下示例将 my-cfo-instance 的数据库实例类更改为 db.m5.16xlarge。该命令还将存储大小更改为 1TiB，将存储类型更改为 io1，将预调配 IOPS 更改为 3000，并将选项组更改为 cfo-ee-19-mt。

Example

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier my-cfo-instance \
    --db-instance-class db.m5.16xlarge \
    --storage-type io1 \
    --iops 3000 \
    --allocated-storage 1024 \
    --option-group cfo-ee-19-mt \
    --apply-immediately
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier my-cfo-instance ^
    --db-instance-class db.m5.16xlarge ^
    --storage-type io1 ^
    --iops 3000 ^
    --allocated-storage 1024 ^
    --option-group cfo-ee-19-mt ^
    --apply-immediately
```

更改 RDS Custom for Oracle 数据库实例的字符集

RDS Custom for Oracle 默认使用字符集 US7ASCII。您可能需要指定不同的字符集以满足语言或多字节字符要求。使用 RDS Custom for Oracle 时，可以暂停自动化，然后手动更改数据库的字符集。

更改 RDS Custom for Oracle 数据库实例的字符集具有以下要求：

- 您只能更改新预调配的 RDS 自定义实例上的字符，该实例具有空数据库或初学者数据库，其中没有应用程序数据。对于所有其他情况，请使用 DMU（Unicode 的数据库迁移助手）更改字符集。
- 您只能更改为 RDS for Oracle 支持的字符集。有关更多信息，请参阅 支持的数据库字符集。

更改 RDS Custom for Oracle 数据库实例的字符集

1. 暂停 RDS Custom 自动化。有关更多信息，请参阅 暂停和恢复 RDS Custom 数据库实例。
2. 以具有 `SYSDBA` 权限的用户身份登录数据库。
3. 在受限模式下重新启动数据库，更改字符集，然后以正常模式重新启动数据库。

在 SQL 客户端中运行以下脚本：

```
SHUTDOWN IMMEDIATE;
STARTUP RESTRICT;
ALTER DATABASE CHARACTER SET INTERNAL_CONVERT AL32UTF8;
SHUTDOWN IMMEDIATE;
STARTUP;
SELECT VALUE FROM NLS_DATABASE_PARAMETERS WHERE PARAMETER = 'NLS_CHARACTERSET';
```

验证输出是否显示了正确的字符集：

```
VALUE
--------
AL32UTF8
```

4. 恢复 RDS Custom 自动化。有关更多信息，请参阅 暂停和恢复 RDS Custom 数据库实例。

在 RDS Custom for Oracle 中设置 NLS_LANG 值

区域设置 是一组满足语言和文化要求的信息，这些信息对应于给定的语言和国家/地区。要指定 Oracle 软件的区域设置行为，请在客户端主机上设置 NLS_LANG 环境变量。此变量设置客户端应用程序在数据库会话中使用的语言、区域和字符集。

对于 RDS Custom for Oracle，在 NLS_LANG 变量中只能设置语言：区域和字符使用默认值。语言用于 Oracle 数据库消息、排序规则、日期名称和月份名称。每种支持的语言都有一个唯一名称，例如美语、法语或德语。如果未指定语言，则值默认为美语。

创建 RDS Custom for Oracle 数据库后，可以在客户端主机上将 NLS_LANG 设置为英语以外的其他语言。要查看 Oracle 数据库支持的语言列表，请登录 RDS Custom for Oracle 数据库并运行以下查询：

```
SELECT VALUE FROM V$NLS_VALID_VALUES WHERE PARAMETER='LANGUAGE' ORDER BY VALUE;
```

可以在主机命令行上设置 NLS_LANG。以下示例使用 Linux 上的 Z Shell 将客户端应用程序的语言设置为德语。

```
export NLS_LANG=German
```

您的应用程序在启动时读取 NLS_LANG 值，然后在连接时将其传送给数据库。

有关更多信息，请参阅《Oracle 数据库全球化支持指南》中的使用 NLS_LANG 环境变量选择区域设置。

支持透明数据加密

RDS Custom 支持 RDS Custom for Oracle 数据库实例的透明数据加密 (TDE)。

但是，您不能像在 RDS for Oracle 中那样使用自定义选项组中的选项启用 TDE。您可以手动打开 TDE。有关使用 Oracle 透明数据加密的信息，请参阅 Oracle 文档中的使用透明数据加密保护存储数据安全。

为 RDS Custom for Oracle 资源添加标签

您可以将 RDS Custom 资源与 Amazon RDS 资源一样标记，但有一些重要的区别：

- 不要创建或修改 RDS Custom 自动化所需的 `AWSRDSCustom` 标签。如果您这样做，可能会破坏自动化。
- `Name` 标签将添加到 RDS Custom 资源，前缀值为 `do-not-delete-rds-custom` 或 `rds-custom!oracle-do-not-delete`。任何客户传递的键值都将被覆盖。
- 在创建过程中添加到 RDS Custom 数据库实例的标签将传播到所有其他相关 RDS Custom 资源。
- 当您在创建数据库实例后将标签添加到 RDS Custom 资源时，标签不会传播。

有关资源标记的一般信息，请参阅为 Amazon RDS 资源添加标签。

删除 RDS Custom for Oracle 数据库实例

要删除 RDS Custom 数据库实例，请执行以下操作：

- 提供数据库实例的名称。
- 清除拍摄数据库实例的最终数据库快照的选项。
- 选择或清除保留自动备份的选项。

您可以使用控制台或 CLI 删除 RDS Custom 数据库实例。删除数据库实例所需的时间可能会根据备份保留期（也即，要删除的备份数）以及删除的数据量而有所不同。

控制台

要删除 RDS Custom 数据库实例

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)，然后选择要删除的 RDS Custom 数据库实例。RDS Custom 数据库实例显示角色 Instance (RDS Custom)（实例 (RDS Custom)）。
3. 对于操作，选择删除。
4. 要保留自动备份，请选择 Retain automated backups (保留自动备份)。
5. 在框中输入 **`delete me`**。
6. 选择删除。

Amazon CLI

您可以通过使用 delete-db-instance Amazon CLI 命令删除 RDS Custom 数据库实例。使用必需的参数 `--db-instance-identifier` 识别数据库实例。其余参数与 Amazon RDS 数据库实例的参数相同，但以下各项除外：

- `--skip-final-snapshot` 是必需的。
- `--no-skip-final-snapshot` 不受支持。
- `--final-db-snapshot-identifier` 不受支持。

以下示例删除名为 `my-custom-instance` 的 RDS Custom 数据库实例，并保留自动备份。

Example

对于 Linux、macOS 或 Unix：

```
aws rds delete-db-instance \
    --db-instance-identifier my-custom-instance \
    --skip-final-snapshot \
    --no-delete-automated-backups
```

对于 Windows：

```
aws rds delete-db-instance ^
```

```
    --db-instance-identifier my-custom-instance ^
    --skip-final-snapshot ^
    --no-delete-automated-backups
```

使用适用于 RDS Custom for Oracle 的 Oracle 副本

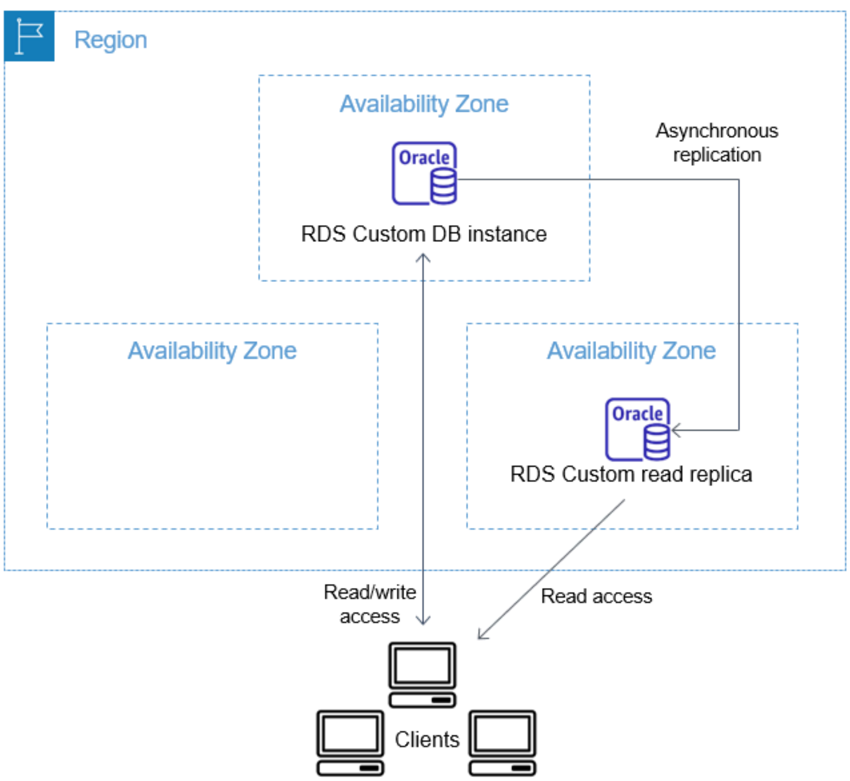
您可以为运行 Oracle Enterprise Edition 的 RDS Custom for Oracle 数据库实例创建 Oracle 副本。支持容器数据库（CDB）和非 CDB。Standard Edition 2 不支持 Oracle Data Guard。

创建 RDS Custom for Oracle 副本类似于创建 RDS for Oracle 副本，但有重大区别。有关创建和管理 Oracle 副本的一般信息，请参阅使用数据库实例只读副本和使用 Amazon RDS for Oracle 的只读副本。

主题

RDS Custom for Oracle 复制概述

RDS Custom for Oracle 复制的架构类似于 RDS for Oracle 复制。主数据库实例异步复制到一个或多个 Oracle 副本。



最大副本数

与 RDS for Oracle 一样，您最多可以创建 RDS Custom for Oracle 主数据库实例的五个托管式 Oracle 副本。您还可以创建自己的手动配置的（外部）Oracle 副本。外部副本不计入您的数据库实例限制。它们还位于 RDS Custom 支持外围之外。有关支持外围的更多信息，请参阅 RDS Custom 支持外围。

副本命名约定

Oracle 副本名称基于数据库的唯一名称。格式为 *DB_UNIQUE_NAME*_*X*，并按顺序附加字母。例如，如果您的数据库唯一名称为 `ORCL`，则前两个副本命名为 `ORCL_A` 和 `ORCL_B`。前六个字母 A–F 是为 RDS Custom 保留的。RDS Custom 将数据库参数从主数据库实例复制到副本。有关更多信息，请参阅 Oracle 文档中的 DB_UNIQUE_NAME。

副本备份保留期

原定设置情况下，RDS Custom Oracle 副本使用与主数据库实例相同的备份保留期。您可以将备份保留期修改为 1–35 天。RDS Custom 支持备份、还原和时间点故障恢复（PITR）。有关备份和还原 RDS Custom 数据库实例的更多信息，请参阅备份和还原 Amazon RDS Custom for Oracle 数据库实例。

> **Note**
>
> 当创建 Oracle 副本时，RDS Custom 会暂时暂停清理重做日志文件。通过这种方式，RDS Custom 可确保在新的 Oracle 副本变为可用后将这些日志应用于该副本。

副本提升

您可以使用控制台、`promote-read-replica` Amazon CLI 命令或 `PromoteReadReplica` API 在 RDS Custom for Oracle 中提升托管式 Oracle 副本。如果您删除了主数据库实例，并且所有副本都运行正常，则 RDS Custom for Oracle 会自动将您的托管式副本提升为独立实例。如果副本已暂停自动化或超出支持外围，则必须先修复副本，RDS Custom 才能自动对其进行提升。您只能手动提升外部 Oracle 副本。

RDS Custom for Oracle 复制的指南和限制

创建 RDS Custom for Oracle 副本时，并非所有 RDS Oracle 副本选项都受支持。

主题

- RDS Custom for Oracle 复制的一般指南
- RDS Custom for Oracle 复制的一般限制
- RDS Custom for Oracle 复制的网络要求和限制
- RDS Custom for Oracle 的外部副本限制

RDS Custom for Oracle 复制的一般指南

使用 RDS Custom for Oracle 时，请遵循以下指南：

- 您只能在 Oracle Enterprise Edition 中使用 RDS Custom for Oracle 复制。不支持 Standard Edition 2。
- 强烈建议您实施 VPN 隧道来加密主实例和备用实例之间的通信。有关更多信息，请参阅 在 RDS Custom for Oracle 主实例和副本实例之间配置 VPN 隧道。
- 不要修改 `RDS_DATAGUARD` 用户。此用户保留用于 RDS Custom for Oracle 自动化。修改此用户可能会导致不希望的结果，例如无法为 RDS Custom for Oracle 数据库实例创建 Oracle 副本。
- 不要更改复制用户密码。管理 RDS Custom 主机上的 Oracle Data Guard 配置需要使用此密码。如果您更改密码，RDS Custom for Oracle 可能会将您的 Oracle 副本置于支持外围之外。有关更多信息，请参阅 RDS Custom 支持外围。

  此密码存储在 Amazon Secrets Manager 中，用数据库资源 ID 进行标记。每个 Oracle 副本在 Secret Manager 中都有自己的密钥。密钥使用以下任一命名格式。

  ```
  do-not-delete-rds-custom-db-DB_resource_id-uuid-dg
  rds-custom!oracle-do-not-delete-DB_resource_id-uuid-dg
  ```

- 不要更改主数据库实例的 `DB_UNIQUE_NAME`。更改名称会导致任何还原操作卡住。
- 不要在 RDS Custom CDB 的 `CREATE PLUGGABLE DATABASE` 命令中指定子句 `STANDBYS=NONE`。这样，如果发生失效转移，您的备用 CDB 将包含所有 PDB。

RDS Custom for Oracle 复制的一般限制

RDS Custom for Oracle 副本具有以下限制：

- 无法在只读模式下创建 RDS Custom for Oracle 副本。但是，您可以将已挂载副本的模式手动更改为只读，以及从只读更改为已挂载。有关更多信息，请参阅 create-db-instance-read-replica Amazon CLI 命令的文档。

- 您无法创建跨区域 RDS Custom for Oracle 副本。
- 您无法更改 Oracle Data Guard `CommunicationTimeout` 参数的值。对于 RDS Custom for Oracle 数据库实例，此参数设置为 15 秒。

RDS Custom for Oracle 复制的网络要求和限制

确保您的网络配置支持 RDS Custom for Oracle 副本。请考虑以下事项：

- 确保为主数据库实例和其所有副本的虚拟私有云（VPC）内的入站和出站通信启用端口 1140。这对于只读副本之间的 Oracle Data Guard 通信是必需的。
- RDS Custom for Oracle 会在创建 Oracle 副本时对网络进行验证。如果主数据库实例和新副本无法通过网络连接，则 RDS Custom for Oracle 不会创建副本并将其置于 `INCOMPATIBLE_NETWORK` 状态。
- 对于外部 Oracle 副本，例如您在 Amazon EC2 或本地创建的只读副本，请使用另一个端口和侦听器进行 Oracle Data Guard 复制。尝试使用端口 1140 可能会导致与 RDS Custom 自动化发生冲突。
- `/rdsdbdata/config/tnsnames.ora` 文件包含映射到侦听器协议地址的网络服务名称。请注意以下要求和建议：
  - 当处理 Oracle 副本操作时，`tnsnames.ora` 中前缀为 `rds_custom_` 的条目保留用于 RDS Custom。

    在 `tnsnames.ora` 中创建手动条目时，不要使用此前缀。
  - 在某些情况下，您可能希望手动切换或进行失效转移，或者使用快速启动失效转移 (FSFO) 等失效转移技术。如果是这样，请确保将 `tnsnames.ora` 条目从主数据库实例手动同步到所有备用实例。此建议同时适用于 RDS Custom 管理的 Oracle 副本和外部 Oracle 副本。

    RDS Custom 自动化只会更新主数据库实例上的 `tnsnames.ora` 条目。请务必在添加或删除 Oracle 副本时也要进行同步。

    如果您不同步 `tnsnames.ora` 文件并且手动切换或失效转移，主数据库实例上的 Oracle Data Guard 可能无法与 Oracle 副本进行通信。

RDS Custom for Oracle 的外部副本限制

RDS Custom for Oracle 外部副本（包括本地副本）具有以下限制：

- RDS Custom for Oracle 在手动失效转移（如 FSFO）时不会检测到外部 Oracle 副本的实例角色更改。

RDS Custom for Oracle 的确会检测托管式副本的更改。角色更改在事件日志中进行了记录。您还可以通过使用 describe-db-instances Amazon CLI 命令查看新状态。

- RDS Custom for Oracle 不检测到外部 Oracle 副本的高复制滞后。

  RDS Custom for Oracle 的确会检测托管式副本的滞后。高复制滞后会产生 `Replication has stopped` 事件。您还可以通过使用 describe-db-instances Amazon CLI 命令查看复制状态，但它的更新可能会延迟。

- 如果您删除主数据库实例，RDS Custom for Oracle 不会自动提升外部 Oracle 副本。

  自动提升功能仅适用于托管式 Oracle 副本。有关手动提升 Oracle 副本的信息，请参阅白皮书在 Amazon RDS Custom for Oracle 上使用 Data Guard 启用高可用性。

将 RDS Custom for Oracle 副本提升为独立的数据库实例

就像使用 RDS for Oracle 一样，您可以将 RDS Custom for Oracle 副本提升为独立的数据库实例。提升 Oracle 副本时，RDS Custom for Oracle 会在数据库实例变为可用之前重新启动数据库实例。有关提升 Oracle 副本的更多信息，请参阅将只读副本提升为独立的数据库实例。

提升副本时，请注意以下指南：

- 在 RDS Custom for Oracle 提升您的副本时不要启动失效转移。否则，提升工作流可能会卡住。
- 在 RDS Custom for Oracle 提升 Oracle 副本时，不要切换主数据库实例。否则，提升工作流可能会卡住。
- 在 RDS Custom for Oracle 提升 Oracle 副本时，不要关闭主数据库实例。否则，提升工作流可能会卡住。
- 不要尝试将新提升的数据库实例作为目标来重新开始复制。在 RDS Custom for Oracle 提升 Oracle 副本后，它将成为一个独立的数据库实例，不再具有副本角色。

注意 RDS Custom for Oracle 副本提升存在以下限制：

- 当 RDS Custom for Oracle 正在备份副本时，您无法提升副本。
- 提升 Oracle 副本时，不能将备份保留期更改为 `0`。
- 当副本未处于正常运行状态时，您无法提升它。

  如果您在主数据库实例上发出 `delete-db-instance`，RDS Custom for Oracle 会验证每个托管式 Oracle 副本是否正常运行以及是否可供提升。副本可能不符合提升资格，因为自动化已暂停或它不

在支持外围内。在这种情况下，RDS Custom for Oracle 会发布一个事件来解释此问题，以便您可以手动修复 Oracle 副本。

以下步骤说明将 Oracle 副本提升为数据库实例的一般过程：

1. 停止将任何事务写入到主数据库实例。
2. 等待 RDS Custom for Oracle 将所有更新应用到 Oracle 副本。
3. 通过在 Amazon RDS 控制台上选择 Promote（提升）选项、Amazon CLI 命令 `promote-read-replica` 或 `PromoteReadReplica` Amazon RDS API 操作来提升 Oracle 副本。

提升一个 Oracle 副本需要几分钟才能完成。在此过程中，RDS Custom for Oracle 会停止复制并重启您的副本。完成重启后，Oracle 副本即可用作独立的数据库实例。有关排查副本提升问题的信息，请参阅排查 RDS Custom for Oracle 的副本提升问题。

控制台

将 RDS Custom for Oracle 副本提升为独立的数据库实例

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在 Amazon RDS 控制台中，选择数据库。

   随后会显示数据库窗格。每个 Oracle 副本在 Role（角色）列中显示 Replica（副本）。
3. 选择您要提升的 RDS Custom for Oracle 副本。
4. 对于操作，请选择提升。
5. 在 Promote Oracle replica（提升 Oracle 副本）页面上，输入新提升的数据库实例的备份保留期和备份时段。您不能将此值设置为 0。
6. 根据需要完成设置后，选择 Promote Oracle replica（提升 Oracle 副本）。

Amazon CLI

要将 RDS Custom for Oracle 副本提升为独立的数据库实例，请使用 Amazon CLI `promote-read-replica` 命令。

Example

对于 Linux、macOS 或 Unix：

```
aws rds promote-read-replica \
--db-instance-identifier my-custom-read-replica \
--backup-retention-period 2 \
--preferred-backup-window 23:00-24:00
```

对于 Windows：

```
aws rds promote-read-replica ^
--db-instance-identifier my-custom-read-replica ^
--backup-retention-period 2 ^
--preferred-backup-window 23:00-24:00
```

RDS API

要将 RDS Custom for Oracle 副本提升为独立的数据库实例，请调用具有必要参数 `PromoteReadReplica` 的 Amazon RDS API [`DBInstanceIdentifier`](#) 操作。

在 RDS Custom for Oracle 主实例和副本实例之间配置 VPN 隧道

VPN 隧道是通过网络在两个或更多个设备之间建立的加密连接。为了确保 RDS Custom for Oracle 中的 Oracle Data Guard 实例具有最高级别的安全性，我们强烈建议您实施 VPN 隧道来对主实例与备用实例之间的通信进行加密。该隧道在实例之间进行网络传输时，可保护敏感数据。虽然此配置是可选的，但我们建议将其作为实现数据安全性和法规合规性的最佳实践。

请确保满足以下先决条件：

- 您拥有主要主机和备用主机的根访问权限。
- 您拥有运行 `ipsec` 命令的技术专业知识。

在 RDS Custom for Oracle 中配置主副本之间的 VPN 隧道

1. 使用以下规则将主实例和备用实例的安全组添加到允许列表中：

    ```
    ACTION FLOW SOURCE PROTO PORT

    ALLOW ingress this-SG 50 (ESP) all (N/A)
    ALLOW egress this-SG 50 (ESP) all (N/A)

    ALLOW ingress this-SG 17 (UDP) 500 (IKE)
    ```

```
ALLOW egress this-SG 17 (UDP) 500 (IKE)
```

2. 切换到根用户。

```
$ sudo su – root
```

3. 在主实例和备用实例上运行以下命令，以初始化用户 `root` 下的网络安全服务（NSS）数据库。

```
ipsec initnss --nssdir /etc/ipsec.d
```

4. 按如下方式生成 RSA 密钥：

   a. 在主实例上，使用以下任一 `ipsec` 命令生成密钥，具体取决于您的操作系统版本。

   ```
   ipsec newhostkey --nssdir /etc/ipsec.d       ## for Oracle Linux Version 8
   ipsec newhostkey --output /etc/ipsec.secrets ## for Oracle Linux version 7.9
   ```

   b. 获取创建配置所需的公有密钥。在以下示例中，主实例是 `left`，因为在 `ipsec` 术语中，`left` 是指您当前正在配置的设备，而 `right` 是指隧道另一端的设备。

   ```
   ipsec showhostkey --left --ckaid ckaid-returned-in-last-statement
   ```

   c. 在备用实例上，为备用实例生成密钥。

   ```
   ipsec newhostkey --nssdir /etc/ipsec.d       ## for Oracle Linux Version 8
   ipsec newhostkey --output /etc/ipsec.secrets ## for Oracle Linux version 7.9
   ```

   d. 获取备用实例的公有密钥，创建配置时需要该公有密钥。在以下示例中，备用实例为 `right`，因为它指的是隧道另一端的设备。

   ```
   ipsec showhostkey --right --ckaid ckaid-returned-in-last-statement
   ```

5. 根据您获得的 RSA 密钥生成配置。主实例与备用实例的配置相同。您可以在 Amazon 控制台中找到主实例 IPv4 地址和备用实例 IPv4 地址。

   在主实例和备用实例上，将以下配置保存到文件 `/etc/ipsec.d/custom-fb-tunnel.conf` 中。

```
conn custom-db-tunnel
 type=transport
 auto=add
 authby=rsasig
```

```
 left=IPV4-for-primary
 leftrsasigkey=RSA-key-generated-on-primary
 right=IPV4-for-standby
 rightrsasigkey=RSA-key-generated-on-standby
```

6. 在主实例和备用实例上，在两台主机上启动 `ipsec` 进程守护程序。

```
ipsec setup start
```

7. 在主实例或备用实例上启动隧道。 输出应类似于以下内容。

```
[root@ip-172-31-6-81 ~]# ipsec auto --up custom-db-tunnel
181 "custom-db-tunnel" #1: initiating IKEv2 connection
181 "custom-db-tunnel" #1: sent IKE_SA_INIT request to 172.31.32.196:500
182 "custom-db-tunnel" #1: sent IKE_AUTH request {cipher=AES_GCM_16_256 integ=n/a
 prf=HMAC_SHA2_512 group=DH19}
003 "custom-db-tunnel" #1: initiator established IKE SA; authenticated peer '3584-
bit PKCS#1 1.5 RSA with SHA1' signature using preloaded certificate '172.31.32.196'
004 "custom-db-tunnel" #2: initiator established Child SA using #1; IPsec transport
 [172.31.6.81-172.31.6.81:0-65535 0] -> [172.31.32.196-172.31.32.196:0-65535 0]
 {ESP/ESN=>0xda9c4815 <0xb742ca42 xfrm=AES_GCM_16_256-NONE DPD=passive}
[root@ip-172-31-6-81 ~]#
```

备份和还原 Amazon RDS Custom for Oracle 数据库实例

与 Amazon RDS 相同，RDS Custom 会在数据库实例的备份时段中，创建并保存 RDS Custom for Oracle 数据库实例的自动备份。您还可以手动备份数据库实例。

此过程与拍摄 Amazon RDS 数据库实例的快照相同。RDS Custom 数据库实例的第一个快照包含完整数据库实例的数据。后续快照为增量快照。

使用 Amazon Web Services Management Console 或 Amazon CLI 还原数据库快照。

主题

创建 RDS Custom for Oracle 快照

RDS Custom for Oracle 可创建数据库实例的存储卷快照，并备份整个数据库实例而不仅仅是单个数据库。当您的数据库实例包含容器数据库（CDB）时，该实例的快照包括根 CDB 和所有 PDB。

创建 RDS Custom for Oracle 快照时，请指定要备份哪个 RDS Custom 数据库实例。为您的快照命名，以便稍后通过它进行还原。

当您创建快照时，RDS Custom for Oracle 会为附加到数据库实例的每个卷创建 Amazon EBS 快照。RDS Custom for Oracle 使用根卷的 EBS 快照注册新的 Amazon Machine Image（AMI）。为了使快照轻松与特定数据库实例关联，请使用 `DBSnapshotIdentifier`、`DbiResourceId` 和 `VolumeType` 标记它们。

创建数据库快照会导致短暂的输入/输出暂停。此暂停可持续数秒到几分钟，具体取决于数据库实例的大小和类。快照创建时间因数据库的大小而异。由于快照包含整个存储卷，因此，文件（如临时文件）的大小也会影响创建快照创建时间。要了解创建快照的更多信息，请参阅 为 Amazon RDS 的单可用区数据库实例创建数据库快照。

使用控制台或 Amazon CLI 创建 RDS Custom for Oracle 快照。

控制台

创建 RDS Custom 快照

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases（数据库）。
3. 在 RDS Custom 数据库实例列表中，选择您想拍摄快照的实例。
4. 对于操作，选择拍摄快照。

   将显示 Take DB snapshot (拍摄数据库快照) 窗口。
5. 在 Snapshot name（快照名称）框中，请输入快照的名称。
6. 选择拍摄快照。

Amazon CLI

您可以通过使用 create-db-snapshot Amazon CLI 命令创建 RDS Custom 数据库实例的快照。

指定以下选项：

- `--db-instance-identifier` – 标识要备份的 RDS Custom 数据库实例
- `--db-snapshot-identifier` – 命名 RDS Custom 快照，以便以后通过它进行还原

在此示例中，您将给名为 *my-custom-instance* 的 RDS Custom 数据库实例创建名为 *my-custom-snapshot* 的数据库快照。

Example

对于 Linux、macOS 或 Unix：

```
aws rds create-db-snapshot \
    --db-instance-identifier my-custom-instance \
    --db-snapshot-identifier my-custom-snapshot
```

对于 Windows：

```
aws rds create-db-snapshot ^
    --db-instance-identifier my-custom-instance ^
    --db-snapshot-identifier my-custom-snapshot
```

从 RDS Custom for Oracle 数据库快照还原

在还原 RDS Custom for Oracle 数据库实例时，需要提供数据库快照的名称以及新实例的名称。您不能从快照还原到现有的 RDS Custom 数据库实例。还原时将新建一个 RDS Custom for Oracle 数据库实例。

该还原过程与 Amazon RDS 中的还原有以下方式不同：

- 还原快照之前，RDS Custom for Oracle 会备份现有配置文件。这些文件提供在目录 `/rdsdbdata/config/backup` 中的还原实例上。RDS Custom for Oracle 会使用原定设置参数还原数据库快照，并用现有参数覆盖以前的数据库配置文件。因此，还原的实例不会保留自定义参数和对数据库配置文件的更改。
- 还原的数据库的名称与快照中的名称相同。您不能指定不同的名称。（对于 RDS Custom for Oracle，原定设置值为 `ORCL`。）

控制台

要从数据库快照还原 RDS Custom 数据库实例

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择快照。
3. 选择要从其还原的数据库快照。
4. 对于操作，选择还原快照。
5. 请在 Restore DB instance（还原数据库实例）页面上，为 DB instance identifier（数据库实例标识符）输入还原的 RDS Custom 数据库实例的名称。
6. 选择还原数据库实例。

Amazon CLI

您可以通过使用 restore-db-instance-from-db-snapshot Amazon CLI 命令还原 RDS Custom 数据库快照。

如果要从中还原的快照适用于私有数据库实例，请务必同时指定正确的 `db-subnet-group-name` 和 `no-publicly-accessible`。否则，数据库实例原定设置为可公开访问。以下选项为必填：

- `db-snapshot-identifier` – 标识要从中进行还原的快照

- db-instance-identifier – 指定要从数据库快照创建的 RDS Custom 数据库实例的名称
- custom-iam-instance-profile – 指定与 RDS Custom 数据库实例的基础 Amazon EC2 实例关联的实例配置文件。

以下代码为 my-custom-instance 还原名为 my-custom-snapshot 的快照。

Example

对于 Linux、macOS 或 Unix：

```
aws rds restore-db-instance-from-db-snapshot \
  --db-snapshot-identifier my-custom-snapshot \
  --db-instance-identifier my-custom-instance \
  --custom-iam-instance-profile AWSRDSCustomInstanceProfileForRdsCustomInstance \
  --no-publicly-accessible
```

对于 Windows：

```
aws rds restore-db-instance-from-db-snapshot ^
  --db-snapshot-identifier my-custom-snapshot ^
  --db-instance-identifier my-custom-instance ^
  --custom-iam-instance-profile AWSRDSCustomInstanceProfileForRdsCustomInstance ^
  --no-publicly-accessible
```

将 RDS Custom for Oracle 实例还原到某个时间点

您可以将数据库实例还原到某个特定时间点 (PITR)，以创建新数据库实例。要支持 PITR，您的数据库实例必须将备份保留期设置为非零值。

RDS Custom for Oracle 数据库实例的最近可还原时间取决于多个因素，但通常为当前时间的 5 分钟内。要查看某个数据库实例的最近可还原时间，请使用 Amazon CLI describe-db-instances 命令，并查看该数据库实例的 LatestRestorableTime 字段中返回的值。要在 Amazon RDS 控制台中查看每个数据库实例的最近可还原时间，请选择自动备份。

您可以还原至备份保留期内的任何时间点。要查看每个数据库实例的最早可还原时间，请在 Amazon RDS 控制台中选择自动备份。

有关 PITR 的一般信息，请参阅将 Amazon RDS 的数据库实例还原到指定时间。

主题

- RDS Custom for Oracle 的 PITR 注意事项

RDS Custom for Oracle 的 PITR 注意事项

在 RDS Custom for Oracle 中，PITR 与 Amazon RDS 中的 PITR 有以下重要的区别：

- 还原的数据库的名称与源数据库实例中的名称相同。您不能指定不同的名称。默认为 `ORCL`。
- `AWSRDSCustomIamRolePolicy` 需要新的权限。有关更多信息，请参阅 步骤 2：将访问策略添加到 AWSRDSCustomInstanceRoleForRdsCustomInstance 中。
- RDS Custom for Oracle 的所有数据库实例都必须将备份保留期设置为非零值。
- 如果更改操作系统或数据库实例时区，PITR 可能无法正常工作。有关更改时区的信息，请参阅Oracle 时区。
- 如果将自动化设置为 `ALL_PAUSED`，RDS Custom 会暂停上传存档的重做日志文件，包括在最近可还原时间（LRT）之前创建的日志。我们建议您短暂暂停自动化。

  为了进行说明，假设您的 LRT 为 10 分钟前。您暂停了自动化。暂停期间，RDS Custom 不上载存档的重做日志。如果数据库实例崩溃，则只能恢复到暂停时存在的 LRT 之前的一段时间。当您恢复自动化时，RDS Custom 会恢复上载日志。LRT 推进。将适用正常的 PITR 规则。
- 在 RDS Custom 中，您可以手动指定 RDS Custom 在上载后将存档的重做日志删除之前将其保留的任意小时数。按如下方式指定小时数：
  1. 创建名为 `/opt/aws/rdscustomagent/config/redo_logs_custom_configuration.json` 的文本文件。
  2. 按以下格式添加 JSON 对象：`{"archivedLogRetentionHours" : "`*`num_of_hours`*`"}`。该数字必须是一个介于 1–840 之间的整数。
- 假设您将非 CDB 作为 PDB 插入容器数据库（CDB），然后尝试 PITR。仅当您先前备份了 PDB 时，该操作才会成功。创建或修改 PDB 后，我们建议您始终对其进行备份。
- 我们建议您不要自定义数据库初始化参数。例如，修改以下参数会影响 PITR：
  - `CONTROL_FILE_RECORD_KEEP_TIME` 影响上传和删除日志的规则。
  - `LOG_ARCHIVE_DEST_`*`n`* 不支持多个目的地。
  - `ARCHIVE_LAG_TARGET` 影响最近可还原时间。`ARCHIVE_LAG_TARGET` 设置为 `300`，因为恢复点目标（RPO）为 5 分钟。为了实现这一目标，RDS 每 5 分钟切换一次在线重做日志，并将其存储在 Amazon S3 存储桶中。如果日志切换的频率导致 RDS Custom for Oracle 数据库出现性能问题，则可以将数据库实例和存储扩展到具有更高 IOPS 和吞吐量的数据库实例和存储。如果恢复计划需要，可以将 `ARCHIVE_LAG_TARGET` 初始化参数的设置调整为 60–7200 之间的值。
- 如果您自定义数据库初始化参数，我们强烈建议您只自定义以下内容：
  - `COMPATIBLE`

- MAX_STRING_SIZE
- DB_FILES
- UNDO_TABLESPACE
- ENABLE_PLUGGABLE_DATABASE
- CONTROL_FILES
- AUDIT_TRAIL
- AUDIT_TRAIL_DEST

对于所有其他初始化参数，RDS Custom 将还原原定设置值。如果修改不在前面列表中的参数，可能会对 PITR 产生不利影响，并导致不可预测的结果。例如，CONTROL_FILE_RECORD_KEEP_TIME 影响上传和删除日志的规则。

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 将 RDS Custom 数据库实例还原到某个时间点。

控制台

要将 RDS Custom 数据库实例还原至指定时间

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Automated backups（自动备份）。
3. 选择要还原的 RDS Custom 数据库实例。
4. 对于 Actions (操作)，选择 Restore to point in time (还原到时间点)。

   此时会显示还原到时间点窗口。
5. 选择最近可还原时间以还原到可能的最近时间，或选择自定义来选择时间。

   如果您选择 Custom（自定义），请输入要将实例集群还原到的日期和时间。

   时间以您的本地时区显示，表示为协调世界时 (UTC) 的偏移量。例如，UTC-5 是东部标准时间/中部夏令时。
6. 对于 DB instance identifier（数据库实例标识符），请输入目标还原后的 RDS Custom 数据库实例的名称。名称必须唯一。
7. 根据需要选择其他选项，例如数据库实例类。
8. 选择还原到时间点。

Amazon CLI

通过使用 restore-db-instance-to-point-in-time Amazon CLI 命令创建新的 RDS Custom 数据库实例，将数据库实例还原到指定时间。

使用以下选项之一指定要从中恢复的备份：

- `--source-db-instance-identifier` *`mysourcedbinstance`*
- `--source-dbi-resource-id` *`dbinstanceresourceID`*
- `--source-db-instance-automated-backups-arn` *`backupARN`*

`custom-iam-instance-profile` 选项是必需的。

截至指定的时间，以下示例会将 `my-custom-db-instance` 还原到名为 `my-restored-custom-db-instance` 的新数据库实例。

Example

对于 Linux、macOS 或 Unix：

```
aws rds restore-db-instance-to-point-in-time \
    --source-db-instance-identifier my-custom-db-instance\
    --target-db-instance-identifier my-restored-custom-db-instance \
    --custom-iam-instance-profile AWSRDSCustomInstanceProfileForRdsCustomInstance \
    --restore-time 2022-10-14T23:45:00.000Z
```

对于 Windows：

```
aws rds restore-db-instance-to-point-in-time ^
    --source-db-instance-identifier my-custom-db-instance ^
    --target-db-instance-identifier my-restored-custom-db-instance ^
    --custom-iam-instance-profile AWSRDSCustomInstanceProfileForRdsCustomInstance ^
    --restore-time 2022-10-14T23:45:00.000Z
```

删除 RDS Custom for Oracle 快照

如果不再需要，您可以删除 RDS Custom for Oracle 管理的数据库快照。对于 Amazon RDS 和 RDS Custom 数据库实例而言，删除过程是相同的。

二进制卷和根卷的 Amazon EBS 快照会在您的账户中保留更长时间，因为它们可能与您账户中运行的某些实例或其他 RDS Custom for Oracle 快照关联。当这些 EBS 快照不再与任何现有 RDS Custom for Oracle 资源（数据库实例或备份）相关后，将会被自动删除。

控制台

要删除 RDS Custom 数据库实例的快照

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择快照。
3. 选择要删除的数据库快照。
4. 对于 Actions（操作），选择 Delete snapshot（删除快照）。
5. 在确认页面上选择 Delete (删除)。

Amazon CLI

要删除 RDS Custom 快照，请使用 Amazon CLI 命令 delete-db-snapshot。

以下选项为必填项：

- `--db-snapshot-identifier` – 要删除的快照

以下示例将删除 `my-custom-snapshot` 数据库快照。

Example

对于 Linux、macOS 或 Unix：

```
aws rds delete-db-snapshot \
  --db-snapshot-identifier my-custom-snapshot
```

对于 Windows：

```
aws rds delete-db-snapshot ^
  --db-snapshot-identifier my-custom-snapshot
```

删除 RDS Custom for Oracle 自动备份

当不再需要 RDS Custom for Oracle 的保留自动备份时，您可以删除它们。此过程与删除 Amazon RDS 备份的过程相同。

控制台

删除保留的自动备份

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Automated backups（自动备份）。
3. 选择 Retained (保留)。
4. 选择要删除的保留自动备份。
5. 对于 Actions (操作)，选择 Delete (删除)。
6. 在确认页面上，输入 **delete me** 并选择 Delete (删除)。

Amazon CLI

可以通过使用 Amazon CLI 命令 delete-db-instance-automated-backup 删除保留的自动备份。

以下选项用于删除保留的自动备份：

- `--dbi-resource-id` – 源 RDS Custom 数据库实例的资源标识符。

  可以通过使用 Amazon CLI 命令 describe-db-instance-automated-backups 查找保留自动备份的源数据库实例的资源标识符。

以下示例删除具有源数据库实例资源标识符 `custom-db-123ABCEXAMPLE` 的保留自动备份。

Example

对于 Linux、macOS 或 Unix：

```
aws rds delete-db-instance-automated-backup \
    --dbi-resource-id custom-db-123ABCEXAMPLE
```

对于 Windows：

```
aws rds delete-db-instance-automated-backup ^
    --dbi-resource-id custom-db-123ABCEXAMPLE
```

使用 RDS Custom for Oracle 中的选项组

RDS Custom 使用选项组来启用和配置其它功能。选项组指定可用于 RDS Custom for Oracle 数据库实例的功能（称为选项）。选项可能具有指定该选项工作方式的设置。当您将 RDS Custom for Oracle 数据库实例与选项组相关联时，将为此实例启用指定的选项和选项设置。有关 Amazon RDS 中的选项组的一般信息，请参阅使用选项组。

主题

- RDS Custom for Oracle 中的选项组概述
- Oracle 时区

RDS Custom for Oracle 中的选项组概述

要为 Oracle 数据库启用选项，可将它们添加到选项组中，然后将选项组关联到数据库实例。有关更多信息，请参阅 使用选项组。

主题

- RDS Custom for Oracle 选项摘要
- 将选项添加到 RDS Custom for Oracle 数据库实例的基本步骤
- 在 RDS Custom for Oracle 中创建选项组
- 将选项组与 RDS Custom for Oracle 数据库实例关联

RDS Custom for Oracle 选项摘要

RDS Custom for Oracle 支持以下数据库实例选项。

选项	选项 ID	描述
Oracle 时区	`Timezone`	RDS Custom for Oracle 数据库实例所使用的时区。

将选项添加到 RDS Custom for Oracle 数据库实例的基本步骤

将选项添加到您的 RDS Custom for Oracle 数据库实例的一般过程如下：

1. 创建新的选项组，或者复制或修改现有选项组。

2. 将 选项添加到该选项组。
3. 在创建或修改选项组时，将其与数据库实例相关联。

在 RDS Custom for Oracle 中创建选项组

您可以创建一个新的选项组，该选项组从原定设置选项组派生其设置。然后，向新的选项组中添加一个或多个选项。或者，如果您已经有一个现有选项组，则可复制带有您已添加到新选项组的所有选项的选项组。要了解如何复制选项组，请参阅复制选项组。

RDS Custom for Oracle 的默认选项组如下所示：

- `default:custom-oracle-ee`
- `default:custom-oracle-se2`
- `default:custom-oracle-ee-cdb`
- `default:custom-oracle-se2-cdb`

创建选项组时，设置源自默认选项组。添加了 `TIME_ZONE` 选项后，即可将选项组与数据库实例相关联。

控制台

一种创建选项组的方法是使用 Amazon Web Services Management Console。

通过使用控制台创建新选项组

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择选项组。
3. 选择创建组。
4. 在创建选项组窗口中，执行以下操作：
   a. 对于 Name (名称)，输入选项组的名称，在 Amazon 账户内该名称需具有唯一性。此名称只能包含字母、数字和连字符。
   b. 对于说明，输入选项组的简要说明。此说明用于显示说明。
   c. 对于引擎，请选择以下任何 RDS Custom for Oracle 数据库引擎之一：
      - custom-oracle-ee

- custom-oracle-se2
- custom-oracle-ee-cdb
- custom-oracle-se2-cdb

d. 对于主要引擎版本，请选择 RDS Custom for Oracle 支持的主要引擎版本。有关更多信息，请参阅 [支持 RDS Custom for Oracle 的区域和数据库引擎]。

5. 要继续，请选择创建。要取消操作，请选择取消。

Amazon CLI

要创建选项组，请使用带下列必需参数的 Amazon CLI `create-option-group` 命令。

- `--option-group-name`
- `--engine-name`
- `--major-engine-version`
- `--option-group-description`

Example

下例创建了名为 `testoptiongroup` 的选项组，它是与 Oracle 企业版数据库引擎相关联的选项组。引号中的内容是描述。

对于 Linux、macOS 或 Unix：

```
aws rds create-option-group \
    --option-group-name testoptiongroup \
    --engine-name custom-oracle-ee-cdb \
    --major-engine-version 19 \
    --option-group-description "Test option group for a Custom Oracle CDB"
```

对于 Windows：

```
aws rds create-option-group ^
    --option-group-name testoptiongroup ^
    --engine-name custom-oracle-ee-cdb ^
    --major-engine-version 19 ^
    --option-group-description "Test option group for a Custom Oracle CDB"
```

RDS API

要创建选项组，请调用 Amazon RDS API `CreateOptionGroup` 操作。

将选项组与 RDS Custom for Oracle 数据库实例关联

您可将您的选项组与新的或现有的数据库实例关联：

- 对于新数据库实例，请在创建实例时应用选项组。有关更多信息，请参阅 创建 RDS Custom for Oracle 数据库实例。
- 对于现有数据库实例，请修改实例并附加新的选项组以应用选项组。有关更多信息，请参阅 修改 RDS Custom for Oracle 数据库实例。

Oracle 时区

要更改您的 RDS Custom for Oracle 数据库实例使用的系统时区，请使用时区选项。例如，您可能需要更改数据库实例的时区，从而与本地环境或旧版应用程序兼容。时区选项在主机级别更改时区。更改时区会影响所有日期列和值 (包括 `SYSDATE` 和 `SYSTIMESTAMP`)。

主题

- RDS Custom for Oracle 中的时区选项设置
- RDS Custom for Oracle 中的可用时区
- 有关在 RDS Custom for Oracle 中设置时区的注意事项
- 有关 RDS Custom for Oracle 中的时区设置的限制
- 将时区选项添加到选项组
- 删除时区选项

RDS Custom for Oracle 中的时区选项设置

Amazon RDS 支持时区选项的以下设置。

选项设置	有效值	描述
`TIME_ZONE`	可用时区之一。完整列表请参阅 RDS Custom for Oracle 中的可用时区。	数据库实例的新时区。

RDS Custom for Oracle 中的可用时区

以下值可用于时区选项。

区	时区
非洲	非洲/开罗、非洲/卡萨布兰卡、非洲/哈拉雷、非洲/拉各斯、非洲/罗安达、非洲/蒙罗维亚、非洲/内罗毕、非洲/的黎波里、非洲/温得和克
美洲	美洲/阿拉瓜伊纳、美洲/阿根廷/布宜诺斯艾利斯、美洲/亚森松、美洲/Bogota、美洲/加拉加斯、美洲/芝加哥、美洲/奇瓦瓦、美洲/库亚巴、美洲/丹佛、美洲/底特律、美洲/福塔莱萨、美洲/戈特霍布、美洲/危地马拉、美洲/哈利法克斯、美洲/利马、美洲/洛杉矶、美洲/玛瑙斯、美洲/马塔莫罗斯、美洲/墨西哥城、美洲/蒙特雷、美洲/蒙特维多、美洲/纽约、美洲/菲尼克斯、美洲/圣地亚哥、美洲/圣保罗、美洲/蒂华纳、美洲/多伦多
亚洲	亚洲/阿曼、亚洲/阿什哈巴德、亚洲/巴格达、亚洲/巴库、亚洲/曼谷、亚洲/贝鲁特、亚洲/加尔各答、亚洲/大马士革、亚洲/达卡、亚洲/香港、亚洲/伊尔库茨克、亚洲/雅加达、亚洲/耶路撒冷、亚洲/喀布尔、亚洲/卡拉奇、亚洲/加德满都、亚洲/加尔各答、亚洲/克拉斯诺亚尔斯克、亚洲/马加丹、亚洲/马尼拉、亚洲/马斯喀特、亚洲/新西伯利亚、亚洲/阳光、亚洲/利雅得、亚洲/首尔、亚洲/上海、亚洲/新加坡、亚洲/台北、亚洲/德黑兰、亚洲/东京、亚洲/乌兰巴托、亚洲/符拉迪沃斯托克、亚洲/雅库茨克、亚洲/埃里温
大西洋	大西洋/亚速尔、大西洋/佛得角
澳大利亚	澳大利亚/阿德莱德、澳大利亚/布里斯班、澳大利亚/达尔文、澳大利亚/尤克拉、澳大利亚/霍巴特、澳大利亚/豪勋爵岛、澳大利亚/珀斯、澳大利亚/悉尼
巴西	巴西/迪诺罗尼亚、巴西/东部
加拿大	加拿大/纽芬兰、加拿大/萨斯喀彻温省
ETC	ETC/GMT-3
欧洲	欧洲/阿姆斯特丹、欧洲/雅典、欧洲/柏林、欧洲/都柏林、欧洲/赫尔辛基、欧洲/加里宁格勒、欧洲/伦敦、欧洲/马德里、欧洲/莫斯科、欧洲/巴黎、欧洲/布拉格、欧洲/罗马、欧洲/萨拉热窝

区	时区
太平洋	太平洋/阿皮亚、太平洋/奥克兰、太平洋/查塔姆、太平洋/斐济、太平洋/关岛、太平洋/檀香山、太平洋/圣诞岛、太平洋/玛贵斯、太平洋/萨摩亚、太平洋/汤加塔布岛、太平洋/韦克
US	美国/阿拉斯加、美国/中部、美国/印第安纳东部、美国/西部、美国/太平洋地区
UTC	UTC

有关在 RDS Custom for Oracle 中设置时区的注意事项

如果您选择设置数据库实例的时区，请考虑以下事项：

- 添加时区选项时，数据库实例在自动重新启动时会出现短暂中断。
- 如果不小心错误地设置了时区，则必须将数据库实例恢复到其以前的时区设置。因此，我们强烈建议您在将时区选项添加到实例之前使用以下策略之一：
  - 如果 RDS Custom for Oracle 数据库实例使用默认选项组，请拍摄数据库实例的快照。有关更多信息，请参阅 [创建 RDS Custom for Oracle 快照]。
  - 如果您的数据库实例当前使用非默认选项组，请拍摄数据库实例的快照，然后使用时区选项创建一个新的选项组。
- 强烈建议您在应用 `Timezone` 选项后手动备份数据库实例。
- 强烈建议您在向生产数据库实例添加时区选项之前，先在测试数据库实例上测试此选项。对于使用系统日期添加日期或时间的表，添加时区选项可能引起问题。我们建议您对数据和应用程序进行分析，以评测更改时区造成的影响。

有关 RDS Custom for Oracle 中的时区设置的限制

请注意以下限制：

- 如果不将时区移到支持外围之外，就无法直接在主机上更改时区。要更改数据库时区，必须创建一个选项组。
- 由于时区选项是一个持久选项（但不是永久选项），因此您无法执行以下操作：
  - 添加选项后，将其从选项组中删除。
  - 将此选项的时区设置修改为其他时区。
- 您不能将多个选项组与 RDS Custom for Oracle 数据库实例关联。

- 您无法为 CDB 中的单个 PDB 设置时区。

将时区选项添加到选项组

RDS Custom for Oracle 的默认选项组如下所示：

- `default:custom-oracle-ee`
- `default:custom-oracle-se2`
- `default:custom-oracle-ee-cdb`
- `default:custom-oracle-se2-cdb`

创建选项组时，设置源自默认选项组。有关 Amazon RDS 中的选项组的一般信息，请参阅使用选项组。

控制台

将时区选项添加到选项组

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择选项组。
3. 选择要修改的选项组，然后选择 Add Option (添加选项)。
4. 在添加选项窗口中，执行以下操作：
   a. 选择时区。
   b. 在选项设置中，选择时区。
   c. 要在添加选项后在所有关联的 RDS Custom for Oracle 数据库实例上启用该选项，对于立即应用，请选择是。如果选择否（默认），则会在下一个维护时段为每个关联数据库实例启用此选项。
   d. **Important**

      如果在已附加到一个或多个数据库实例的现有选项组中添加时区选项，所有数据库实例自动重新启动时都会发生短暂中断。
5. 根据需要设置完毕后，选择 Add Option (添加选项)。

6. 备份时区已更新的 RDS Custom for Oracle 数据库实例。有关更多信息，请参阅 创建 RDS Custom for Oracle 快照。

Amazon CLI

以下示例使用 Amazon CLI add-option-to-option-group 命令将 `Timezone` 选项和 `TIME_ZONE` 选项设置添加到名为 `testoptiongroup` 的选项组。时区设置为 `America/Los_Angeles`。

对于 Linux、macOS 或 Unix：

```
aws rds add-option-to-option-group \
    --option-group-name "testoptiongroup" \
    --options "OptionName=Timezone,OptionSettings=[{Name=TIME_ZONE,Value=America/
Los_Angeles}]" \
    --apply-immediately
```

对于 Windows：

```
aws rds add-option-to-option-group ^
    --option-group-name "testoptiongroup" ^
    --options "OptionName=Timezone,OptionSettings=[{Name=TIME_ZONE,Value=America/
Los_Angeles}]" ^
    --apply-immediately
```

删除时区选项

时区选项是持久选项，但不是永久选项。将此选项添加到选项组后不能将其删除。要取消旧选项组与数据库实例的关联，请执行以下操作：

1. 使用更新后的 `Timezone` 选项创建新的选项组。
2. 修改数据库实例时，将新选项组与数据库实例关联。

将本地数据库迁移到 RDS Custom for Oracle

在将本地 Oracle 数据库迁移到 RDS Custom for Oracle 之前，需要考虑以下因素：

- 应用程序可以承受的停机时间
- 源数据库的大小
- 网络连接
- 对回退计划的要求
- 源和目标 Oracle 数据库版本和数据库实例操作系统类型
- 可用的复制工具，例如 Amazon Database Migration Service、Oracle GoldenGate 或第三方复制工具

基于这些因素，您可以选择物理迁移、逻辑迁移或两者的组合。如果选择物理迁移，可以使用以下技术：

RMAN 复制

主动数据库复制不需要备份您的源数据库。这种复制通过网络将数据库文件复制到辅助实例，将实时源数据库复制到目标主机。RMAN `DUPLICATE` 命令将所需文件复制为映像副本或备份集。要了解此技术，请参阅 Amazon 博客文章使用 RMAN 复制将 Oracle 数据库物理迁移到 Amazon RDS Custom。

Oracle Data Guard

使用此技术，您备份一个主本地数据库，并将备份复制到 Amazon S3 桶中。然后，将备份复制到 RDS Custom for Oracle 备用数据库实例。执行必要的配置后，手动将主数据库切换到 RDS Custom for Oracle 备用数据库。要了解此技术，请参阅 Amazon 博客文章使用 Data Guard 将 Oracle 数据库物理迁移到 Amazon RDS Custom。

有关将在逻辑上数据导入到 RDS for Oracle 的一般信息，请参阅将数据导入到 Amazon RDS 上的 Oracle。

为 Amazon RDS Custom for Oracle 升级数据库实例

您可以通过修改 Amazon RDS Custom 数据库实例以使用新的自定义引擎版本 (CEV) 来升级该实例。有关升级的一般信息，请参阅升级数据库实例引擎版本。

主题

RDS Custom for Oracle 升级概述

使用 RDS Custom for Oracle，您可以创建新的 CEV，然后将您的实例修改为使用新的 CEV，以此修补您的 Oracle 数据库或数据库实例操作系统（OS）。

主题

CEV 升级选项

当您创建 CEV 以进行升级时，可以选择以下互斥的选项：

仅限数据库

重复使用数据库实例当前所使用的亚马逊机器映像（AMI），但要指定不同的数据库二进制文件。RDS Custom 会分配一个新的二进制卷，然后将其附加到现有的 Amazon EC2 实例。RDS Custom 用使用目标数据库版本的新卷替换整个数据库卷。

仅限操作系统

> 重复使用数据库实例当前所使用的数据库二进制文件，但要指定不同的 AMI。RDS Custom 会分配一个新的 Amazon EC2 实例，然后将现有二进制卷附加到新实例。保留现有的数据库卷。

如果要同时升级操作系统和数据库，则必须升级 CEV 两次。您可以先升级操作系统后再升级数据库，也可以先升级数据库后再升级操作系统。

> **Warning**
>
> 修补操作系统时，会丢失根卷数据和任何现有的操作系统自定义项。因此，我们强烈建议您不要使用根卷来进行安装或存储永久数据或文件。我们还建议您在升级之前先备份数据。

在没有 CEV 的情况下进行修补

我们强烈建议您使用 CEV 升级 RDS Custom for Oracle 数据库实例。RDS Custom for Oracle 自动化可将补丁元数据与数据库实例上的数据库二进制文件同步。

在特殊情况下，RDS Custom 支持使用 OPatch 实用工具将“一次性”数据库补丁直接应用于底层 Amazon EC2 实例。有效的使用案例可能是您想要立即应用的数据库补丁，但 RDS Custom 团队正在升级 CEV 功能，这会导致延迟。要手动应用数据库补丁，请执行以下步骤：

1. 暂停 RDS Custom 自动化。
2. 将您的补丁应用于 Amazon EC2 实例上的数据库二进制文件。
3. 恢复 RDS Custom 自动化。

上述技术的缺点是必须手动将数据库补丁应用到要升级的每个实例。相比之下，当您创建新的 CEV 时，您可以使用相同的 CEV 创建或升级多个数据库实例。

使用 CEV 修补数据库实例的一般步骤

无论您修补操作系统还是数据库，请执行以下基本步骤：

1. 创建包含以下任一内容的 CEV，具体取决于您要修补数据库还是操作系统：
   - 要应用于数据库实例的 Oracle 数据库 RU
   - 不同的 AMI（最新的 AMI，或是您指定的 AMI），以及一个用作源的现有 CEV

   按照 创建 CEV 中的步骤操作。

2. （对于数据库修补为可选）通过运行 `describe-db-engine-versions` 来检查可用的引擎版本升级。
3. 通过运行 `modify-db-instance` 启动修补过程。

   正在修补的实例的状态不同，如下所示：

   - 当 RDS 修补数据库时，数据库实例的状态将更改为正在升级。
   - 当 RDS 修补操作系统时，数据库实例的状态将更改为正在修改。

   当数据库实例的状态为可用时，修补已完成。
4. 通过运行 `describe-db-instances` 来确认您的数据库实例使用的是新 CEV。

RDS Custom for Oracle 升级的要求

在将您的 RDS Custom for Oracle 数据库实例升级到目标 CEV 时，请确保满足以下要求：

- 要升级到的目标 CEV 必须存在。
- 您必须以单个操作升级操作系统或数据库。不支持在单个 API 调用中同时升级操作系统和数据库。
- 目标 CEV 必须使用当前 CEV 清单中的安装参数设置。例如，您无法将使用默认 Oracle 主目录的数据库升级到使用非默认 Oracle 主目录的 CEV。
- 对于数据库升级，目标 CEV 必须使用新的次要数据库版本，而不是新的主要版本。例如，您无法从 Oracle Database 12c CEV 升级到 Oracle Database 19c CEV。但是，您可以从版本 21.0.0.0.ru-2023-04.rur-2023-04.r1 版本升级到版本 21.0.0.0.ru-2023-07.rur-2023-07.r1。
- 对于操作系统升级，目标 CEV 必须使用不同的 AMI，但具有相同的主要版本。

RDS Custom for Oracle 数据库升级注意事项

如果您计划升级数据库，请考虑以下事项：

- 当前支持的操作系统（OS）版本为 Oracle Linux 8。要继续接收来自 RDS Custom for Oracle 的最新安全更新和补丁，请通过指定基于此操作系统的 CEV，将您的数据库实例升级到 Oracle Linux 8。Oracle Database 12c 版本 1（12.1）、Oracle Database 版本 2（12.2）和 Oracle Database 19c 是唯一支持 Oracle Linux 8 的版本。要迁移到最新的 Oracle Linux 8 AMI，请将您的操作系统升级到最新的 AMI。有关更多信息，请参阅 [升级 RDS Custom for Oracle 数据库实例]。

  Oracle Linux 7.9 于 2024 年 12 月 31 日终止支持。要在支持终止后继续运行 Oracle Linux 7，需要购买 Oracle 扩展支持许可证。您负责安全更新，并且必须手动修补 RDS Custom for Oracle 实

例。有关更多信息，请参阅 [Lifetime Support Policy: Coverage for Oracle Open Source Service Offerings]。

- 当您升级主数据库实例中的数据库二进制文件时，RDS Custom for Oracle 会自动升级您的只读副本。但是，在升级操作系统时，您必须手动升级只读副本。
- 在将容器数据库（CDB）升级为新的数据库版本时，RDS Custom for Oracle 会检查所有 PDB 是否已打开或是否能够打开。如果不满足这些条件，RDS Custom 会停止检查并将数据库返回到其原始状态，而无需尝试升级。如果满足条件，RDS Custom 首先修补 CDB 根目录，然后并行修补所有其他 PDB（包括 `PDB$SEED`）。

  完成修补后，RDS Custom 会尝试打开所有 PDB。如果任何 PDB 无法打开，则会收到以下事件：`The following PDBs failed to open: *list-of-PDBs*`。如果 RDS Custom 无法修补 CDB 根目录或任何 PDB，则实例将进入 `PATCH_DB_FAILED` 状态。
- 您可能需要同时执行主要数据库版本升级和将非 CDB 转换为 CDB。在这种情况下，建议您按以下步骤操作：

  1. 创建一个使用 Oracle 多租户架构的新 RDS Custom for Oracle 数据库实例。
  2. 将非 CDB 插入 CDB 根目录，将其创建为 PDB。确保非 CDB 与您的 CDB 是相同的主要版本。
  3. 通过运行 `noncdb_to_pdb.sql` Oracle SQL 脚本来转换您的 PDB。
  4. 验证您的 CDB 实例。
  5. 升级 CDB 实例。

RDS Custom for Oracle 操作系统升级注意事项

在计划操作系统升级时，请考虑以下事项：

- 您不能提供自己的 AMI 以在 RDS Custom for Oracle CEV 中使用。您可以指定使用 Oracle Linux 8 的默认 AMI，也可以指定 RDS Custom for Oracle CEV 以前使用过的 AMI。

  > **Note**
  >
  > 当发现常见漏洞时，RDS Custom for Oracle 会发布新的默认 AMI。没有固定的时间表可用或提供保证。RDS Custom for Oracle 往往每 30 天发布一次新的默认 AMI。

- 当您升级主数据库实例中的操作系统时，必须手动升级其关联的只读副本。
- 请在开始修补操作系统之前，为可用区中的实例类型预留足够的 Amazon EC2 计算容量。

创建容量预留时，您需要指定可用区、实例数量和实例属性（包括实例类型）。例如，如果数据库实例使用底层 EC2 实例类型 r5.large，建议您在可用区中为 r5.large 预留 EC2 容量。在操作系统修补期间，RDS Custom 会创建一个类型为 db.r5.large 的新主机，如果可用区缺少用于该实例类型的 EC2 容量，则该主机可能会卡顿。如果您预留 EC2 容量，则可以降低因容量限制而导致修补受阻的风险。有关更多信息，请参阅《Amazon EC2 用户指南》中的按需容量预留。

- 在升级操作系统之前，请对数据库实例进行备份。升级会删除您的根卷数据和任何现有的操作系统自定义项。
- 在责任共担模型中，您有责任使操作系统保持最新状态。RDS Custom for Oracle 不强制要求您将哪些补丁应用于您的操作系统。如果您的 RDS Custom for Oracle 功能正常，则可以无限期地使用与此 CEV 关联的 AMI。

查看 RDS Custom for Oracle 数据库实例的有效 CEV 升级目标

您可以在 Amazon Web Services Management Console 中的 Custom engine version（自定义引擎版本）页面中查看现有的 CEV。

您也可以使用 describe-db-engine-versions Amazon CLI 命令查找适用于数据库实例升级的有效 CEV，如以下示例所示。此示例假设您使用引擎版本 `19.my_cev1` 创建了数据库实例，并且升级版本 `19.my_cev2` 和 `19.my_cev` 存在。

```
aws rds describe-db-engine-versions --engine custom-oracle-ee --engine-version
 19.my_cev1
```

输出与以下内容类似。`ImageId` 字段显示 AMI ID。

```
{
    "DBEngineVersions": [
        {
            "Engine": "custom-oracle-ee",
            "EngineVersion": "19.my_cev1",
            ...
            "Image": {
                "ImageId": "ami-2345",
                "Status": "active"
            },
            "DBEngineVersionArn": "arn:aws:rds:us-west-2:123456789012:cev:custom-
oracle-ee/19.my_cev1/12a34b5c-67d8-90e1-2f34-gh56ijk78lm9"
            "ValidUpgradeTarget": [
                {
```

```
                "Engine": "custom-oracle-ee",
                "EngineVersion": "19.my_cev2",
                "Description": "19.my_cev2 description",
                "AutoUpgrade": false,
                "IsMajorVersionUpgrade": false
            },
            {
                "Engine": "custom-oracle-ee",
                "EngineVersion": "19.my_cev3",
                "Description": "19.my_cev3 description",
                "AutoUpgrade": false,
                "IsMajorVersionUpgrade": false
            }
        ]
        ...
```

升级 RDS Custom for Oracle 数据库实例

要升级您的 RDS Custom for Oracle 数据库实例，请将其修改为使用新的 CEV。此 CEV 可以包含新的数据库二进制文件或新的 AMI。例如，要将 Oracle Linux 7.9 数据库实例升级到 Oracle Linux 8，请指定使用 Oracle Linux 8 的最新 AMI。如果要升级数据库和操作系统，则必须分别执行两次升级。

> **Note**
>
> 如果升级数据库，RDS Custom 会在升级主数据库实例后自动升级只读副本。如果升级操作系统，则必须手动升级副本。

开始之前，请查看 RDS Custom for Oracle 升级的要求和 RDS Custom for Oracle 数据库升级注意事项。

控制台

升级 RDS Custom for Oracle 数据库实例

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择数据库，然后选择要升级的 RDS Custom for Oracle 数据库实例。
3. 选择修改。将显示修改数据库实例页面。
4. 对于数据库引擎版本，请选择新的 CEV。执行以下操作：

- 如果您要修补数据库，请确保 CEV 指定的数据库二进制文件与数据库实例所用的二进制文件不同，并且不指定与数据库实例当前所用 AMI 不同的 AMI。
- 如果您要修补操作系统，请确保 CEV 指定的 AMI 与数据库实例当前所用的 AMI 不同，并且不指定不同的数据库二进制文件。

> **Warning**
>
> 修补操作系统时，会丢失根卷数据和任何现有的操作系统自定义项。

5. 请选择 Continue（继续），以查看修改摘要。

   请选择 Apply immediately（立即应用）以立即应用更改。
6. 如果更改正确无误，请选择 Modify DB Instance（修改数据库实例）。也可以选择 Back (返回) 编辑您的更改，或选择 Cancel (取消) 取消更改。

Amazon CLI

以下示例显示了可能的升级方案。这些示例假设您创建了具有以下特征的 RDS Custom for Oracle 数据库实例：

- 数据库实例名为 `my-custom-instance`
- CEV 名为 `19.my_cev1`
- 使用非 CDB 架构的 Oracle Database 19c
- 使用 AMI `ami-1234` 的 Oracle Linux 8

提供最新服务的 AMI 是 `ami-2345`。您可以通过运行 CLI 命令 `describe-db-engine-versions` 找到您的 AMI：

主题

- [升级操作系统]
- [升级数据库]

升级操作系统

在本示例中，您要将 ami-1234 升级到 ami-2345，即提供最新服务的 AMI。由于这是操作系统升级，所以 ami-1234 和 ami-2345 的数据库二进制文件必须相同。您可以基于 19.my_cev1 创建一个名为 19.my_cev2 的新 CEV。

Example

对于 Linux、macOS 或 Unix：

```
aws rds create-custom-db-engine-version \
    --engine custom-oracle-ee \
    --engine-version 19.my_cev2 \
    --description "Non-CDB CEV based on ami-2345" \
    --kms-key-id key-name \
    --source-custom-db-engine-version-identifer arn:aws:rds:us-
west-2:123456789012:cev:custom-oracle-ee/19.my_cev1/12345678-ab12-1234-cde1-
abcde123456789 \
    --image-id ami-2345
```

对于 Windows：

```
aws rds create-custom-db-engine-version ^
    --engine custom-oracle-ee ^
    --engine-version 19.my_cev2 ^
    --description "Non-CDB CEV based on ami-2345" ^
    --kms-key-id key-name ^
    --source-custom-db-engine-version-identifer arn:aws:rds:us-
west-2:123456789012:cev:custom-oracle-ee/19.my_cev1/12345678-ab12-1234-cde1-
abcde123456789 ^
    --image-id ami-2345
```

要升级 RDS Custom 数据库实例，请使用带有下列参数的 modify-db-instance Amazon CLI 命令：

- `--db-instance-identifier` – 指定要升级的 RDS Custom for Oracle 数据库实例。
- `--engine-version` – 指定具有新 AMI 的 CEV。
- `--no-apply-immediately` | `--apply-immediately` – 指定是立即执行升级还是等到计划的维护时段

以下示例将 my-custom-instance 升级到版本 19.my_cev2。只升级了操作系统。

Example

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier my-custom-instance \
    --engine-version 19.my_cev2 \
    --apply-immediately
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier my-custom-instance ^
    --engine-version 19.my_cev2 ^
    --apply-immediately
```

升级数据库

在此示例中，您希望将 Oracle 补丁 p35042068 应用到 RDS for Oracle 数据库实例。由于您在升级操作系统中升级了操作系统，因此您的数据库实例当前正在使用 `19.my_cev2`，其基于 `ami-2345`。您创建了一个名为 `19.my_cev3` 的新 CEV，该 CEV 也使用 `ami-2345`，但您在 `$MANIFEST` 环境变量中指定一个新的 JSON 清单。因此，只有新 CEV 中的数据库二进制文件与您的实例当前所用的 CEV 不同。

Example

对于 Linux、macOS 或 Unix：

```
aws rds create-custom-db-engine-version \
    --engine custom-oracle-ee \
    --engine-version 19.my_cev3 \
    --description "Non-CDB CEV with p35042068 based on ami-2345" \
    --kms-key-id key-name \
    --image-id ami-2345 \
    --manifest $MANIFEST
```

对于 Windows：

```
aws rds create-custom-db-engine-version ^
    --engine custom-oracle-ee ^
    --engine-version 19.my_cev3 ^
```

```
    --description "Non-CDB CEV with p35042068 based on ami-2345" ^
    --kms-key-id key-name ^
    --image-id ami-2345 ^
    --manifest $MANIFEST
```

以下示例将 my-custom-instance 升级到引擎版本 19.my_cev3。只升级了数据库。

Example

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier my-custom-instance \
    --engine-version 19.my_cev3 \
    --apply-immediately
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier my-custom-instance ^
    --engine-version 19.my_cev3 ^
    --apply-immediately
```

查看 RDS Custom 数据库实例的待定数据库升级

您可以使用 describe-db-instances 或 describe-pending-maintenance-actions Amazon CLI 命令查看 Amazon RDS Custom 数据库实例的待定数据库升级。

但是，如果您使用 --apply-immediately 选项或升级正在进行中，则此方法将不起作用。

以下 describe-db-instances 命令会显示 my-custom-instance 的待定数据库升级。

```
aws rds describe-db-instances --db-instance-identifier my-custom-instance
```

输出与以下内容类似。

```
{
    "DBInstances": [
        {
           "DBInstanceIdentifier": "my-custom-instance",
            "EngineVersion": "19.my_cev1",
```

```
            ...
            "PendingModifiedValues": {
                "EngineVersion": "19.my_cev3"
            ...
            }
        }
    ]
}
```

排查 RDS Custom for Oracle 数据库实例升级失败的问题

如果 RDS Custom 数据库实例升级失败，将生成 RDS 事件并且数据库实例状态变为 `upgrade-failed`。

您可以使用 [describe-db-instances](https://docs.aws.amazon.com/cli/latest/reference/rds/describe-db-instances.html) Amazon CLI 命令查看此状态，如以下示例所示。

```
aws rds describe-db-instances --db-instance-identifier my-custom-instance
```

输出与以下内容类似。

```
{
    "DBInstances": [
        {
           "DBInstanceIdentifier": "my-custom-instance",
            "EngineVersion": "19.my_cev1",
            ...
            "PendingModifiedValues": {
                "EngineVersion": "19.my_cev3"
            ...
            }
            "DBInstanceStatus": "upgrade-failed"
        }
    ]
}
```

升级失败后，所有数据库操作都将被阻止，修改数据库实例以执行以下任务除外：

- 重试相同的升级
- 暂停和恢复 RDS Custom 自动化
- 时间点恢复 (PITR)
- 删除数据库实例

> **Note**
>
> 如果已为 RDS Custom 数据库实例暂停自动化操作，则在恢复自动化之前，您将无法重试升级。
>
> RDS 管理的只读副本的升级失败与主副本的升级失败情况适用相同的操作。

有关更多信息，请参阅 [排查 RDS Custom for Oracle 的升级问题]。

针对 Amazon RDS Custom for Oracle 排查数据库问题

RDS Custom 的责任共担模型提供了操作系统外壳级别的访问权限和数据库管理员访问权限。RDS Custom 会在您的账户中运行资源，与 Amazon RDS 不同，因为后者会在系统账户中运行资源。访问权限更多，责任也更大。在下面的各个部分，您可以了解如何排查 Amazon RDS Custom 数据库实例的问题。

> **Note**
>
> 本节介绍如何排查 RDS Custom for Oracle 的问题。有关排查 RDS Custom for Oracle 的问题，请参阅针对 Amazon RDS Custom for SQL Server 排查数据库的问题。

主题

查看 RDS Custom 事件

对于 RDS Custom 和 Amazon RDS 数据库实例而言，查看事件的流程是相同的。有关更多信息，请参阅 查看 Amazon RDS 事件。

要使用 Amazon CLI 查看 RDS Custom 事件通知，请使用 `describe-events` 命令。RDS Custom 推出了几项新事件。事件类别与 Amazon RDS 的事件类别相同。有关事件列表，请参阅 Amazon RDS 事件类别和事件消息。

以下示例检索指定 RDS Custom 数据库实例发生的事件的详细信息。

```
aws rds describe-events \
    --source-identifier my-custom-instance \
    --source-type db-instance
```

订阅 RDS Custom 事件

对于 RDS Custom 和 Amazon RDS 数据库实例而言，订阅事件的流程是相同的。有关更多信息，请参阅 订阅 Amazon RDS 事件通知。

要使用 CLI 订阅 RDS Custom 事件通知，请使用 `create-event-subscription` 命令。包括以下必需参数：

- `--subscription-name`
- `--sns-topic-arn`

以下示例为当前的 Amazon 账户中的 RDS Custom 数据库实例创建了备份和恢复事件的订阅。您也可以将通知发送到 `--sns-topic-arn` 指定的 Amazon Simple Notification Service (Amazon SNS) 主题。

```
aws rds create-event-subscription \
    --subscription-name my-instance-events \
    --source-type db-instance \
    --event-categories '["backup","recovery"]' \
    --sns-topic-arn arn:aws:sns:us-east-1:123456789012:interesting-events
```

为 RDS Custom for Oracle 排除自定义引擎版本创建的故障

当 CEV 创建失败时，RDS Custom 将发布带有消息 `Creation failed for custom engine version` *`major-engine-version.cev_name`* 的 `RDS-EVENT-0198`，并包含有关失败的详细信息。例如，事件会打印丢失的文件。

CEV 创建可能会因为以下问题而失败：

- 包含安装文件的 Amazon S3 存储桶与您的 CEV 不在同一个 Amazon 区域。
- 当您首次在 Amazon Web Services 区域 中提出 CEV 创建请求时，RDS Custom 将创建 S3 存储桶，用于存储 RDS Custom 资源（例如 CEV 构件，Amazon CloudTrail 日志和事务日志）。

  如果 RDS Custom 无法创建 S3 存储桶，则 CEV 创建将会失败。要么调用者没有 步骤 5：为您的 IAM 用户或角色授予所需的权限 中所述的 S3 权限，要么 S3 存储桶的数量已达到限制。
- 调用者没有权限以从包含安装媒体文件的 S3 存储桶中获取文件。步骤 7：添加必要的 IAM 权限 中描述了这些权限。

- 您的 IAM 策略有 `aws:SourceIp` 条件。请务必遵循《Amazon Identity and Access Management 用户指南》中 Amazon 根据源 IP 拒绝对 Amazon 的访问中的建议。还要确保调用方具有 步骤 5：为您的 IAM 用户或角色授予所需的权限 中所述的 S3 权限。
- CEV 清单中列出的安装媒体文件不在 S3 存储桶中。
- RDS Custom 不知道安装文件的 SHA-256 校验和。

  确认所提供文件的 SHA-256 校验和与 Oracle 网站上的 SHA-256 校验和相符。如果校验和相符，请联系 Amazon Support 并提供失败的 CEV 名称、文件名和校验和。
- OPatch 版本与补丁文件不兼容。您可能会收到以下消息：`OPatch is lower than minimum required version. Check that the version meets the requirements for all patches, and try again`。要应用 Oracle 补丁，必须使用兼容的 OPatch 实用工具版本。您可以在补丁的自述文件中找到所需的 OPatch 实用工具版本。从 My Oracle Support 下载最新的 OPatch 实用工具，然后再次尝试创建 CEV。
- CEV 清单中所指定补丁的顺序错误。

您可以在 RDS 控制台（在导航窗格中，选择 Events（事件））中或者使用 `describe-events` Amazon CLI 命令查看 RDS 事件。默认持续时间为 60 分钟。如果没有返回任何事件，请指定更长的持续时间，如以下示例中所示。

```
aws rds describe-events --duration 360
```

目前，从 Amazon S3 导入文件以创建 CEV 的 MediaImport 服务没有与 Amazon CloudTrail 整合。因此，如果您在 CloudTrail 中为 Amazon RDS 启用数据记录功能，对诸如 `CreateCustomDbEngineVersion` 事件等 MediaImport 服务的调用将不会记录。

但是，您可能会看到来自访问您 Amazon S3 存储桶的 API Gateway 的调用。这些调用来自 `CreateCustomDbEngineVersion` 事件的 MediaImport 服务。

修复 RDS Custom for Oracle 中不支持的配置

在责任共担模式中，您应负责修复导致 RDS Custom for Oracle 数据库实例进入 `unsupported-configuration` 状态的配置问题。如果问题与 Amazon 基础设施有关，则可以使用控制台或 Amazon CLI 修复该问题。如果问题与操作系统或数据库配置有关，则可以登录到主机进行修复。

> **Note**
>
> 本节介绍如何修复 RDS Custom for Oracle 中不支持的配置。有关 RDS Custom for SQL Server 的信息，请参阅修复 RDS Custom for SQL Server 中不支持的配置。

下表包含支持外围所发送的通知和事件以及如何修复这些问题的说明。这些通知和支持外围可能会更改。有关支持外围的背景，请参阅RDS Custom 支持外围。有关事件描述，请参阅Amazon RDS 事件类别和事件消息。

事件 ID	配置	RDS 事件消息	操作
SP-00000	手动不支持的配置	RDS Custom 数据库实例状态设置为 [不受支持的配置]，原因是 *##*。。	要解决此问题，请创建 Amazon Web Services 支持 案例。
Amazon 资源（基础设施）			
SP-O1001	Amazon Elastic Block Store（Amazon EBS）卷	以下 EBS 卷已添加到 EC2 实例 *ec2_id* 中：*volume_id* 。要解决此问题，请将指定的卷与实例分离。	除了从 Amazon Machine Image（AMI）创建的根卷之外，RDS Custom 还创建了两种类型的 EBS 卷，并将它们与 EC2 实例关联。 • 二进制卷是数据库软件二进制文件所在的位置 • 数据卷是数据库文件所在的位置 当您创建数据库实例时，您指定的存储配置将配置数据卷。 支持外围将监控以下内容： • 使用数据库实例创建的初始 EBS 卷仍与该实例相关联。 • 初始 EBS 卷仍具有与最初设置相同的配置：存储类型、大小、预置 IOPS 和存储吞吐量。 • 没有额外的 EBS 卷附加到数据库实例。

<table>
<tr><th>事件 ID</th><th>配置</th><th>RDS 事件消息</th><th>操作</th></tr>
<tr><td></td><td></td><td></td><td>使用以下 CLI 命令比较 EBS 卷详细信息和 RDS Custom for Oracle DB 实例详细信息的卷类型。<pre><code>aws rds describe-db-instances \
    --db-instance-identifier db-instance-
name | grep StorageType</code></pre></td></tr>
<tr><td>SP-O1002</td><td>Amazon Elastic Block Store（Amazon EBS）卷</td><td>EBS 卷 volume_id 已与 EC2 实例 [ec2_id] 相分离。您无法将原始卷与该实例进行分离。要解决此问题，请将 volume_id 重新附加到 ec2_id。</td><td>除了从 Amazon Machine Image（AMI）创建的根卷之外，RDS Custom 还创建了两种类型的 EBS 卷，并将它们与 EC2 实例关联。
二进制卷是数据库软件二进制文件所在的位置
数据卷是数据库文件所在的位置
当您创建数据库实例时，您指定的存储配置将配置数据卷。

支持外围将监控以下内容：
使用数据库实例创建的初始 EBS 卷仍与该实例相关联。
初始 EBS 卷仍具有与最初设置相同的配置：存储类型、大小、预置 IOPS 和存储吞吐量。
没有额外的 EBS 卷附加到数据库实例。
使用以下 CLI 命令比较 EBS 卷详细信息和 RDS Custom for Oracle DB 实例详细信息的卷类型。<pre><code>aws rds describe-db-instances \
    --db-instance-identifier db-instance-
name | grep StorageType</code></pre></td></tr>
</table>

<table>
<tr><th>事件 ID</th><th>配置</th><th>RDS 事件消息</th><th>操作</th></tr>
<tr>
<td>SP-O1003</td>
<td>Amazon Elastic Block Store（Amazon EBS）卷</td>
<td>对已附加到 EC2 实例 ec2_id 的源 EBS 卷 volume_id 进行了如下修改：大小从 [X] 更改为 [Y]，类型从 [N] 更改为 [M]，IOPS 从 [J] 更改为 [K]。要解决此问题，请恢复修改。</td>
<td>除了从 Amazon Machine Image（AMI）创建的根卷之外，RDS Custom 还创建了两种类型的 EBS 卷，并将它们与 EC2 实例关联。
二进制卷是数据库软件二进制文件所在的位置
数据卷是数据库文件所在的位置
当您创建数据库实例时，您指定的存储配置将配置数据卷。

支持外围将监控以下内容：
使用数据库实例创建的初始 EBS 卷仍与该实例相关联。
初始 EBS 卷仍具有与最初设置相同的配置：存储类型、大小、预置 IOPS 和存储吞吐量。
没有额外的 EBS 卷附加到数据库实例。
使用以下 CLI 命令比较 EBS 卷详细信息和 RDS Custom for Oracle DB 实例详细信息的卷类型。
<pre>aws rds describe-db-instances \
    --db-instance-identifier db-instance-name | grep StorageType</pre>
</td>
</tr>
<tr>
<td>SP-O1004</td>
<td>Amazon EC2 实例状态</td>
<td>自动恢复使 EC2 实例 [ec2_id] 处于受损状态。要解决此问题，请参阅实例恢复故障排除。</td>
<td>要检查数据库实例的状态，请使用控制台或运行以下 Amazon CLI 命令：
<pre>aws rds describe-db-instances \
    --db-instance-identifier db-instance-name |grep DBInstanceStatus</pre>
</td>
</tr>
</table>

事件 ID	配置	RDS 事件消息	操作
SP-O1005	Amazon EC2 实例属性	对 EC2 实例 [*ec2_id*] 进行了如下修改：属性 [*att1*] 从 [*val-old*] 更改为 [*val-new*]，属性 [*att2*] 从 [*val-old*] 更改为 [*val-new*]。要解决此问题，请恢复到原始值。	
SP-O1006	Amazon EC2 实例状态	EC2 实例 [*ec2_id*] 已终止或找不到。要解决问题，请删除 RDS Custom 数据库实例。	支持外围将监控 EC2 实例状态更改通知。EC2 实例必须始终运行。 删除数据库实例 1. 要检查数据库实例的状态，请使用控制台或运行以下 Amazon CLI 命令： `aws rds describe-db-instances \ --db-instance-identifier db-instance-name |grep DBInstanceStatus` 2. 删除 RDS Custom for Oracle 数据库实例。

事件 ID	配置	RDS 事件消息	操作
SP-O1007	Amazon EC2 实例状态	EC2 实例 [*ec2_id*] 已停止。要解决问题，请启动该实例。	支持外围将监控 EC2 实例状态更改通知。EC2 实例必须始终运行。 重新启动数据库实例 1. 要检查数据库实例的状态，请使用控制台或运行以下 Amazon CLI 命令： `aws rds describe-db-instances \ --db-instance-identifier db-instance-name |grep DBInstanceStatus` 2. 启动数据库实例。 3. 重新装载二进制文件和数据卷。
操作系统			

事件 ID	配置	RDS 事件消息	操作

SP-O2001	RDS Custom 代理状态	RDS Custom 代理未在 EC2 实例 [*ec2_id*] 上运行。确保代理在 [*ec2_id*] 上运行。	在 RDS Custom for Oracle 上，如果 RDS Custom 代理停止，则数据库实例将超出支持外围。代理每 30 秒会向 Amazon CloudWatch 发布一次 `IamAlive` 指标。如果指标在 30 秒之后未能发布，则会触发告警。支持外围还会每隔 30 分钟监控主机上的 RDS Custom 代理进程状态。
			
			重新启动 RDS Custom 代理
			
			1. 登录到主机并确保 RDS Custom 代理正在运行。
			2. 运行以下命令之一以检查代理的状态。
			`service rdscustomagent status`
			3. 要启动代理，请使用以下命令。
			`service rdscustomagent start`
			
			当 RDS Custom 代理再次运行时，`IamAlive` 指标将发布到 Amazon CloudWatch，并且告警将切换到 OK 状态。此切换操作将通知支持外围代理正在运行。

SP-O2002	Amazon Systems Manager 代理（SSM 代理）状态	无法在 EC2 实例 [*ec2_id*] 上访问 Systems Manager 代理。请确保您已正确配置网络、代理和 IAM 权限。	SSM 代理必须始终运行。RDS Custom 代理负责确保 Systems Manager 代理正在运行。如果 SSM 代理终止并重新启动，RDS Custom 代理将向 CloudWatch 发布指标。RDS Custom 代理对指标设置了一项告警，当过去每三分钟内出现一次重新启动时，即会触发该告警。支持外围还会每隔 30 分钟监控主机上 SSM 代理的进程状态。 有关更多信息，请参阅 SSM 代理故障排除。

事件 ID	配置	RDS 事件消息	操作
SP-O2003	Amazon Systems Manager 代理（SSM 代理）状态	EC2 实例 [*ec2_id*] 上的 Systems Manager 代理多次崩溃。有关更多信息，请参阅 SSM 代理故障排除文档。	有关更多信息，请参阅 SSM 代理故障排除。
SP-O2004	操作系统时区	EC2 实例 [*ec2_id*] 上的时区已更改。要解决此问题，请将时区恢复为之前的 [*previous-time-zone*] 设置。然后使用 RDS 选项组更改时区。	RDS 自动化检测到在未使用选项组的情况下更改了主机上的时区。这种主机级别的更改可能会导致 RDS 自动化失败，因此 EC2 实例处于 `unsupported-configuration` 状态。 修复时区设置 1. 登录您的 EC2 主机并检查操作系统时区，如下所示： `timedatectl` 2. 暂停 RDS Custom 自动化。有关更多信息，请参阅暂停和恢复 RDS Custom 数据库实例。 3. 停止数据库实例。 4. 恢复操作系统上的时区更改。 5. 开启数据库实例。 6. 恢复 RDS Custom 自动化。 数据库实例将在 30 分钟内可用。为了防止将来移出外围，请通过选项组修改您的时区。有关更多信息，请参阅 Oracle 时区。

事件 ID	配置	RDS 事件消息	操作
SP-O2005	sudo 配置	EC2 实例 [*ec2_id*] 上的 sudo 配置缺少必要的权限。要解决此问题，请恢复最近对 sudo 配置进行的更改。	支持外围将验证是否允许某些操作系统用户在主机上运行某些命令。它监视 sudo 配置并将其与受支持的状态进行比较。 当 sudo 配置不受支持时，RDS Custom 会尝试覆盖它们并返回到之前的受支持状态。如果尝试成功，RDS Custom 会发送以下通知： RDS Custom 成功覆盖了您的配置。 如果覆盖失败，您的数据库实例将仍处于不受支持的配置状态。要解决此问题，要么恢复 `sudoers.d/` 文件中的更改，要么修复权限。 调查对 **sudo** 配置进行的更改 1. 登录到主机。 2. 运行以下命令。 `visudo -c -f /etc/sudoers.d/ individual_sudo_files` 3. 根据需要修改 sudo 配置。 在支持外围确定 sudo 配置受支持时，RDS Custom for Oracle 数据库实例会在 30 分钟内变为可用状态。
SP-O2006	S3 存储桶可访问性	RDS Custom 自动化功能无法从 EC2 实例 [*ec2_id*] 上的 S3 存储桶下载文件。检查网络配置，并确保该实例允许与 S3 建立双向连接。	

事件 ID	配置	RDS 事件消息	操作
数据库			
SP-O3001	数据库存档滞后目标	EC2 实例 [*ec2_id*] 上的 ARCHIVE_LAG_TARGET 参数超出建议的范围 *value_range* 。要解决此问题，请将参数设置为 value_range 范围内的值。	支持外围监控 `ARCHIVE_LAG_TARGET` 数据库参数，以验证数据库实例的最新可还原时间是否在合理的范围内。 更改已存档重做日志的延迟目标 1. 登录您的 EC2 主机 2. 连接到 RDS Custom for Oracle 数据库实例 3. 将 `ARCHIVE_LAG_TARGET` 参数更改为介于 60–7200 之间的值。例如，使用以下 SQL 语句。 `ALTER SYSTEM SET ARCHIVE_LAG_TARGET=300 SCOPE=BOTH;` 数据库实例将在 30 分钟内可用。

<table>
<tr><th>事件 ID</th><th>配置</th><th>RDS 事件消息</th><th>操作</th></tr>
<tr><td>SP-O3002</td><td>Oracle Data Guard 角色</td><td>EC2 实例 [ec2_id] 上的 Oracle Data Guard 不支持数据库角色 [role_name]。要解决此问题，请将 DATABASE_ROLE 参数设置为 PRIMARY 或 PHYSICAL STANDBY。</td><td>
支持外围每 15 秒监控当前数据库角色，并在数据库角色发生更改时发送 CloudWatch 通知。Oracle Data Guard <code>DATABASE_ROLE</code> 参数必须为 <code>PRIMARY</code> 或 <code>PHYSICAL STANDBY</code>。
将 Oracle Data Guard 数据库角色还原为受支持的值

运行以下语句检查 Oracle Data Guard 角色：<pre><code>SELECT DATABASE_ROLE FROM V$DATABASE;</code></pre>
如果您的数据库实例是独立的，则可以使用以下任一语句将其改回为 <code>PRIMARY</code> 角色：<pre><code>ALTER DATABASE COMMIT TO SWITCHOVER
 PRIMARY;
ALTER DATABASE ACTIVATE STANDBY DATABASE;</code></pre>
如果您的数据库实例是副本，则可以使用以下语句将其改回为 <code>PHYSICAL STANDBY</code> 角色:
<pre><code>ALTER DATABASE CONVERT TO PHYSICAL STANDBY;</code></pre>

在支持外围确定数据库角色受支持之后，RDS Custom for Oracle 数据库实例将在 15 秒内变为可用状态。
</td></tr>
</table>

事件 ID	配置	RDS 事件消息	操作
SP-O3003	数据库运行状况	Oracle 数据库的 SMON 进程处于僵尸状态。要解决此问题，请手动恢复 EC2 实例 [*ec2_id*] 上的数据库，打开数据库，然后立即对其进行备份。如需更多帮助，请联系Amazon Web Services 支持。	支持外围会监控数据库实例状态。其还将监控前一小时和一天中发生的重启次数。当实例处于仍然存在的状态时，您会收到通知，但您无法与其交互。 要设置支持边界，请评估您的实例状态 1. 登录到主机并确定数据库状态。 `ps -eo pid,state,command | grep smon` 2. 如有必要，请重新启动数据库实例。如果重新启动失败，请继续执行下一步。 3. 如有必要，请重新启动 EC2 主机。 重启数据库实例后，RDS Custom 代理会检测到数据库实例不再处于无响应状态。然后，它会通知支持外围以重新评估数据库实例状态。

事件 ID	配置	RDS 事件消息	操作

SP-O3004	数据库日志模式	EC2 实例 [*ec2_id*] 上的数据库日志模式已更改为 [*value_b*]。要解决此问题，请将日志模式设置为 [*value_a*]。	将数据库实例日志模式更改为 **ARCHIVELOG**
			
			1. 登录您的 EC2 主机。
			2. 连接到数据库并运行以下语句：
			`SELECT LOG_MODE FROM V$DATABASE;`
			或者，也可以在 SQL*Plus 中运行以下命令：
			`ARCHIVE LOG LIST`
			3. 运行以下 SQL*Plus 命令以启动一致关机。
			`SHUTDOWN IMMEDIATE`
			
			RDS Custom 代理将自动重启数据库实例并将日志模式设置为 `ARCHIVELOG` 。数据库实例将在 30 分钟内可用。

SP-O3005	Oracle 主路径	EC2 实例 [*ec2_id*] 上的 Oracle 主目录已更改为 *new_path*。要解决此问题，请将设置恢复为 *old_path*。	

事件 ID	配置	RDS 事件消息	操作

SP-O3006	数据库唯一名称	EC2 实例 [*ec2_id*] 上的数据库唯一名称已更改为 *new_value* 。要解决此问题，请将名称恢复为 *old_value* 。	更改数据库实例的数据库唯一名称
			
			1. 登录您的 EC2 主机。
			2. 连接到数据库并运行以下语句：
			`SELECT DB_UNIQUE_NAME FROM V$DATABASE;`
			3. 使用命令 `ALTER SYSTEM SET DB_UNIQUE _NAME` 指定源数据库唯一名称。
			4. 运行以下 SQL 语句启动一致关机。
			`SHUTDOWN IMMEDIATE;`
			
			RDS Custom 代理将自动重启数据库实例并将日志模式设置为 `ARCHIVELOG` 。数据库实例将在 30 分钟内可用。

排查 RDS Custom for Oracle 的升级问题

RDS Custom for Oracle 实例的升级可能会失败。接下来，您可以找到一些技术，以在升级 RDS Custom for Oracle 数据库实例时使用：

- 检查数据库实例上 `/tmp` 目录中的升级输出日志文件。日志的名称取决于数据库引擎版本。例如，您可能会看到包含字符串 `catupgrd` 或 `catup` 的日志。
- 检查位于 `/rdsdbdata/log/trace` 目录中的 `alert.log` 文件。
- 请在 `root` 目录中运行以下 `grep` 命令来跟踪升级操作系统的过程。此命令显示日志文件的写入位置并确定升级过程的状态。

  ```
  ps -aux | grep upg
  ```

  下面显示了示例输出。

```
root      18884  0.0  0.0 235428  8172 ?        S<   17:03   0:00 /usr/bin/
sudo -u rdsdb /rdsdbbin/scripts/oracle-control ORCL op_apply_upgrade_sh RDS-
UPGRADE/2.upgrade.sh
rdsdb     18886  0.0  0.0 153968 12164 ?        S<   17:03   0:00 /usr/bin/perl -T -
w /rdsdbbin/scripts/oracle-control ORCL op_apply_upgrade_sh RDS-UPGRADE/2.upgrade.sh
rdsdb     18887  0.0  0.0 113196  3032 ?        S<   17:03   0:00 /bin/sh /rdsdbbin/
oracle/rdbms/admin/RDS-UPGRADE/2.upgrade.sh
rdsdb     18900  0.0  0.0 113196  1812 ?        S<   17:03   0:00 /bin/sh /rdsdbbin/
oracle/rdbms/admin/RDS-UPGRADE/2.upgrade.sh
rdsdb     18901  0.1  0.0 167652 20620 ?        S<   17:03   0:07 /rdsdbbin/oracle/
perl/bin/perl catctl.pl -n 4 -d /rdsdbbin/oracle/rdbms/admin -l /tmp catupgrd.sql
root      29944  0.0  0.0 112724  2316 pts/0    S+   18:43   0:00 grep --color=auto
 upg
```

- 请运行以下 SQL 查询以验证组件的当前状态，以便查找数据库版本和数据库实例上安装的选项。

```
SET LINESIZE 180
COLUMN COMP_ID FORMAT A15
COLUMN COMP_NAME FORMAT A40 TRUNC
COLUMN STATUS FORMAT A15 TRUNC
SELECT COMP_ID, COMP_NAME, VERSION, STATUS FROM DBA_REGISTRY ORDER BY 1;
```

输出与以下内容类似。

```
COMP_NAME                                STATUS               PROCEDURE
---------------------------------------- --------------------
 --------------------------------------------------
Oracle Database Catalog Views            VALID
 DBMS_REGISTRY_SYS.VALIDATE_CATALOG
Oracle Database Packages and Types       VALID
 DBMS_REGISTRY_SYS.VALIDATE_CATPROC
Oracle Text                              VALID                VALIDATE_CONTEXT
Oracle XML Database                      VALID                DBMS_REGXDB.VALIDATEXDB

4 rows selected.
```

- 请运行以下 SQL 查询以检查是否存在可能干扰升级过程的无效对象。

```
SET PAGES 1000 LINES 2000
COL OBJECT FOR A40
SELECT SUBSTR(OWNER,1,12) OWNER,
```

```
        SUBSTR(OBJECT_NAME,1,30) OBJECT,
        SUBSTR(OBJECT_TYPE,1,30) TYPE, STATUS,
        CREATED
FROM    DBA_OBJECTS
WHERE   STATUS <>'VALID'
AND     OWNER IN ('SYS','SYSTEM','RDSADMIN','XDB');
```

排查 RDS Custom for Oracle 的副本提升问题

您可以使用控制台、`promote-read-replica` Amazon CLI 命令或 `PromoteReadReplica` API 在 RDS Custom for Oracle 中提升托管式 Oracle 副本。如果您删除了主数据库实例，并且所有副本都运行正常，则 RDS Custom for Oracle 会自动将您的托管式副本提升为独立实例。如果副本已暂停自动化或超出支持范围，则必须先修复副本，RDS Custom 才能自动对其进行提升。有关更多信息，请参阅 将 RDS Custom for Oracle 副本提升为独立的数据库实例。

在以下情况下，副本提升工作流可能会卡住：

- 主数据库实例处于 `STORAGE_FULL` 状态。
- 主数据库无法对其所有的在线重做日志归档。
- Oracle 副本和主数据库上的存档重做日志文件之间存在差距。

响应卡住的工作流

1. 同步 Oracle 副本数据库实例上的重做日志差距。
2. 强制将只读副本提升为最新应用的重做日志。在 SQL*Plus 中运行以下命令：

```
ALTER DATABASE ACTIVATE STANDBY DATABASE;
SHUTDOWN IMMEDIATE
STARTUP
```

3. 联系 Amazon Web Services 支持 并要求他们将您的数据库实例移到 `available` 状态。

Amazon RDS Custom for Oracle 的已知问题

使用 RDS Custom for Oracle 时，请注意数据库实例存在的以下问题。

- 不支持调整根卷或 dbbin 卷的大小。

> **Warning**
> 强烈建议您不要手动调整根卷或 dbbin 卷的大小。建议您将所有配置存储在修补后仍然存在的数据卷中，并且仅使用 RDS 扩展存储 API 调整卷的大小。

- 当数据库实例位于较旧的 AMI（比如使用 Oracle Linux 7 的 AMI）上时，某些 RDS API 可能会被阻止。要解决此问题，请使用操作系统修补功能将数据库实例修补到最新的 AMI。有关更多信息，请参阅 CEV 升级选项。
- 在执行 RDS 操作之前，请确保您的 Amazon Web Services 账户中有足够的计算和存储配额。
- 如果数据库处于创建状态，并且您主动登录该数据库或 Amazon EC2 主机并运行命令，则数据库创建可能无法完成。
- 由于只读副本问题，目前不支持控制文件多路复用。在创建只读副本之前，请确保在源数据库上的 `CONTROL_FILES` 初始化参数中仅指定一个文件名。
- 您无法将数据库模式从 `PHYSICAL STANDBY`（已装载或只读）更改为 `SNAPSHOT STANDBY`（转换为读/写）。
- 如果 Amazon Web Services 账户是具有服务控制策略（SCP）的 Amazon Organization 的一部分，并且 SCP 包含条件键，则可能无法创建 RDS Custom for Oracle 数据库实例，并会显示以下错误：

```
You can't create the DB instance because of incompatible resources.
The IAM instance profile role [AWSRDSCustomInstanceRole1-us-east-1] is missing the
 following permissions:
EFFECT [Allow] on ACTION(S) [ssm:DescribeAssociation, ssm:DescribeDocument,
 ssm:GetConnectionStatus,
  ssm:GetDeployablePatchSnapshotForInstance, ssmmessages:OpenControlChannel,
 ssm:GetParameters,
  ssm:ListInstanceAssociations, ssm:PutConfigurePackageResult,
 ssmmessages:CreateControlChannel,
  ssm:GetParameter, ssm:UpdateAssociationStatus, ssm:GetManifest,
 ssmmessages:CreateDataChannel,
  ssm:PutInventory, ssm:UpdateInstanceInformation, ssm:DescribeInstanceInformation,
  ssmmessages:OpenDataChannel, ssm:GetDocument, ssm:ListAssociations,
 ssm:PutComplianceItems,
```

```
 ssm:UpdateInstanceAssociationStatus] for RESOURCE(S) [], EFFECT [Allow] on
ACTION(S) [ec2messages:DeleteMessage,
 ec2messages:FailMessage, ec2messages:GetEndpoint, ec2messages:AcknowledgeMessage,
ec2messages:GetMessages,
 ec2messages:SendReply] for RESOURCE(S) [], EFFECT [Allow] on ACTION(S)
[logs:CreateLogStream,
 logs:DescribeLogStreams, logs:PutRetentionPolicy, logs:PutLogEvents]
```

要解决此问题，请向 Amazon Web Services 支持 创建支持工单。

数据库用户帐户存在的已知问题

请注意以下问题：

- 不要删除以字符串 RDS 开头的数据库用户账户，例如 RDSADMIN 和 RDS_DATAGUARD。RDS Custom for Oracle 使用 RDS 账户执行自动化操作。如果删除此用户帐户，RDS Custom 会将实例移至不受支持的配置状态。
- 不能使用 `ModifyDBInstance` API 更改 RDS Custom for Oracle 数据库实例的主用户名。
- RDS Custom for Oracle 会轮换所有数据库实例的用户账户凭证。有关更多信息，请参阅 轮换 RDS Custom for Oracle 凭证以遵循合规性计划。如果使用本地主/备用配置，则凭证轮换可能会影响以下资源：
  - 手动创建的备用 RDS Custom for Oracle 实例

    要解决此问题，请删除手动创建的备用数据库，然后使用 API 调用创建 Oracle 只读副本。手动管理手动创建的备用数据库的密钥，使其与源数据库实例相匹配。
  - 手动创建的跨区域只读副本

    要解决此问题，请手动保存密钥，使其与主数据库实例相匹配。

参数和配置文件存在的已知问题

- 在执行扩展计算、操作系统升级以及 RDS Custom 替换根卷的其他操作之后，必须配置 `crontab` 文件。强烈建议您保留 `crontab` 的备份。
- 配置 `listener.ora` 文件时，请注意以下准则：
  - 确保该文件中的每个条目都位于一行中。这种方法可以避免在创建实例期间出现缩进问题。
  - 确保 GLOBAL_DBNAME 等于 SID_NAME 的值。
  - 确保 LISTENER 的值遵循命名约定 L_*dbname*_001。

- 确保 `listener.ora` 文件保持与数据库名称的联系。RDS Custom 使用此联系来验证数据库的启动。如果您错误地修改此文件，则诸如扩展计算或修补之类的操作可能会失败。

  以下示例显示正确配置的 `listener.ora` 参数。

  ```
  ADR_BASE_L_ORCL_001=/rdsdbdata/log/
  USE_SID_AS_SERVICE_L_ORCL_001=ON
  SID_LIST_L_ORCL_001=(SID_LIST = (SID_DESC = (SID_NAME = ORCL)(GLOBAL_DBNAME = ORCL)
   (ORACLE_HOME = /rdsdbbin/oracle.19.custom.r1.EE.1)))
  SUBSCRIBE_FOR_NODE_DOWN_EVENT_L_ORCL_001=OFF
  L_ORCL_001=(DESCRIPTION_LIST = (DESCRIPTION = (ADDRESS = (PROTOCOL = TCP)(PORT =
   XXXX)(HOST = x.x.x.x))) (DESCRIPTION = (ADDRESS = (PROTOCOL = TCP)(PORT = XXXX)
  (HOST = 127.0.0.1))))
  ```
- 在服务器参数文件或初始化参数文件中不支持注释。
- 您必须在服务器参数文件（`/rdsdbdata/config/oracle_pfile`）中声明以下初始化参数：
  - `MEMORY_MAX_TARGET`
  - `MEMORY_TARGET`
  - `PGA_AGGREGATE_TARGET`
  - `PROCESSES`
  - `SGA_TARGET`
  - `USE_LARGE_PAGES`

  如果未在 `/rdsdbdata/config/oracle_pfile` 中声明上述参数，则只读副本创建和扩展计算可能会失败。
- 您无法删除配置文件（例如服务器参数文件、审计文件、`listener.ora`、`tnsnames.ora` 或 `sqlnet.ora`）的符号链接。也无法修改这些文件的目录结构。RDS Custom 自动化要求这些文件存在于特定的目录结构中。

  要从初始化参数文件创建服务器参数文件，请使用以下语法。

  ```
  CREATE SPFILE='/rdsdbdata/admin/$ORACLE_SID/pfile/spfile$ORACLE_SID.ora'
      FROM PFILE='/rdsdbdata/config/oracle_pfile';
  ```

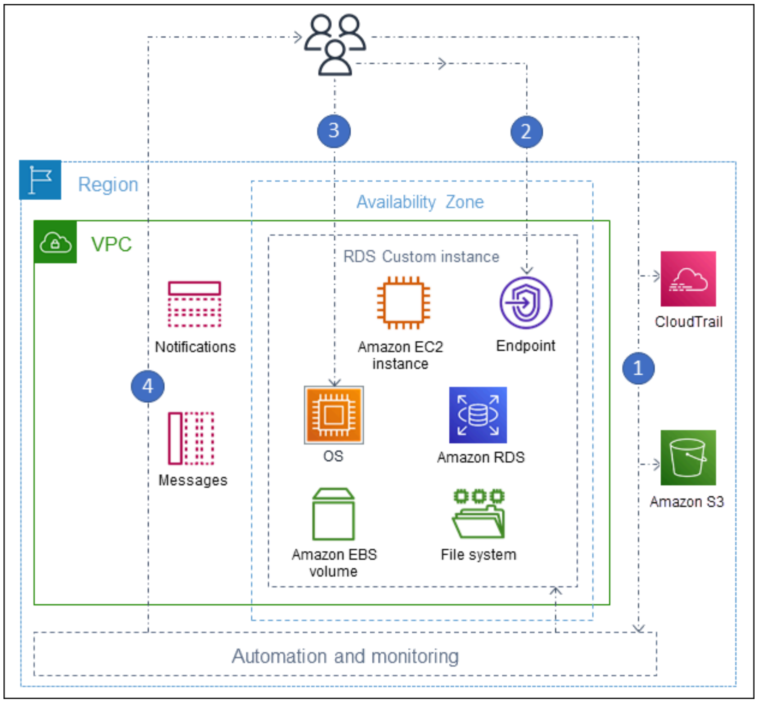
使用 RDS Custom for SQL Server

您可在以下内容中找到有关创建、管理和维护 RDS Custom for SQL Server 数据库实例的说明。

主题

RDS Custom for SQL Server 工作流

下图显示了 RDS Custom for SQL Server 的典型工作流。



步骤如下所示：

1. 从 RDS Custom 提供的引擎版本创建 RDS Custom for SQL Server 数据库实例。

   有关更多信息，请参阅 创建 RDS Custom for SQL Server 数据库实例。
2. 将应用程序连接到 RDS Custom 数据库实例端点。

   有关更多信息，请参阅 使用 Amazon Systems Manager 连接到 RDS Custom 数据库实例 和 使用 RDP 连接到 RDS Custom 数据库实例。
3. （可选）访问主机以自定义您的软件。
4. 监控 RDS Custom 自动化生成的通知和消息。

为 RDS Custom 创建数据库实例

您可以使用 `create-db-instance` 命令创建 RDS Custom 数据库实例。该过程与创建 Amazon RDS 实例类似。但是，有些参数是不同的。有关更多信息，请参阅 为 Amazon RDS Custom for SQL Server 创建并连接到数据库实例。

数据库连接

与 Amazon RDS 数据库实例一样，您的 RDS Custom for SQL Server 数据库实例驻留在 VPC 中。您的应用程序使用 SQL Server 管理套件（SSMS）等客户端连接到 RDS Custom 实例，就像在 RDS for SQL Server 中一样。

RDS Custom 自定义

您可以访问 RDS Custom 主机来安装或自定义软件。为避免更改与 RDS Custom 自动化之间发生冲突，您可以在指定的时间内暂停自动化。在此期间，RDS Custom 不执行监控或实例恢复。在此时期结束时，RDS Custom 将恢复完全自动化。有关更多信息，请参阅 暂停和恢复 RDS Custom 自动化。

Amazon RDS Custom for SQL Server 的要求和限制

您可以在下面找到 Amazon RDS Custom for SQL Server 要求和限制的摘要以便快速参考。要求和限制也出现在相关章节中。

主题

区域和版本可用性

功能可用性和支持因每个数据库引擎的特定版本以及 Amazon Web Services 区域而异。有关适用于 Amazon RDS Custom for SQL Server 的 Amazon RDS 的版本和区域可用性的更多信息，请参阅支持 RDS Custom for SQL Server 的区域和数据库引擎。

RDS Custom for SQL Server 的一般要求

确保遵循 Amazon RDS Custom for SQL Server 的以下要求：

- 使用 RDS Custom for SQL Server 的数据库实例类支持 中显示的实例类。唯一支持的存储类型是 gp2、gp3、io1 和 io2 Block Express 类型的固态硬盘（SSD）。io1、gp2 和 gp3 的最大存储限制为 16 TiB，而 io2 支持 64 TiB。
- 确保您有创建 RDS Custom 数据库实例的对称 Amazon KMS 密钥。有关更多信息，请参阅 请确保您有对称的加密 Amazon KMS 密钥。
- 确保创建 Amazon Identity and Access Management (IAM) 角色和实例配置文件。有关更多信息，请参阅手动创建您的 IAM 角色和实例配置文件和使用 Amazon Web Services Management Console自动创建实例配置文件。
- 确保提供 RDS Custom 可用于访问其他 Amazon Web Services 服务的联网配置。有关特定要求，请参阅 步骤 2：配置网络、实例配置文件和加密。

- RDS Custom 和 Amazon RDS 数据库实例的总数不能超过您的配额限制。例如，如果您的配额为 40 个数据库实例，则可以有 20 个 RDS Custom for SQL Server 数据库实例和 20 个 Amazon RDS 数据库实例。
- RDS Custom 会自动创建名称以 `do-not-delete-rds-custom-` 开头的 Amazon CloudTrail 跟踪记录。RDS Custom 支持外围依赖于来自 CloudTrail 的事件来确定您的操作是否影响 RDS Custom 自动化。RDS Custom 会在您创建第一个数据库实例时创建跟踪记录。要使用已经存在的 CloudTrail，请联系Amazon支持人员。有关更多信息，请参阅 Amazon CloudTrail。

RDS Custom for SQL Server 的数据库实例类支持

使用 describe-orderable-db-instance-options 命令，检查您的区域中是否支持该数据库实例类。

RDS Custom for SQL Server 支持下表所示的数据库实例类：

SQL Server 版本	RDS Custom 支持
Enterprise Edition	db.r5.xlarge–db.r5.24xlarge db.r5b.xlarge–db.r5b.24xlarge db.m5.xlarge–db.m5.24xlarge db.r6i.xlarge–db.r6i.32xlarge db.m6i.xlarge–db.m6i.32xlarge db.x2iedn.xlarge–db.x2iedn.32xlarge
标准版	db.r5.large–db.r5.24xlarge db.r5b.large–db.r5b.8xlarge db.m5.large–db.m5.24xlarge db.r6i.large–db.r6i.8xlarge db.m6i.large–db.m6i.8xlarge

SQL Server 版本	RDS Custom 支持
	db.x2iedn.xlarge–db.x2iedn.8xlarge
开发人员版本	db.r5.xlarge–db.r5.24xlarge db.r5b.xlarge–db.r5b.24xlarge db.m5.xlarge–db.m5.24xlarge db.r6i.xlarge–db.r6i.32xlarge db.m6i.xlarge–db.m6i.32xlarge db.x2iedn.xlarge–db.x2iedn.32xlarge
Web 版	db.r5.large–db.r5.4xlarge db.m5.large–db.m5.4xlarge db.r6i.large–db.r6i.4xlarge db.m6i.large–db.m6i.4xlarge db.r5b.large–db.r5b.4xlarge

以下建议适用于 db.x2iedn 类类型：

- 创建时，本地存储是未分配的原始设备。在使用具有此实例类的数据库实例之前，必须挂载并格式化本地存储。之后，在其之上配置 `tempdb` 以确保最佳性能。有关更多信息，请参阅 [Optimize tempdb performance in Amazon RDS Custom for SQL Server using local instance storage]。
- 当您运行数据库实例操作 [例如扩展计算、实例更换、快照还原或时间点故障恢复（PITR）] 时，本地存储会恢复到其原始和未分配状态。在这些情况下，必须重新挂载、重新格式化和重新配置驱动器和 `tempdb` 来还原功能。
- 对于多可用区实例，我们建议您在备用数据库实例上执行配置。这样，如果发生故障转移，系统可以继续运行而不会出现问题，因为配置已在备用实例上就位。

RDS Custom for SQL Server 的限制

以下限制适用于 RDS Custom for SQL Server：

- 您无法在 RDS Custom for SQL Server 的 Amazon RDS 数据库实例中创建只读副本。但是，您可以使用多可用区部署自动配置高可用性。有关更多信息，请参阅 管理 RDS Custom for SQL Server 的多可用区部署。
- 您无法修改现有 RDS Custom for SQL Server 数据库实例的数据库实例标识符。
- 对于不是使用自定义引擎版本（CEV）创建的 RDS Custom for SQL Server 数据库实例，不能保证保留对 Microsoft Windows 操作系统进行的更改。例如，当您启动快照或时间点还原操作时，将丢失这些更改。如果 RDS Custom for SQL Server 数据库实例是使用 CEV 创建的，则会保留这些更改。
- 并非所有选项都受支持。例如，当您创建 RDS Custom for SQL Server 数据库实例时，您无法执行以下操作：
  - 更改数据库实例类中的 CPU 内核数和每个内核的线程数。
  - 启用存储弹性伸缩。
  - 指定自己的数据库参数组、选项组或字符集。
  - 开启性能详情。
  - 开启自动次要版本升级。
- 最大的数据库实例存储空间为 64TiB。
- 您无法将 RDS 代理与 RDS Custom for SQL Server 一起使用。
- 您无法将 `describe-reserved-db-instances` API 用于 RDS Custom for SQL Server 数据库实例。

设置 RDS Custom for SQL Server 数据库实例的字符集和排序规则

概述

使用 RDS Custom for SQL Server 数据库实例，可以配置字符集和排序规则设置，来确定如何对数据进行存储和排序。字符集定义支持哪些字符，而排序规则指定用于对数据进行排序和比较的规则。为处理多语言数据或具有特定排序要求的应用程序设置适当的字符集和排序规则至关重要。例如，您可能需要处理重音字符并定义语言特定的排序规则，或者在不同区域设置之间保持数据完整性。以下各节提供有关 RDS Custom for SQL Server 数据库实例的字符集和排序规则支持的信息。

RDS Custom for SQL Server 支持各种服务器排序规则，包括传统编码和 UTF-8 编码，适用于 SQL_Latin、日语、德语和阿拉伯语区域设置。原定设置的服务器排序规则为

`SQL_Latin1_General_CP1_CI_AS`，但您可以选择其他支持的排序规则来使用。您可以使用与 RDS for SQL Server 所用的相同过程来选择排序规则。有关更多信息，请参阅 管理 Amazon RDS for Microsoft SQL Server 的排序规则和字符集。

注意事项

在 RDS Custom for SQL Server 上使用服务器排序规则时，以下要求和限制适用：

- 您可以在创建 RDS Custom for SQL Server 数据库实例时设置服务器排序规则。创建数据库实例后，您无法修改服务器级别的排序规则。
- 从数据库快照还原或时间点故障恢复（PITR）期间，您无法修改服务器级别的排序规则。
- 当您从 RDS Custom for SQL Server CEV 创建数据库实例时，该数据库实例不会继承来自 CEV 的服务器排序规则。而是使用原定设置的服务器排序规则 `SQL_Latin1_General_CP1_CI_AS`。如果您已在 RDS Custom for SQL Server CEV 上配置了非原定设置服务器排序规则，并且想要在新数据库实例上使用相同的服务器排序规则，请务必在从 CEV 创建数据库实例时选择该相同的排序规则。

> **Note**
>
> 如果您在创建数据库实例时选择的排序规则与 CEV 的排序规则不同，则将重建新 RDS Custom for SQL Server 数据库实例上的 Microsoft SQL Server 系统数据库，以使用更新后的排序规则。重建过程仅在新的 RDS Custom for SQL Server 数据库实例上执行，对 CEV 本身没有影响。重建系统数据库后，您之前在 CEV 上对系统数据库所做的任何修改都不会保留在新的 RDS Custom for SQL Server 数据库实例上。一些修改的示例包括 `master` 数据库中的用户定义对象、`msdb` 数据库中的计划作业或对 CEV 上 `model` 数据库中原定设置数据库设置的更改。创建新的 RDS Custom for SQL Server 数据库实例后，您可以手动重新创建您的修改。

- 当您从 RDS Custom for SQL Server 自定义引擎版本（CEV）创建数据库实例并选择与 CEV 所用不同的排序规则时，请确保用于创建 CEV 的黄金映像（AMI）满足以下要求，以便重建新数据库实例上的 Microsoft SQL Server 系统数据库：
  - 对于 SQL Server 2022，请确保 `setup.exe` 文件位于以下路径中：`C:\Program Files\Microsoft SQL Server\160\Setup Bootstrap\SQL2022\setup.exe`
  - 对于 SQL Server 2019，请确保 `setup.exe` 文件位于以下路径中：`C:\Program Files\Microsoft SQL Server\150\Setup Bootstrap\SQL2019\setup.exe`
  - `master`、`model` 和 `msdb` 数据库的数据和日志模板的副本必须存在于其原定设置位置。有关更多信息，请参阅 Microsoft 公共文档中的重建系统数据库。

  - 确保您的 SQL Server 数据库引擎使用 `NT Service\MSSQLSERVER` 或 `NT AUTHORITY\NETWORK SERVICE` 作为服务账户。在为数据库实例配置非原定设置服务器排序规则时，任何其他账户对 `C:\` 驱动器都没有所需的权限。
- 如果为新数据库实例选择的服务器排序规则与在 CEV 上配置的服务器排序规则相同，则新 RDS Custom for SQL Server 数据库实例上的 Microsoft SQL Server 系统数据库不会经历重建过程。您之前在 CEV 上对系统数据库所做的任何修改都将自动保留到新的 RDS Custom for SQL Server 数据库实例中。

支持的排序规则

您可以将排序规则设置为下表中列出的值之一。

排序规则	描述
Arabic_100_BIN	Arabic-100，二进制排序
Arabic_100_BIN2	Arabic-100，二进制代码点比较排序
Arabic_100_CI_AI	Arabic-100，不区分大小写，不区分重音，不区分假名，不区分全半角
Arabic_100_CI_AI_KS	Arabic-100，不区分大小写，不区分重音，区分假名，不区分全半角
Arabic_100_CI_AI_KS_SC	Arabic-100，不区分大小写，不区分重音，区分假名，不区分全半角，补充字符
Arabic_100_CI_AI_KS_SC_UTF8	Arabic-100，不区分大小写，不区分重音，区分假名，不区分全半角，补充字符，UTF8
Arabic_100_CI_AI_KS_WS	Arabic-100，不区分大小写，不区分重音，区分假名，区分全半角
Arabic_100_CI_AI_KS_WS_SC	Arabic-100，不区分大小写，不区分重音，区分假名，区分全半角，补充字符
Arabic_100_CI_AI_KS_WS_SC_UTF8	Arabic-100，不区分大小写，不区分重音，区分假名，区分全半角，补充字符，UTF8

排序规则	描述
Arabic_100_CI_AI_SC	Arabic-100，不区分大小写，不区分重音，不区分假名，不区分全半角，补充字符
Arabic_100_CI_AI_SC_UTF8	Arabic-100，不区分大小写，不区分重音，不区分假名，不区分全半角，补充字符，UTF8
Arabic_100_CI_AI_WS	Arabic-100，不区分大小写，不区分重音，不区分假名，区分全半角
Arabic_100_CI_AI_WS_SC	Arabic-100，不区分大小写，不区分重音，不区分假名，区分全半角，补充字符
Arabic_100_CI_AI_WS_SC_UTF8	Arabic-100，不区分大小写，不区分重音，不区分假名，区分全半角，补充字符，UTF8
Arabic_100_CI_AS	Arabic-100，不区分大小写，区分重音，不区分假名，不区分全半角
Arabic_100_CI_AS_KS	Arabic-100，不区分大小写，区分重音，区分假名，不区分全半角
Arabic_100_CI_AS_KS_SC	Arabic-100，不区分大小写，区分重音，区分假名，不区分全半角，补充字符
Arabic_100_CI_AS_KS_SC_UTF8	Arabic-100，不区分大小写，区分重音，区分假名，不区分全半角，补充字符，UTF8
Arabic_100_CI_AS_KS_WS	Arabic-100，不区分大小写，区分重音，区分假名，区分全半角
Arabic_100_CI_AS_KS_WS_SC	Arabic-100，不区分大小写，区分重音，区分假名，区分全半角，补充字符
Arabic_100_CI_AS_KS_WS_SC_UTF8	Arabic-100，不区分大小写，区分重音，区分假名，区分全半角，补充字符，UTF8
Arabic_100_CI_AS_SC	Arabic-100，不区分大小写，区分重音，不区分假名，不区分全半角，补充字符

排序规则	描述
Arabic_100_CI_AS_SC_UTF8	Arabic-100，不区分大小写，区分重音，不区分假名，不区分全半角，补充字符，UTF8
Arabic_100_CI_AS_WS	Arabic-100，不区分大小写，区分重音，不区分假名，区分全半角
Arabic_100_CI_AS_WS_SC	Arabic-100，不区分大小写，区分重音，不区分假名，区分全半角，补充字符
Arabic_100_CI_AS_WS_SC_UTF8	Arabic-100，不区分大小写，区分重音，不区分假名，区分全半角，补充字符，UTF8
Arabic_100_CS_AI	Arabic-100，区分大小写，不区分重音，不区分假名，不区分全半角
Arabic_100_CS_AI_KS	Arabic-100，区分大小写，不区分重音，区分假名，不区分全半角
Arabic_100_CS_AI_KS_SC	Arabic-100，区分大小写，不区分重音，区分假名，不区分全半角，补充字符
Arabic_100_CS_AI_KS_SC_UTF8	Arabic-100，区分大小写，不区分重音，区分假名，不区分全半角，补充字符，UTF8
Arabic_100_CS_AI_KS_WS	Arabic-100，区分大小写，不区分重音，区分假名，区分全半角
Arabic_100_CS_AI_KS_WS_SC	Arabic-100，区分大小写，不区分重音，区分假名，区分全半角，补充字符
Arabic_100_CS_AI_KS_WS_SC_UTF8	Arabic-100，区分大小写，不区分重音，区分假名，区分全半角，补充字符，UTF8
Arabic_100_CS_AI_SC	Arabic-100，区分大小写，不区分重音，不区分假名，不区分全半角，补充字符
Arabic_100_CS_AI_SC_UTF8	Arabic-100，区分大小写，不区分重音，不区分假名，不区分全半角，补充字符，UTF8

排序规则	描述
Arabic_100_CS_AI_WS	Arabic-100，区分大小写，不区分重音，不区分假名，区分全半角
Arabic_100_CS_AI_WS_SC	Arabic-100，区分大小写，不区分重音，不区分假名，区分全半角，补充字符
Arabic_100_CS_AI_WS_SC_UTF8	Arabic-100，区分大小写，不区分重音，不区分假名，区分全半角，补充字符，UTF8
Arabic_100_CS_AS	Arabic-100，区分大小写，区分重音，不区分假名，不区分全半角
Arabic_100_CS_AS_KS	Arabic-100，区分大小写，区分重音，区分假名，不区分全半角
Arabic_100_CS_AS_KS_SC	Arabic-100，区分大小写，区分重音，区分假名，不区分全半角，补充字符
Arabic_100_CS_AS_KS_SC_UTF8	Arabic-100，区分大小写，区分重音，区分假名，不区分全半角，补充字符，UTF8
Arabic_100_CS_AS_KS_WS	Arabic-100，区分大小写，区分重音，区分假名，区分全半角
Arabic_100_CS_AS_KS_WS_SC	Arabic-100，区分大小写，区分重音，区分假名，区分全半角，补充字符
Arabic_100_CS_AS_KS_WS_SC_UTF8	Arabic-100，区分大小写，区分重音，区分假名，区分全半角，补充字符，UTF8
Arabic_100_CS_AS_SC	Arabic-100，区分大小写，区分重音，不区分假名，不区分全半角，补充字符
Arabic_100_CS_AS_SC_UTF8	Arabic-100，区分大小写，区分重音，不区分假名，不区分全半角，补充字符，UTF8
Arabic_100_CS_AS_WS	Arabic-100，区分大小写，区分重音，不区分假名，区分全半角

排序规则	描述
Arabic_100_CS_AS_WS_SC	Arabic-100，区分大小写，区分重音，不区分假名，区分全半角，补充字符
Arabic_100_CS_AS_WS_SC_UTF8	Arabic-100，区分大小写，区分重音，不区分假名，区分全半角，补充字符，UTF8
Arabic_BIN	Arabic，二进制排序
Arabic_BIN2	Arabic，二进制代码点比较排序
Arabic_CI_AI	Arabic，不区分大小写，不区分重音，不区分假名，不区分全半角
Arabic_CI_AI_KS	Arabic，不区分大小写，不区分重音，区分假名，不区分全半角
Arabic_CI_AI_KS_WS	Arabic，不区分大小写，不区分重音，区分假名，区分全半角
Arabic_CI_AI_WS	Arabic，不区分大小写，不区分重音，不区分假名，区分全半角
Arabic_CI_AS	Arabic，不区分大小写，区分重音，不区分假名，不区分全半角
Arabic_CI_AS_KS	Arabic，不区分大小写，区分重音，区分假名，不区分全半角
Arabic_CI_AS_KS_WS	Arabic，不区分大小写，区分重音，区分假名，区分全半角
Arabic_CI_AS_WS	Arabic，不区分大小写，区分重音，不区分假名，区分全半角
Arabic_CS_AI	Arabic，区分大小写，不区分重音，不区分假名，不区分全半角

排序规则	描述
Arabic_CS_AI_KS	Arabic，区分大小写，不区分重音，区分假名，不区分全半角
Arabic_CS_AI_KS_WS	Arabic，区分大小写，不区分重音，区分假名，区分全半角
Arabic_CS_AI_WS	Arabic，区分大小写，不区分重音，不区分假名，区分全半角
Arabic_CS_AS	Arabic，区分大小写，区分重音，不区分假名，不区分全半角
Arabic_CS_AS_KS	Arabic，区分大小写，区分重音，区分假名，不区分全半角
Arabic_CS_AS_KS_WS	Arabic，区分大小写，区分重音，区分假名，区分全半角
Arabic_CS_AS_WS	Arabic，区分大小写，区分重音，不区分假名，区分全半角
Chinese_PRC_BIN2	Chinese-PRC，二进制代码点比较排序
Chinese_PRC_CI_AS	Chinese-PRC，不区分大小写，区分重音，不区分假名，不区分全半角
Danish_Norwegian_CI_AS	Danish-Norwegian，不区分大小写，区分重音，不区分假名，不区分全半角
Finnish_Swedish_CI_AS	Finnish-Swedish，不区分大小写，区分重音，不区分假名，不区分全半角
French_CI_AS	French，不区分大小写，区分重音，不区分假名，不区分全半角
German_PhoneBook_100_BIN	German-PhoneBook-100，二进制排序

排序规则	描述
German_PhoneBook_100_BIN2	German-PhoneBook-100，二进制代码点比较排序
German_PhoneBook_100_CI_AI	German-PhoneBook-100，不区分大小写，不区分重音，不区分假名，不区分全半角
German_PhoneBook_100_CI_AI_KS	German-PhoneBook-100，不区分大小写，不区分重音，区分假名，不区分全半角
German_PhoneBook_100_CI_AI_KS_SC	German-PhoneBook-100，不区分大小写，不区分重音，区分假名，不区分全半角，补充字符
German_PhoneBook_100_CI_AI_KS_SC_UTF8	German-PhoneBook-100，不区分大小写，不区分重音，区分假名，不区分全半角，补充字符，UTF8
German_PhoneBook_100_CI_AI_KS_WS	German-PhoneBook-100，不区分大小写，不区分重音，区分假名，区分全半角
German_PhoneBook_100_CI_AI_KS_WS_SC	German-PhoneBook-100，不区分大小写，不区分重音，区分假名，区分全半角，补充字符
German_PhoneBook_100_CI_AI_KS_WS_SC_UTF8	German-PhoneBook-100，不区分大小写，不区分重音，区分假名，区分全半角，补充字符，UTF8
German_PhoneBook_100_CI_AI_SC	German-PhoneBook-100，不区分大小写，不区分重音，不区分假名，不区分全半角，补充字符
German_PhoneBook_100_CI_AI_SC_UTF8	German-PhoneBook-100，不区分大小写，不区分重音，不区分假名，不区分全半角，补充字符，UTF8
German_PhoneBook_100_CI_AI_WS	German-PhoneBook-100，不区分大小写，不区分重音，不区分假名，区分全半角
German_PhoneBook_100_CI_AI_WS_SC	German-PhoneBook-100，不区分大小写，不区分重音，不区分假名，区分全半角，补充字符

排序规则	描述
German_PhoneBook_100_CI_AI_WS_SC_UTF8	German-PhoneBook-100，不区分大小写，不区分重音，不区分假名，区分全半角，补充字符，UTF8
German_PhoneBook_100_CI_AS	German-PhoneBook-100，不区分大小写，区分重音，不区分假名，不区分全半角
German_PhoneBook_100_CI_AS_KS	German-PhoneBook-100，不区分大小写，区分重音，区分假名，不区分全半角
German_PhoneBook_100_CI_AS_KS_SC	German-PhoneBook-100，不区分大小写，区分重音，区分假名，不区分全半角，补充字符
German_PhoneBook_100_CI_AS_KS_SC_UTF8	German-PhoneBook-100，不区分大小写，区分重音，区分假名，不区分全半角，补充字符，UTF8
German_PhoneBook_100_CI_AS_KS_WS	German-PhoneBook-100，不区分大小写，区分重音，区分假名，区分全半角
German_PhoneBook_100_CI_AS_KS_WS_SC	German-PhoneBook-100，不区分大小写，区分重音，区分假名，区分全半角，补充字符
German_PhoneBook_100_CI_AS_KS_WS_SC_UTF8	German-PhoneBook-100，不区分大小写，区分重音，区分假名，区分全半角，补充字符，UTF8
German_PhoneBook_100_CI_AS_SC	German-PhoneBook-100，不区分大小写，区分重音，不区分假名，不区分全半角，补充字符
German_PhoneBook_100_CI_AS_SC_UTF8	German-PhoneBook-100，不区分大小写，区分重音，不区分假名，不区分全半角，补充字符，UTF8
German_PhoneBook_100_CI_AS_WS	German-PhoneBook-100，不区分大小写，区分重音，不区分假名，区分全半角

排序规则	描述
German_PhoneBook_100_CI_AS_WS_SC	German-PhoneBook-100，不区分大小写，区分重音，不区分假名，区分全半角，补充字符
German_PhoneBook_100_CI_AS_WS_SC_UTF8	German-PhoneBook-100，不区分大小写，区分重音，不区分假名，区分全半角，补充字符，UTF8
German_PhoneBook_100_CS_AI	German-PhoneBook-100，区分大小写，不区分重音，不区分假名，不区分全半角
German_PhoneBook_100_CS_AI_KS	German-PhoneBook-100，区分大小写，不区分重音，区分假名，不区分全半角
German_PhoneBook_100_CS_AI_KS_SC	German-PhoneBook-100，区分大小写，不区分重音，区分假名，不区分全半角，补充字符
German_PhoneBook_100_CS_AI_KS_SC_UTF8	German-PhoneBook-100，区分大小写，不区分重音，区分假名，不区分全半角，补充字符，UTF8
German_PhoneBook_100_CS_AI_KS_WS	German-PhoneBook-100，区分大小写，不区分重音，区分假名，区分全半角
German_PhoneBook_100_CS_AI_KS_WS_SC	German-PhoneBook-100，区分大小写，不区分重音，区分假名，区分全半角，补充字符
German_PhoneBook_100_CS_AI_KS_WS_SC_UTF8	German-PhoneBook-100，区分大小写，不区分重音，区分假名，区分全半角，补充字符，UTF8
German_PhoneBook_100_CS_AI_SC	German-PhoneBook-100，区分大小写，不区分重音，不区分假名，不区分全半角，补充字符
German_PhoneBook_100_CS_AI_SC_UTF8	German-PhoneBook-100，区分大小写，不区分重音，不区分假名，不区分全半角，补充字符，UTF8

排序规则	描述
German_PhoneBook_100_CS_AI_WS	German-PhoneBook-100，区分大小写，不区分重音，不区分假名，区分全半角
German_PhoneBook_100_CS_AI_WS_SC	German-PhoneBook-100，区分大小写，不区分重音，不区分假名，区分全半角，补充字符
German_PhoneBook_100_CS_AI_WS_SC_UTF8	German-PhoneBook-100，区分大小写，不区分重音，不区分假名，区分全半角，补充字符，UTF8
German_PhoneBook_100_CS_AS	German-PhoneBook-100，区分大小写，区分重音，不区分假名，不区分全半角
German_PhoneBook_100_CS_AS_KS	German-PhoneBook-100，区分大小写，区分重音，区分假名，不区分全半角
German_PhoneBook_100_CS_AS_KS_SC	German-PhoneBook-100，区分大小写，区分重音，区分假名，不区分全半角，补充字符
German_PhoneBook_100_CS_AS_KS_SC_UTF8	German-PhoneBook-100，区分大小写，区分重音，区分假名，不区分全半角，补充字符，UTF8
German_PhoneBook_100_CS_AS_KS_WS	German-PhoneBook-100，区分大小写，区分重音，区分假名，区分全半角
German_PhoneBook_100_CS_AS_KS_WS_SC	German-PhoneBook-100，区分大小写，区分重音，区分假名，区分全半角，补充字符
German_PhoneBook_100_CS_AS_KS_WS_SC_UTF8	German-PhoneBook-100，区分大小写，区分重音，区分假名，区分全半角，补充字符，UTF8
German_PhoneBook_BIN	German-PhoneBook，二进制排序
German_PhoneBook_BIN2	German-PhoneBook，二进制代码点比较排序
German_PhoneBook_CI_AI	German-PhoneBook，不区分大小写，不区分重音，不区分假名，不区分全半角

排序规则	描述
German_PhoneBook_CI_AI_KS	German-PhoneBook，不区分大小写，不区分重音，区分假名，不区分全半角
German_PhoneBook_CI_AI_KS_WS	German-PhoneBook，不区分大小写，不区分重音，区分假名，区分全半角
German_PhoneBook_CI_AI_WS	German-PhoneBook，不区分大小写，不区分重音，不区分假名，区分全半角
German_PhoneBook_CI_AS	German-PhoneBook，不区分大小写，区分重音，不区分假名，不区分全半角
German_PhoneBook_CI_AS_KS	German-PhoneBook，不区分大小写，区分重音，区分假名，不区分全半角
German_PhoneBook_CI_AS_KS_WS	German-PhoneBook，不区分大小写，区分重音，区分假名，区分全半角
German_PhoneBook_CI_AS_WS	German-PhoneBook，不区分大小写，区分重音，不区分假名，区分全半角
German_PhoneBook_CS_AI	German-PhoneBook，区分大小写，不区分重音，不区分假名，不区分全半角
German_PhoneBook_CS_AI_KS	German-PhoneBook，区分大小写，不区分重音，区分假名，不区分全半角
German_PhoneBook_CS_AI_KS_WS	German-PhoneBook，区分大小写，不区分重音，区分假名，区分全半角
German_PhoneBook_CS_AI_WS	German-PhoneBook，区分大小写，不区分重音，不区分假名，区分全半角
German_PhoneBook_CS_AS	German-PhoneBook，区分大小写，区分重音，不区分假名，不区分全半角
German_PhoneBook_CS_AS_KS	German-PhoneBook，区分大小写，区分重音，区分假名，不区分全半角

排序规则	描述
German_PhoneBook_CS_AS_KS_WS	German-PhoneBook，区分大小写，区分重音，区分假名，区分全半角
German_PhoneBook_CS_AS_WS	German-PhoneBook，区分大小写，区分重音，不区分假名，区分全半角
Hebrew_BIN	Hebrew，二进制排序
Hebrew_CI_AS	Hebrew，不区分大小写，区分重音，不区分假名，不区分全半角
Japanese_90_BIN	Japanese-90，二进制排序
Japanese_90_BIN2	Japanese-90，二进制代码点比较排序
Japanese_90_CI_AI	Japanese-90，不区分大小写，不区分重音，不区分假名，不区分全半角
Japanese_90_CI_AI_KS	Japanese-90，不区分大小写，不区分重音，区分假名，不区分全半角
Japanese_90_CI_AI_KS_SC	Japanese-90，不区分大小写，不区分重音，区分假名，不区分全半角，补充字符
Japanese_90_CI_AI_KS_SC_UTF8	Japanese-90，不区分大小写，不区分重音，区分假名，不区分全半角，补充字符，UTF8
Japanese_90_CI_AI_KS_WS	Japanese-90，不区分大小写，不区分重音，区分假名，区分全半角
Japanese_90_CI_AI_KS_WS_SC	Japanese-90，不区分大小写，不区分重音，区分假名，区分全半角，补充字符
Japanese_90_CI_AI_KS_WS_SC_UTF8	Japanese-90，不区分大小写，不区分重音，区分假名，区分全半角，补充字符，UTF8
Japanese_90_CI_AI_SC	Japanese-90，不区分大小写，不区分重音，不区分假名，不区分全半角，补充字符

排序规则	描述
Japanese_90_CI_AI_SC_UTF8	Japanese-90，不区分大小写，不区分重音，不区分假名，不区分全半角，补充字符，UTF8
Japanese_90_CI_AI_WS	Japanese-90，不区分大小写，不区分重音，不区分假名，区分全半角
Japanese_90_CI_AI_WS_SC	Japanese-90，不区分大小写，不区分重音，不区分假名，区分全半角，补充字符
Japanese_90_CI_AI_WS_SC_UTF8	Japanese-90，不区分大小写，不区分重音，不区分假名，区分全半角，补充字符，UTF8
Japanese_90_CI_AS	Japanese-90，不区分大小写，区分重音，不区分假名，不区分全半角
Japanese_90_CI_AS_KS	Japanese-90，不区分大小写，区分重音，区分假名，不区分全半角
Japanese_90_CI_AS_KS_SC	Japanese-90，不区分大小写，区分重音，区分假名，不区分全半角，补充字符
Japanese_90_CI_AS_KS_SC_UTF8	Japanese-90，不区分大小写，区分重音，区分假名，不区分全半角，补充字符，UTF8
Japanese_90_CI_AS_KS_WS	Japanese-90，不区分大小写，区分重音，区分假名，区分全半角
Japanese_90_CI_AS_KS_WS_SC	Japanese-90，不区分大小写，区分重音，区分假名，区分全半角，补充字符
Japanese_90_CI_AS_KS_WS_SC_UTF8	Japanese-90，不区分大小写，区分重音，区分假名，区分全半角，补充字符，UTF8
Japanese_90_CI_AS_SC	Japanese-90，不区分大小写，区分重音，不区分假名，不区分全半角，补充字符
Japanese_90_CI_AS_SC_UTF8	Japanese-90，不区分大小写，区分重音，不区分假名，不区分全半角，补充字符，UTF8

排序规则	描述
Japanese_90_CI_AS_WS	Japanese-90，不区分大小写，区分重音，不区分假名，区分全半角
Japanese_90_CI_AS_WS_SC	Japanese-90，不区分大小写，区分重音，不区分假名，区分全半角，补充字符
Japanese_90_CI_AS_WS_SC_UTF8	Japanese-90，不区分大小写，区分重音，不区分假名，区分全半角，补充字符，UTF8
Japanese_90_CS_AI	Japanese-90，区分大小写，不区分重音，不区分假名，不区分全半角
Japanese_90_CS_AI_KS	Japanese-90，区分大小写，不区分重音，区分假名，不区分全半角
Japanese_90_CS_AI_KS_SC	Japanese-90，区分大小写，不区分重音，区分假名，不区分全半角，补充字符
Japanese_90_CS_AI_KS_SC_UTF8	Japanese-90，区分大小写，不区分重音，区分假名，不区分全半角，补充字符，UTF8
Japanese_90_CS_AI_KS_WS	Japanese-90，区分大小写，不区分重音，区分假名，区分全半角
Japanese_90_CS_AI_KS_WS_SC	Japanese-90，区分大小写，不区分重音，区分假名，区分全半角，补充字符
Japanese_90_CS_AI_KS_WS_SC_UTF8	Japanese-90，区分大小写，不区分重音，区分假名，区分全半角，补充字符，UTF8
Japanese_90_CS_AI_SC	Japanese-90，区分大小写，不区分重音，不区分假名，不区分全半角，补充字符
Japanese_90_CS_AI_SC_UTF8	Japanese-90，区分大小写，不区分重音，不区分假名，不区分全半角，补充字符，UTF8
Japanese_90_CS_AI_WS	Japanese-90，区分大小写，不区分重音，不区分假名，区分全半角

排序规则	描述
Japanese_90_CS_AI_WS_SC	Japanese-90，区分大小写，不区分重音，不区分假名，区分全半角，补充字符
Japanese_90_CS_AI_WS_SC_UTF8	Japanese-90，区分大小写，不区分重音，不区分假名，区分全半角，补充字符，UTF8
Japanese_90_CS_AS	Japanese-90，区分大小写，区分重音，不区分假名，不区分全半角
Japanese_90_CS_AS_KS	Japanese-90，区分大小写，区分重音，区分假名，不区分全半角
Japanese_90_CS_AS_KS_SC	Japanese-90，区分大小写，区分重音，区分假名，不区分全半角，补充字符
Japanese_90_CS_AS_KS_SC_UTF8	Japanese-90，区分大小写，区分重音，区分假名，不区分全半角，补充字符，UTF8
Japanese_90_CS_AS_KS_WS	Japanese-90，区分大小写，区分重音，区分假名，区分全半角
Japanese_90_CS_AS_KS_WS_SC	Japanese-90，区分大小写，区分重音，区分假名，区分全半角，补充字符
Japanese_90_CS_AS_KS_WS_SC_UTF8	Japanese-90，区分大小写，区分重音，区分假名，区分全半角，补充字符，UTF8
Japanese_90_CS_AS_SC	Japanese-90，区分大小写，区分重音，不区分假名，不区分全半角，补充字符
Japanese_90_CS_AS_SC_UTF8	Japanese-90，区分大小写，区分重音，不区分假名，不区分全半角，补充字符，UTF8
Japanese_90_CS_AS_WS	Japanese-90，区分大小写，区分重音，不区分假名，区分全半角
Japanese_90_CS_AS_WS_SC	Japanese-90，区分大小写，区分重音，不区分假名，区分全半角，补充字符

排序规则	描述
Japanese_90_CS_AS_WS_SC_UTF8	Japanese-90，区分大小写，区分重音，不区分假名，区分全半角，补充字符，UTF8
Japanese_BIN	Japanese，二进制排序
Japanese_BIN2	Japanese，二进制代码点比较排序
Japanese_Bushu_Kakusu_100_BIN	Japanese-Bushu-Kakusu-100，二进制排序
Japanese_Bushu_Kakusu_100_BIN2	Japanese-Bushu-Kakusu-100，二进制代码点比较排序
Japanese_Bushu_Kakusu_100_CI_AI	Japanese-Bushu-Kakusu-100，不区分大小写，不区分重音，不区分假名，不区分全半角
Japanese_Bushu_Kakusu_100_CI_AI_KS	Japanese-Bushu-Kakusu-100，不区分大小写，不区分重音，区分假名，不区分全半角
Japanese_Bushu_Kakusu_100_CI_AI_KS_SC	Japanese-Bushu-Kakusu-100，不区分大小写，不区分重音，区分假名，不区分全半角，补充字符
Japanese_Bushu_Kakusu_100_CI_AI_KS_SC_UTF8	Japanese-Bushu-Kakusu-100，不区分大小写，不区分重音，区分假名，不区分全半角，补充字符，UTF8
Japanese_Bushu_Kakusu_100_CI_AI_KS_WS	Japanese-Bushu-Kakusu-100，不区分大小写，不区分重音，区分假名，区分全半角
Japanese_Bushu_Kakusu_100_CI_AI_KS_WS_SC	Japanese-Bushu-Kakusu-100，不区分大小写，不区分重音，区分假名，区分全半角，补充字符
Japanese_Bushu_Kakusu_100_CI_AI_KS_WS_SC_UTF8	Japanese-Bushu-Kakusu-100，不区分大小写，不区分重音，区分假名，区分全半角，补充字符，UTF8

排序规则	描述
Japanese_Bushu_Kakusu_100_CI_AI_SC	Japanese-Bushu-Kakusu-100，不区分大小写，不区分重音，不区分假名，不区分全半角，补充字符
Japanese_Bushu_Kakusu_100_CI_AI_SC_UTF8	Japanese-Bushu-Kakusu-100，不区分大小写，不区分重音，不区分假名，不区分全半角，补充字符，UTF8
Japanese_Bushu_Kakusu_100_CI_AI_WS	Japanese-Bushu-Kakusu-100，不区分大小写，不区分重音，不区分假名，区分全半角
Japanese_Bushu_Kakusu_100_CI_AI_WS_SC	Japanese-Bushu-Kakusu-100，不区分大小写，不区分重音，不区分假名，区分全半角，补充字符
Japanese_Bushu_Kakusu_100_CI_AI_WS_SC_UTF8	Japanese-Bushu-Kakusu-100，不区分大小写，不区分重音，不区分假名，区分全半角，补充字符，UTF8
Japanese_Bushu_Kakusu_100_CI_AS	Japanese-Bushu-Kakusu-100，不区分大小写，区分重音，不区分假名，不区分全半角
Japanese_Bushu_Kakusu_100_CI_AS_KS	Japanese-Bushu-Kakusu-100，不区分大小写，区分重音，区分假名，不区分全半角
Japanese_Bushu_Kakusu_100_CI_AS_KS_SC	Japanese-Bushu-Kakusu-100，不区分大小写，区分重音，区分假名，不区分全半角，补充字符
Japanese_Bushu_Kakusu_100_CI_AS_KS_SC_UTF8	Japanese-Bushu-Kakusu-100，不区分大小写，区分重音，区分假名，不区分全半角，补充字符，UTF8
Japanese_Bushu_Kakusu_100_CI_AS_KS_WS	Japanese-Bushu-Kakusu-100，不区分大小写，区分重音，区分假名，区分全半角
Japanese_Bushu_Kakusu_100_CI_AS_KS_WS_SC	Japanese-Bushu-Kakusu-100，不区分大小写，区分重音，区分假名，区分全半角，补充字符

排序规则	描述
Japanese_Bushu_Kakusu_100_CI_AS_KS_WS_SC_UTF8	Japanese-Bushu-Kakusu-100，不区分大小写，区分重音，区分假名，区分全半角，补充字符，UTF8
Japanese_Bushu_Kakusu_100_CI_AS_SC	Japanese-Bushu-Kakusu-100，不区分大小写，区分重音，不区分假名，不区分全半角，补充字符
Japanese_Bushu_Kakusu_100_CI_AS_SC_UTF8	Japanese-Bushu-Kakusu-100，不区分大小写，区分重音，不区分假名，不区分全半角，补充字符，UTF8
Japanese_Bushu_Kakusu_100_CI_AS_WS	Japanese-Bushu-Kakusu-100，不区分大小写，区分重音，不区分假名，区分全半角
Japanese_Bushu_Kakusu_100_CI_AS_WS_SC	Japanese-Bushu-Kakusu-100，不区分大小写，区分重音，不区分假名，区分全半角，补充字符
Japanese_Bushu_Kakusu_100_CI_AS_WS_SC_UTF8	Japanese-Bushu-Kakusu-100，不区分大小写，区分重音，不区分假名，区分全半角，补充字符，UTF8
Japanese_Bushu_Kakusu_100_CS_AI	Japanese-Bushu-Kakusu-100，区分大小写，不区分重音，不区分假名，不区分全半角
Japanese_Bushu_Kakusu_100_CS_AI_KS	Japanese-Bushu-Kakusu-100，区分大小写，不区分重音，区分假名，不区分全半角
Japanese_Bushu_Kakusu_100_CS_AI_KS_SC	Japanese-Bushu-Kakusu-100，区分大小写，不区分重音，区分假名，不区分全半角，补充字符
Japanese_Bushu_Kakusu_100_CS_AI_KS_SC_UTF8	Japanese-Bushu-Kakusu-100，区分大小写，不区分重音，区分假名，不区分全半角，补充字符，UTF8
Japanese_Bushu_Kakusu_100_CS_AI_KS_WS	Japanese-Bushu-Kakusu-100，区分大小写，不区分重音，区分假名，区分全半角

排序规则	描述
Japanese_Bushu_Kakusu_100_CS_AI_KS_WS_SC	Japanese-Bushu-Kakusu-100，区分大小写，不区分重音，区分假名，区分全半角，补充字符
Japanese_Bushu_Kakusu_100_CS_AI_KS_WS_SC_UTF8	Japanese-Bushu-Kakusu-100，区分大小写，不区分重音，区分假名，区分全半角，补充字符，UTF8
Japanese_Bushu_Kakusu_100_CS_AI_SC	Japanese-Bushu-Kakusu-100，区分大小写，不区分重音，不区分假名，不区分全半角，补充字符
Japanese_Bushu_Kakusu_100_CS_AI_SC_UTF8	Japanese-Bushu-Kakusu-100，区分大小写，不区分重音，不区分假名，不区分全半角，补充字符，UTF8
Japanese_Bushu_Kakusu_100_CS_AI_WS	Japanese-Bushu-Kakusu-100，区分大小写，不区分重音，不区分假名，区分全半角
Japanese_Bushu_Kakusu_100_CS_AI_WS_SC	Japanese-Bushu-Kakusu-100，区分大小写，不区分重音，不区分假名，区分全半角，补充字符
Japanese_Bushu_Kakusu_100_CS_AI_WS_SC_UTF8	Japanese-Bushu-Kakusu-100，区分大小写，不区分重音，不区分假名，区分全半角，补充字符，UTF8
Japanese_Bushu_Kakusu_100_CS_AS	Japanese-Bushu-Kakusu-100，区分大小写，区分重音，不区分假名，不区分全半角
Japanese_Bushu_Kakusu_100_CS_AS_KS	Japanese-Bushu-Kakusu-100，区分大小写，区分重音，区分假名，不区分全半角
Japanese_Bushu_Kakusu_100_CS_AS_KS_SC	Japanese-Bushu-Kakusu-100，区分大小写，区分重音，区分假名，不区分全半角，补充字符
Japanese_Bushu_Kakusu_100_CS_AS_KS_SC_UTF8	Japanese-Bushu-Kakusu-100，区分大小写，区分重音，区分假名，不区分全半角，补充字符，UTF8

排序规则	描述
Japanese_Bushu_Kakusu_100_CS_AS_KS_WS	Japanese-Bushu-Kakusu-100，区分大小写，区分重音，区分假名，区分全半角
Japanese_Bushu_Kakusu_100_CS_AS_KS_WS_SC	Japanese-Bushu-Kakusu-100，区分大小写，区分重音，区分假名，区分全半角，补充字符
Japanese_Bushu_Kakusu_100_CS_AS_KS_WS_SC_UTF8	Japanese-Bushu-Kakusu-100，区分大小写，区分重音，区分假名，区分全半角，补充字符，UTF8
Japanese_Bushu_Kakusu_100_CS_AS_SC	Japanese-Bushu-Kakusu-100，区分大小写，区分重音，不区分假名，不区分全半角，补充字符
Japanese_Bushu_Kakusu_100_CS_AS_SC_UTF8	Japanese-Bushu-Kakusu-100，区分大小写，区分重音，不区分假名，不区分全半角，补充字符，UTF8
Japanese_Bushu_Kakusu_100_CS_AS_WS	Japanese-Bushu-Kakusu-100，区分大小写，区分重音，不区分假名，区分全半角
Japanese_Bushu_Kakusu_100_CS_AS_WS_SC	Japanese-Bushu-Kakusu-100，区分大小写，区分重音，不区分假名，区分全半角，补充字符
Japanese_Bushu_Kakusu_100_CS_AS_WS_SC_UTF8	Japanese-Bushu-Kakusu-100，区分大小写，区分重音，不区分假名，区分全半角，补充字符，UTF8
Japanese_Bushu_Kakusu_140_BIN	Japanese-Bushu-Kakusu-140，二进制排序
Japanese_Bushu_Kakusu_140_BIN2	Japanese-Bushu-Kakusu-140，二进制代码点比较排序
Japanese_Bushu_Kakusu_140_CI_AI	Japanese-Bushu-Kakusu-140，不区分大小写，不区分重音，不区分假名，不区分全半角，补充字符，不区分变体选择器

排序规则	描述
Japanese_Bushu_Kakusu_140_CI_AI_KS	Japanese-Bushu-Kakusu-140，不区分大小写，不区分重音，区分假名，不区分全半角，补充字符，不区分变体选择器
Japanese_Bushu_Kakusu_140_CI_AI_KS_UTF8	Japanese-Bushu-Kakusu-140，不区分大小写，不区分重音，区分假名，不区分全半角，补充字符，不区分变体选择器，UTF8
Japanese_Bushu_Kakusu_140_CI_AI_KS_VSS	Japanese-Bushu-Kakusu-140，不区分大小写，不区分重音，区分假名，不区分全半角，补充字符，区分变体选择器
Japanese_Bushu_Kakusu_140_CI_AI_KS_VSS_UTF8	Japanese-Bushu-Kakusu-140，不区分大小写，不区分重音，区分假名，不区分全半角，补充字符，区分变体选择器，UTF8
Japanese_Bushu_Kakusu_140_CI_AI_KS_WS	Japanese-Bushu-Kakusu-140，不区分大小写，不区分重音，区分假名，区分全半角，补充字符，不区分变体选择器
Japanese_Bushu_Kakusu_140_CI_AI_KS_WS_UTF8	Japanese-Bushu-Kakusu-140，不区分大小写，不区分重音，区分假名，区分全半角，补充字符，不区分变体选择器，UTF8
Japanese_Bushu_Kakusu_140_CI_AI_KS_WS_VSS	Japanese-Bushu-Kakusu-140，不区分大小写，不区分重音，区分假名，区分全半角，补充字符，区分变体选择器
Japanese_Bushu_Kakusu_140_CI_AI_KS_WS_VSS_UTF8	Japanese-Bushu-Kakusu-140，不区分大小写，不区分重音，区分假名，区分全半角，补充字符，区分变体选择器，UTF8
Japanese_Bushu_Kakusu_140_CI_AI_UTF8	Japanese-Bushu-Kakusu-140，不区分大小写，不区分重音，不区分假名，不区分全半角，补充字符，不区分变体选择器，UTF8

排序规则	描述
Japanese_Bushu_Kakusu_140_CI_AI_VSS	Japanese-Bushu-Kakusu-140，不区分大小写，不区分重音，不区分假名，不区分全半角，补充字符，区分变体选择器
Japanese_Bushu_Kakusu_140_CI_AI_VSS_UTF8	Japanese-Bushu-Kakusu-140，不区分大小写，不区分重音，不区分假名，不区分全半角，补充字符，区分变体选择器，UTF8
Japanese_Bushu_Kakusu_140_CI_AI_WS	Japanese-Bushu-Kakusu-140，不区分大小写，不区分重音，不区分假名，区分全半角，补充字符，不区分变体选择器
Japanese_Bushu_Kakusu_140_CI_AI_WS_UTF8	Japanese-Bushu-Kakusu-140，不区分大小写，不区分重音，不区分假名，区分全半角，补充字符，不区分变体选择器，UTF8
Japanese_Bushu_Kakusu_140_CI_AI_WS_VSS	Japanese-Bushu-Kakusu-140，不区分大小写，不区分重音，不区分假名，区分全半角，补充字符，区分变体选择器
Japanese_Bushu_Kakusu_140_CI_AI_WS_VSS_UTF8	Japanese-Bushu-Kakusu-140，不区分大小写，不区分重音，不区分假名，区分全半角，补充字符，区分变体选择器，UTF8
Japanese_Bushu_Kakusu_140_CI_AS	Japanese-Bushu-Kakusu-140，不区分大小写，区分重音，不区分假名，不区分全半角，补充字符，不区分变体选择器
Japanese_Bushu_Kakusu_140_CI_AS_KS	Japanese-Bushu-Kakusu-140，不区分大小写，区分重音，区分假名，不区分全半角，补充字符，不区分变体选择器
Japanese_Bushu_Kakusu_140_CI_AS_KS_UTF8	Japanese-Bushu-Kakusu-140，不区分大小写，区分重音，区分假名，不区分全半角，补充字符，不区分变体选择器，UTF8

排序规则	描述
Japanese_Bushu_Kakusu_140_CI_AS_KS_VSS	Japanese-Bushu-Kakusu-140，不区分大小写，区分重音，区分假名，不区分全半角，补充字符，区分变体选择器
Japanese_Bushu_Kakusu_140_CI_AS_KS_VSS_UTF8	Japanese-Bushu-Kakusu-140，不区分大小写，区分重音，区分假名，不区分全半角，补充字符，区分变体选择器，UTF8
Japanese_Bushu_Kakusu_140_CI_AS_KS_WS	Japanese-Bushu-Kakusu-140，不区分大小写，区分重音，区分假名，区分全半角，补充字符，不区分变体选择器
Japanese_Bushu_Kakusu_140_CI_AS_KS_WS_UTF8	Japanese-Bushu-Kakusu-140，不区分大小写，区分重音，区分假名，区分全半角，补充字符，不区分变体选择器，UTF8
Japanese_Bushu_Kakusu_140_CI_AS_KS_WS_VSS	Japanese-Bushu-Kakusu-140，不区分大小写，区分重音，区分假名，区分全半角，补充字符，区分变体选择器
Japanese_Bushu_Kakusu_140_CI_AS_KS_WS_VSS_UTF8	Japanese-Bushu-Kakusu-140，不区分大小写，区分重音，区分假名，区分全半角，补充字符，区分变体选择器，UTF8
Japanese_Bushu_Kakusu_140_CI_AS_UTF8	Japanese-Bushu-Kakusu-140，不区分大小写，区分重音，不区分假名，不区分全半角，补充字符，不区分变体选择器，UTF8
Japanese_Bushu_Kakusu_140_CI_AS_VSS	Japanese-Bushu-Kakusu-140，不区分大小写，区分重音，不区分假名，不区分全半角，补充字符，区分变体选择器
Japanese_Bushu_Kakusu_140_CI_AS_VSS_UTF8	Japanese-Bushu-Kakusu-140，不区分大小写，区分重音，不区分假名，不区分全半角，补充字符，区分变体选择器，UTF8

排序规则	描述
Japanese_Bushu_Kakusu_140_CI_AS_WS	Japanese-Bushu-Kakusu-140，不区分大小写，区分重音，不区分假名，区分全半角，补充字符，不区分变体选择器
Japanese_Bushu_Kakusu_140_CI_AS_WS_UTF8	Japanese-Bushu-Kakusu-140，不区分大小写，区分重音，不区分假名，区分全半角，补充字符，不区分变体选择器，UTF8
Japanese_Bushu_Kakusu_140_CI_AS_WS_VSS	Japanese-Bushu-Kakusu-140，不区分大小写，区分重音，不区分假名，区分全半角，补充字符，区分变体选择器
Japanese_Bushu_Kakusu_140_CI_AS_WS_VSS_UTF8	Japanese-Bushu-Kakusu-140，不区分大小写，区分重音，不区分假名，区分全半角，补充字符，区分变体选择器，UTF8
Japanese_Bushu_Kakusu_140_CS_AI	Japanese-Bushu-Kakusu-140，区分大小写，不区分重音，不区分假名，不区分全半角，补充字符，不区分变体选择器
Japanese_Bushu_Kakusu_140_CS_AI_KS	Japanese-Bushu-Kakusu-140，区分大小写，不区分重音，区分假名，不区分全半角，补充字符，不区分变体选择器
Japanese_Bushu_Kakusu_140_CS_AI_KS_UTF8	Japanese-Bushu-Kakusu-140，区分大小写，不区分重音，区分假名，不区分全半角，补充字符，不区分变体选择器，UTF8
Japanese_Bushu_Kakusu_140_CS_AI_KS_VSS	Japanese-Bushu-Kakusu-140，区分大小写，不区分重音，区分假名，不区分全半角，补充字符，区分变体选择器
Japanese_Bushu_Kakusu_140_CS_AI_KS_VSS_UTF8	Japanese-Bushu-Kakusu-140，区分大小写，不区分重音，区分假名，不区分全半角，补充字符，区分变体选择器，UTF8

排序规则	描述
Japanese_Bushu_Kakusu_140_CS_AI_KS_WS	Japanese-Bushu-Kakusu-140，区分大小写，不区分重音，区分假名，区分全半角，补充字符，不区分变体选择器
Japanese_Bushu_Kakusu_140_CS_AI_KS_WS_UTF8	Japanese-Bushu-Kakusu-140，区分大小写，不区分重音，区分假名，区分全半角，补充字符，不区分变体选择器，UTF8
Japanese_Bushu_Kakusu_140_CS_AI_KS_WS_VSS	Japanese-Bushu-Kakusu-140，区分大小写，不区分重音，区分假名，区分全半角，补充字符，区分变体选择器
Japanese_Bushu_Kakusu_140_CS_AI_KS_WS_VSS_UTF8	Japanese-Bushu-Kaku su-140，区分大小写，不区分重音，区分假名，区分全半角，补充字符，区分变体选择器，UTF8
Japanese_Bushu_Kakusu_140_CS_AI_UTF8	Japanese-Bushu-Kakusu-140，区分大小写，不区分重音，不区分假名，不区分全半角，补充字符，不区分变体选择器，UTF8
Japanese_Bushu_Kakusu_140_CS_AI_VSS	Japanese-Bushu-Kakusu-140，区分大小写，不区分重音，不区分假名，不区分全半角，补充字符，区分变体选择器
Japanese_Bushu_Kakusu_140_CS_AI_VSS_UTF8	Japanese-Bushu-Kakusu-140，区分大小写，不区分重音，不区分假名，不区分全半角，补充字符，区分变体选择器，UTF8
Japanese_Bushu_Kakusu_140_CS_AI_WS	Japanese-Bushu-Kakusu-140，区分大小写，不区分重音，不区分假名，区分全半角，补充字符，不区分变体选择器
Japanese_Bushu_Kakusu_140_CS_AI_WS_UTF8	Japanese-Bushu-Kakusu-140，区分大小写，不区分重音，不区分假名，区分全半角，补充字符，不区分变体选择器，UTF8

排序规则	描述
Japanese_Bushu_Kakusu_140_CS_AI_WS_VSS	Japanese-Bushu-Kakusu-140，区分大小写，不区分重音，不区分假名，区分全半角，补充字符，区分变体选择器
Japanese_Bushu_Kakusu_140_CS_AI_WS_VSS_UTF8	Japanese-Bushu-Kakusu-140，区分大小写，不区分重音，不区分假名，区分全半角，补充字符，区分变体选择器，UTF8
Japanese_Bushu_Kakusu_140_CS_AS	Japanese-Bushu-Kakusu-140，区分大小写，区分重音，不区分假名，不区分全半角，补充字符，不区分变体选择器
Japanese_Bushu_Kakusu_140_CS_AS_KS	Japanese-Bushu-Kakusu-140，区分大小写，区分重音，区分假名，不区分全半角，补充字符，不区分变体选择器
Japanese_Bushu_Kakusu_140_CS_AS_KS_UTF8	Japanese-Bushu-Kakusu-140，区分大小写，区分重音，区分假名，不区分全半角，补充字符，不区分变体选择器，UTF8
Japanese_Bushu_Kakusu_140_CS_AS_KS_VSS	Japanese-Bushu-Kakusu-140，区分大小写，区分重音，区分假名，不区分全半角，补充字符，区分变体选择器
Japanese_Bushu_Kakusu_140_CS_AS_KS_VSS_UTF8	Japanese-Bushu-Kakusu-140，区分大小写，区分重音，区分假名，不区分全半角，补充字符，区分变体选择器，UTF8
Japanese_Bushu_Kakusu_140_CS_AS_KS_WS	Japanese-Bushu-Kakusu-140，区分大小写，区分重音，区分假名，区分全半角，补充字符，不区分变体选择器
Japanese_Bushu_Kakusu_140_CS_AS_KS_WS_UTF8	Japanese-Bushu-Kakusu-140，区分大小写，区分重音，区分假名，区分全半角，补充字符，不区分变体选择器，UTF8

排序规则	描述
Japanese_Bushu_Kakusu_140_CS_AS_KS_WS_VSS	Japanese-Bushu-Kakusu-140，区分大小写，区分重音，区分假名，区分全半角，补充字符，区分变体选择器
Japanese_Bushu_Kakusu_140_CS_AS_KS_WS_VSS_UTF8	Japanese-Bushu-Kakusu-140，区分大小写，区分重音，区分假名，区分全半角，补充字符，区分变体选择器，UTF8
Japanese_Bushu_Kakusu_140_CS_AS_UTF8	Japanese-Bushu-Kakusu-140，区分大小写，区分重音，不区分假名，不区分全半角，补充字符，不区分变体选择器，UTF8
Japanese_Bushu_Kakusu_140_CS_AS_VSS	Japanese-Bushu-Kakusu-140，区分大小写，区分重音，不区分假名，不区分全半角，补充字符，区分变体选择器
Japanese_Bushu_Kakusu_140_CS_AS_VSS_UTF8	Japanese-Bushu-Kakusu-140，区分大小写，区分重音，不区分假名，不区分全半角，补充字符，区分变体选择器，UTF8
Japanese_Bushu_Kakusu_140_CS_AS_WS	Japanese-Bushu-Kakusu-140，区分大小写，区分重音，不区分假名，区分全半角，补充字符，不区分变体选择器
Japanese_Bushu_Kakusu_140_CS_AS_WS_UTF8	Japanese-Bushu-Kakusu-140，区分大小写，区分重音，不区分假名，区分全半角，补充字符，不区分变体选择器，UTF8
Japanese_Bushu_Kakusu_140_CS_AS_WS_VSS	Japanese-Bushu-Kakusu-140，区分大小写，区分重音，不区分假名，区分全半角，补充字符，区分变体选择器
Japanese_Bushu_Kakusu_140_CS_AS_WS_VSS_UTF8	Japanese-Bushu-Kakusu-140，区分大小写，区分重音，不区分假名，区分全半角，补充字符，区分变体选择器，UTF8

排序规则	描述
Japanese_CI_AI	Japanese，不区分大小写，不区分重音，不区分假名，不区分全半角
Japanese_CI_AI_KS	Japanese，不区分大小写，不区分重音，区分假名，不区分全半角
Japanese_CI_AI_KS_WS	Japanese，不区分大小写，不区分重音，区分假名，区分全半角
Japanese_CI_AI_WS	Japanese，不区分大小写，不区分重音，不区分假名，区分全半角
Japanese_CI_AS	Japanese，不区分大小写，区分重音，不区分假名，不区分全半角
Japanese_CI_AS_KS	Japanese，不区分大小写，区分重音，区分假名，不区分全半角
Japanese_CI_AS_KS_WS	Japanese，不区分大小写，区分重音，区分假名，区分全半角
Japanese_CI_AS_WS	Japanese，不区分大小写，区分重音，不区分假名，区分全半角
Japanese_CS_AI	Japanese，区分大小写，不区分重音，不区分假名，不区分全半角
Japanese_CS_AI_KS	Japanese，区分大小写，不区分重音，区分假名，不区分全半角
Japanese_CS_AI_KS_WS	Japanese，区分大小写，不区分重音，区分假名，区分全半角
Japanese_CS_AI_WS	Japanese，区分大小写，不区分重音，不区分假名，区分全半角
Japanese_CS_AS	Japanese，区分大小写，区分重音，不区分假名，不区分全半角

排序规则	描述
Japanese_CS_AS_KS	Japanese，区分大小写，区分重音，区分假名，不区分全半角
Japanese_CS_AS_KS_WS	Japanese，区分大小写，区分重音，区分假名，区分全半角
Japanese_CS_AS_WS	Japanese，区分大小写，区分重音，不区分假名，区分全半角
Japanese_Unicode_BIN	Japanese-Unicode，二进制排序
Japanese_Unicode_BIN2	Japanese-Unicode，二进制代码点比较排序
Japanese_Unicode_CI_AI	Japanese-Unicode，不区分大小写，不区分重音，不区分假名，不区分全半角
Japanese_Unicode_CI_AI_KS	Japanese-Unicode，不区分大小写，不区分重音，区分假名，不区分全半角
Japanese_Unicode_CI_AI_KS_WS	Japanese-Unicode，不区分大小写，不区分重音，区分假名，区分全半角
Japanese_Unicode_CI_AI_WS	Japanese-Unicode，不区分大小写，不区分重音，不区分假名，区分全半角
Japanese_Unicode_CI_AS	Japanese-Unicode，不区分大小写，区分重音，不区分假名，不区分全半角
Japanese_Unicode_CI_AS_KS	Japanese-Unicode，不区分大小写，区分重音，区分假名，不区分全半角
Japanese_Unicode_CI_AS_KS_WS	Japanese-Unicode，不区分大小写，区分重音，区分假名，区分全半角
Japanese_Unicode_CI_AS_WS	Japanese-Unicode，不区分大小写，区分重音，不区分假名，区分全半角

排序规则	描述
Japanese_Unicode_CS_AI	Japanese-Unicode，区分大小写，不区分重音，不区分假名，不区分全半角
Japanese_Unicode_CS_AI_KS	Japanese-Unicode，区分大小写，不区分重音，区分假名，不区分全半角
Japanese_Unicode_CS_AI_KS_WS	Japanese-Unicode，区分大小写，不区分重音，区分假名，区分全半角
Japanese_Unicode_CS_AI_WS	Japanese-Unicode，区分大小写，不区分重音，不区分假名，区分全半角
Japanese_Unicode_CS_AS	Japanese-Unicode，区分大小写，区分重音，不区分假名，不区分全半角
Japanese_Unicode_CS_AS_KS	Japanese-Unicode，区分大小写，区分重音，区分假名，不区分全半角
Japanese_Unicode_CS_AS_KS_WS	Japanese-Unicode，区分大小写，区分重音，区分假名，区分全半角
Japanese_Unicode_CS_AS_WS	Japanese-Unicode，区分大小写，区分重音，不区分假名，区分全半角
Japanese_XJIS_100_BIN	Japanese-XJIS-100，二进制排序
Japanese_XJIS_100_BIN2	Japanese-XJIS-100，二进制代码点比较排序
Japanese_XJIS_100_CI_AI	Japanese-XJIS-100，不区分大小写，不区分重音，不区分假名，不区分全半角
Japanese_XJIS_100_CI_AI_KS	Japanese-XJIS-100，不区分大小写，不区分重音，区分假名，不区分全半角
Japanese_XJIS_100_CI_AI_KS_SC	Japanese-XJIS-100，不区分大小写，不区分重音，区分假名，不区分全半角，补充字符

排序规则	描述
Japanese_XJIS_100_CI_AI_KS_SC_UTF8	Japanese-XJIS-100，不区分大小写，不区分重音，区分假名，不区分全半角，补充字符，UTF8
Japanese_XJIS_100_CI_AI_KS_WS	Japanese-XJIS-100，不区分大小写，不区分重音，区分假名，区分全半角
Japanese_XJIS_100_CI_AI_KS_WS_SC	Japanese-XJIS-100，不区分大小写，不区分重音，区分假名，区分全半角，补充字符
Japanese_XJIS_100_CI_AI_KS_WS_SC_UTF8	Japanese-XJIS-100，不区分大小写，不区分重音，区分假名，区分全半角，补充字符，UTF8
Japanese_XJIS_100_CI_AI_SC	Japanese-XJIS-100，不区分大小写，不区分重音，不区分假名，不区分全半角，补充字符
Japanese_XJIS_100_CI_AI_SC_UTF8	Japanese-XJIS-100，不区分大小写，不区分重音，不区分假名，不区分全半角，补充字符，UTF8
Japanese_XJIS_100_CI_AI_WS	Japanese-XJIS-100，不区分大小写，不区分重音，不区分假名，区分全半角
Japanese_XJIS_100_CI_AI_WS_SC	Japanese-XJIS-100，不区分大小写，不区分重音，不区分假名，区分全半角，补充字符
Japanese_XJIS_100_CI_AI_WS_SC_UTF8	Japanese-XJIS-100，不区分大小写，不区分重音，不区分假名，区分全半角，补充字符，UTF8
Japanese_XJIS_100_CI_AS	Japanese-XJIS-100，不区分大小写，区分重音，不区分假名，不区分全半角
Japanese_XJIS_100_CI_AS_KS	Japanese-XJIS-100，不区分大小写，区分重音，区分假名，不区分全半角
Japanese_XJIS_100_CI_AS_KS_SC	Japanese-XJIS-100，不区分大小写，区分重音，区分假名，不区分全半角，补充字符

排序规则	描述
Japanese_XJIS_100_CI_AS_KS_SC_UTF8	Japanese-XJIS-100，不区分大小写，区分重音，区分假名，不区分全半角，补充字符，UTF8
Japanese_XJIS_100_CI_AS_KS_WS	Japanese-XJIS-100，不区分大小写，区分重音，区分假名，区分全半角
Japanese_XJIS_100_CI_AS_KS_WS_SC	Japanese-XJIS-100，不区分大小写，区分重音，区分假名，区分全半角，补充字符
Japanese_XJIS_100_CI_AS_KS_WS_SC_UTF8	Japanese-XJIS-100，不区分大小写，区分重音，区分假名，区分全半角，补充字符，UTF8
Japanese_XJIS_100_CI_AS_SC	Japanese-XJIS-100，不区分大小写，区分重音，不区分假名，不区分全半角，补充字符
Japanese_XJIS_100_CI_AS_SC_UTF8	Japanese-XJIS-100，不区分大小写，区分重音，不区分假名，不区分全半角，补充字符，UTF8
Japanese_XJIS_100_CI_AS_WS	Japanese-XJIS-100，不区分大小写，区分重音，不区分假名，区分全半角
Japanese_XJIS_100_CI_AS_WS_SC	Japanese-XJIS-100，不区分大小写，区分重音，不区分假名，区分全半角，补充字符
Japanese_XJIS_100_CI_AS_WS_SC_UTF8	Japanese-XJIS-100，不区分大小写，区分重音，不区分假名，区分全半角，补充字符，UTF8
Japanese_XJIS_100_CS_AI	Japanese-XJIS-100，区分大小写，不区分重音，不区分假名，不区分全半角
Japanese_XJIS_100_CS_AI_KS	Japanese-XJIS-100，区分大小写，不区分重音，区分假名，不区分全半角
Japanese_XJIS_100_CS_AI_KS_SC	Japanese-XJIS-100，区分大小写，不区分重音，区分假名，不区分全半角，补充字符

排序规则	描述
Japanese_XJIS_100_CS_AI_KS_SC_UTF8	Japanese-XJIS-100，区分大小写，不区分重音，区分假名，不区分全半角，补充字符，UTF8
Japanese_XJIS_100_CS_AI_KS_WS	Japanese-XJIS-100，区分大小写，不区分重音，区分假名，区分全半角
Japanese_XJIS_100_CS_AI_KS_WS_SC	Japanese-XJIS-100，区分大小写，不区分重音，区分假名，区分全半角，补充字符
Japanese_XJIS_100_CS_AI_KS_WS_SC_UTF8	Japanese-XJIS-100，区分大小写，不区分重音，区分假名，区分全半角，补充字符，UTF8
Japanese_XJIS_100_CS_AI_SC	Japanese-XJIS-100，区分大小写，不区分重音，不区分假名，不区分全半角，补充字符
Japanese_XJIS_100_CS_AI_SC_UTF8	Japanese-XJIS-100，区分大小写，不区分重音，不区分假名，不区分全半角，补充字符，UTF8
Japanese_XJIS_100_CS_AI_WS	Japanese-XJIS-100，区分大小写，不区分重音，不区分假名，区分全半角
Japanese_XJIS_100_CS_AI_WS_SC	Japanese-XJIS-100，区分大小写，不区分重音，不区分假名，区分全半角，补充字符
Japanese_XJIS_100_CS_AI_WS_SC_UTF8	Japanese-XJIS-100，区分大小写，不区分重音，不区分假名，区分全半角，补充字符，UTF8
Japanese_XJIS_100_CS_AS	Japanese-XJIS-100，区分大小写，区分重音，不区分假名，不区分全半角
Japanese_XJIS_100_CS_AS_KS	Japanese-XJIS-100，区分大小写，区分重音，区分假名，不区分全半角
Japanese_XJIS_100_CS_AS_KS_SC	Japanese-XJIS-100，区分大小写，区分重音，区分假名，不区分全半角，补充字符

排序规则	描述
Japanese_XJIS_100_CS_AS_KS_SC_UTF8	Japanese-XJIS-100，区分大小写，区分重音，区分假名，不区分全半角，补充字符，UTF8
Japanese_XJIS_100_CS_AS_KS_WS	Japanese-XJIS-100，区分大小写，区分重音，区分假名，区分全半角
Japanese_XJIS_100_CS_AS_KS_WS_SC	Japanese-XJIS-100，区分大小写，区分重音，区分假名，区分全半角，补充字符
Japanese_XJIS_100_CS_AS_KS_WS_SC_UTF8	Japanese-XJIS-100，区分大小写，区分重音，区分假名，区分全半角，补充字符，UTF8
Japanese_XJIS_100_CS_AS_SC	Japanese-XJIS-100，区分大小写，区分重音，不区分假名，不区分全半角，补充字符
Japanese_XJIS_100_CS_AS_SC_UTF8	Japanese-XJIS-100，区分大小写，区分重音，不区分假名，不区分全半角，补充字符，UTF8
Japanese_XJIS_100_CS_AS_WS	Japanese-XJIS-100，区分大小写，区分重音，不区分假名，区分全半角
Japanese_XJIS_140_BIN	Japanese-XJIS-140，二进制排序
Japanese_XJIS_140_BIN2	Japanese-XJIS-140，二进制代码点比较排序
Japanese_XJIS_140_CI_AI	Japanese-XJIS-140，不区分大小写，不区分重音，不区分假名，不区分全半角，补充字符，不区分变体选择器
Japanese_XJIS_140_CI_AI_KS	Japanese-XJIS-140，不区分大小写，不区分重音，区分假名，不区分全半角，补充字符，不区分变体选择器
Japanese_XJIS_140_CI_AI_KS_UTF8	Japanese-XJIS-140，不区分大小写，不区分重音，区分假名，不区分全半角，补充字符，不区分变体选择器，UTF8

排序规则	描述
Japanese_XJIS_140_CI_AI_KS_VSS	Japanese-XJIS-140，不区分大小写，不区分重音，区分假名，不区分全半角，补充字符，区分变体选择器
Japanese_XJIS_140_CI_AI_KS_VSS_UTF8	Japanese-XJIS-140，不区分大小写，不区分重音，区分假名，不区分全半角，补充字符，区分变体选择器，UTF8
Japanese_XJIS_140_CI_AI_KS_WS	Japanese-XJIS-140，不区分大小写，不区分重音，区分假名，区分全半角，补充字符，不区分变体选择器
Japanese_XJIS_140_CI_AI_KS_WS_UTF8	Japanese-XJIS-140，不区分大小写，不区分重音，区分假名，区分全半角，补充字符，不区分变体选择器，UTF8
Japanese_XJIS_140_CI_AI_KS_WS_VSS	Japanese-XJIS-140，不区分大小写，不区分重音，区分假名，区分全半角，补充字符，区分变体选择器
Japanese_XJIS_140_CI_AI_KS_WS_VSS_UTF8	Japanese-XJIS-140，不区分大小写，不区分重音，区分假名，区分全半角，补充字符，区分变体选择器，UTF8
Japanese_XJIS_140_CI_AI_UTF8	Japanese-XJIS-140，不区分大小写，不区分重音，不区分假名，不区分全半角，补充字符，不区分变体选择器，UTF8
Japanese_XJIS_140_CI_AI_VSS	Japanese-XJIS-140，不区分大小写，不区分重音，不区分假名，不区分全半角，补充字符，区分变体选择器
Japanese_XJIS_140_CI_AI_VSS_UTF8	Japanese-XJIS-140，不区分大小写，不区分重音，不区分假名，不区分全半角，补充字符，区分变体选择器，UTF8

排序规则	描述
Japanese_XJIS_140_CI_AI_WS	Japanese-XJIS-140，不区分大小写，不区分重音，不区分假名，区分全半角，补充字符，不区分变体选择器
Japanese_XJIS_140_CI_AI_WS_UTF8	Japanese-XJIS-140，不区分大小写，不区分重音，不区分假名，区分全半角，补充字符，不区分变体选择器，UTF8
Japanese_XJIS_140_CI_AI_WS_VSS	Japanese-XJIS-140，不区分大小写，不区分重音，不区分假名，区分全半角，补充字符，区分变体选择器
Japanese_XJIS_140_CI_AI_WS_VSS_UTF8	Japanese-XJIS-140，不区分大小写，不区分重音，不区分假名，区分全半角，补充字符，区分变体选择器，UTF8
Japanese_XJIS_140_CI_AS	Japanese-XJIS-140，不区分大小写，区分重音，不区分假名类型，不区分全半角，补充字符，不区分变体选择器
Japanese_XJIS_140_CI_AS_KS	Japanese-XJIS-140，不区分大小写，区分重音，区分假名，不区分全半角，补充字符，不区分变体选择器
Japanese_XJIS_140_CI_AS_KS_UTF8	Japanese-XJIS-140，不区分大小写，区分重音，区分假名，不区分全半角，补充字符，不区分变体选择器，UTF8
Japanese_XJIS_140_CI_AS_KS_VSS	Japanese-XJIS-140，不区分大小写，区分重音，区分假名类型，不区分全半角，补充字符，区分变体选择器
Japanese_XJIS_140_CI_AS_KS_VSS_UTF8	Japanese-XJIS-140，不区分大小写，区分重音，区分假名，不区分全半角，补充字符，区分变体选择器，UTF8

排序规则	描述
Japanese_XJIS_140_CI_AS_KS_WS	Japanese-XJIS-140，不区分大小写，区分重音，区分假名，区分全半角，补充字符，不区分变体选择器
Japanese_XJIS_140_CI_AS_KS_WS_UTF8	Japanese-XJIS-140，不区分大小写，区分重音，区分假名，区分全半角，补充字符，不区分变体选择器，UTF8
Japanese_XJIS_140_CI_AS_KS_WS_VSS	Japanese-XJIS-140，不区分大小写，区分重音，区分假名，区分全半角，补充字符，区分变体选择器
Japanese_XJIS_140_CI_AS_KS_WS_VSS_UTF8	Japanese-XJIS-140，不区分大小写，区分重音，区分假名，区分全半角，补充字符，区分变体选择器，UTF8
Japanese_XJIS_140_CI_AS_UTF8	Japanese-XJIS-140，不区分大小写，区分重音，不区分假名，不区分全半角，补充字符，不区分变体选择器，UTF8
Japanese_XJIS_140_CI_AS_VSS	Japanese-XJIS-140，不区分大小写，区分重音，不区分假名类型，不区分全半角，补充字符，区分变体选择器
Japanese_XJIS_140_CI_AS_VSS_UTF8	Japanese-XJIS-140，不区分大小写，区分重音，不区分假名，不区分全半角，补充字符，区分变体选择器，UTF8
Japanese_XJIS_140_CI_AS_WS	Japanese-XJIS-140，不区分大小写，区分重音，不区分假名，区分全半角，补充字符，不区分变体选择器
Japanese_XJIS_140_CI_AS_WS_UTF8	Japanese-XJIS-140，不区分大小写，区分重音，不区分假名，区分全半角，补充字符，不区分变体选择器，UTF8

排序规则	描述
Japanese_XJIS_140_CI_AS_WS_VSS	Japanese-XJIS-140，不区分大小写，区分重音，不区分假名，区分全半角，补充字符，区分变体选择器
Japanese_XJIS_140_CI_AS_WS_VSS_UTF8	Japanese-XJIS-140，不区分大小写，区分重音，不区分假名，区分全半角，补充字符，区分变体选择器，UTF8
Japanese_XJIS_140_CS_AI	Japanese-XJIS-140，区分大小写，不区分重音，不区分假名，不区分全半角，补充字符，不区分变体选择器
Japanese_XJIS_140_CS_AI_KS	Japanese-XJIS-140，区分大小写，不区分重音，区分假名，不区分全半角，补充字符，不区分变体选择器
Japanese_XJIS_140_CS_AI_KS_UTF8	Japanese-XJIS-140，区分大小写，不区分重音，区分假名，不区分全半角，补充字符，不区分变体选择器，UTF8
Japanese_XJIS_140_CS_AI_KS_VSS	Japanese-XJIS-140，区分大小写，不区分重音，区分假名，不区分全半角，补充字符，区分变体选择器
Japanese_XJIS_140_CS_AI_KS_VSS_UTF8	Japanese-XJIS-140，区分大小写，不区分重音，区分假名，不区分全半角，补充字符，区分变体选择器，UTF8
Japanese_XJIS_140_CS_AI_KS_WS	Japanese-XJIS-140，区分大小写，不区分重音，区分假名，区分全半角，补充字符，不区分变体选择器
Japanese_XJIS_140_CS_AI_KS_WS_UTF8	Japanese-XJIS-140，区分大小写，不区分重音，区分假名，区分全半角，补充字符，不区分变体选择器，UTF8

排序规则	描述
Japanese_XJIS_140_CS_AI_KS_WS_VSS	Japanese-XJIS-140，区分大小写，不区分重音，区分假名，区分全半角，补充字符，区分变体选择器
Japanese_XJIS_140_CS_AI_KS_WS_VSS_UTF8	Japanese-XJIS-140，区分大小写，不区分重音，区分假名，区分全半角，补充字符，区分变体选择器，UTF8
Japanese_XJIS_140_CS_AI_UTF8	Japanese-XJIS-140，区分大小写，不区分重音，不区分假名，不区分全半角，补充字符，不区分变体选择器，UTF8
Japanese_XJIS_140_CS_AI_VSS	Japanese-XJIS-140，区分大小写，不区分重音，不区分假名，不区分全半角，补充字符，区分变体选择器
Japanese_XJIS_140_CS_AI_VSS_UTF8	Japanese-XJIS-140，区分大小写，不区分重音，不区分假名，不区分全半角，补充字符，区分变体选择器，UTF8
Japanese_XJIS_140_CS_AI_WS	Japanese-XJIS-140，区分大小写，不区分重音，不区分假名，区分全半角，补充字符，不区分变体选择器
Japanese_XJIS_140_CS_AI_WS_UTF8	Japanese-XJIS-140，区分大小写，不区分重音，不区分假名，区分全半角，补充字符，不区分变体选择器，UTF8
Japanese_XJIS_140_CS_AI_WS_VSS	Japanese-XJIS-140，区分大小写，不区分重音，不区分假名，区分全半角，补充字符，区分变体选择器
Japanese_XJIS_140_CS_AI_WS_VSS_UTF8	Japanese-XJIS-140，区分大小写，不区分重音，不区分假名，区分全半角，补充字符，区分变体选择器，UTF8

排序规则	描述
Japanese_XJIS_140_CS_AS	Japanese-XJIS-140，区分大小写，区分重音，不区分假名，不区分全半角，补充字符，不区分变体选择器
Japanese_XJIS_140_CS_AS_KS	Japanese-XJIS-140，区分大小写，区分重音，区分假名，不区分全半角，补充字符，不区分变体选择器
Japanese_XJIS_140_CS_AS_KS_UTF8	Japanese-XJIS-140，区分大小写，区分重音，区分假名，不区分全半角，补充字符，不区分变体选择器，UTF8
Japanese_XJIS_140_CS_AS_KS_VSS	Japanese-XJIS-140，区分大小写，区分重音，区分假名，不区分全半角，补充字符，区分变体选择器
Japanese_XJIS_140_CS_AS_KS_VSS_UTF8	Japanese-XJIS-140，区分大小写，区分重音，区分假名，不区分全半角，补充字符，区分变体选择器，UTF8
Japanese_XJIS_140_CS_AS_KS_WS	Japanese-XJIS-140，区分大小写，区分重音，区分假名，区分全半角，补充字符，不区分变体选择器
Japanese_XJIS_140_CS_AS_KS_WS_UTF8	Japanese-XJIS-140，区分大小写，区分重音，区分假名，区分全半角，补充字符，不区分变体选择器，UTF8
Japanese_XJIS_140_CS_AS_KS_WS_VSS	Japanese-XJIS-140，区分大小写，区分重音，区分假名，区分全半角，补充字符，区分变体选择器
Japanese_XJIS_140_CS_AS_KS_WS_VSS_UTF8	Japanese-XJIS-140，区分大小写，区分重音，区分假名，区分全半角，补充字符，区分变体选择器，UTF8

排序规则	描述
Japanese_XJIS_140_CS_AS_UTF8	Japanese-XJIS-140，区分大小写，区分重音，不区分假名，不区分全半角，补充字符，不区分变体选择器，UTF8
Japanese_XJIS_140_CS_AS_VSS	Japanese-XJIS-140，区分大小写，区分重音，不区分假名类型，不区分全半角，补充字符，区分变体选择器
Japanese_XJIS_140_CS_AS_VSS_UTF8	Japanese-XJIS-140，区分大小写，区分重音，不区分假名，不区分全半角，补充字符，区分变体选择器，UTF8
Japanese_XJIS_140_CS_AS_WS	Japanese-XJIS-140，区分大小写，区分重音，不区分假名，区分全半角，补充字符，不区分变体选择器
Japanese_XJIS_140_CS_AS_WS_UTF8	Japanese-XJIS-140，区分大小写，区分重音，不区分假名，区分全半角，补充字符，不区分变体选择器，UTF8
Japanese_XJIS_140_CS_AS_WS_VSS	Japanese-XJIS-140，区分大小写，区分重音，不区分假名，区分全半角，补充字符，区分变体选择器
Japanese_XJIS_140_CS_AS_WS_VSS_UTF8	Japanese-XJIS-140，区分大小写，区分重音，不区分假名，区分全半角，补充字符，区分变体选择器，UTF8
Korean_Wansung_CI_AS	Korean_Wansung，不区分大小写，区分重音，不区分假名，不区分全半角
Latin1_General_100_BIN	Latin1-General-100，二进制排序
Latin1_General_100_BIN2	Latin1-General-100，二进制代码点比较排序
Latin1_General_100_BIN2_UTF8	Latin1-General-100，二进制代码点比较排序，UTF8

排序规则	描述
Latin1_General_100_CI_AS	Latin1-General-100，不区分大小写，区分重音，不区分假名，不区分全半角
Latin1_General_100_CI_AS_SC_UTF8	Latin1-General-100，不区分大小写，区分重音，不区分假名，不区分全半角，补充字符，UTF8
Latin1_General_BIN	Latin1-General，二进制排序
Latin1_General_BIN2	Latin1-General，二进制代码点比较排序
Latin1_General_CI_AI	Latin1-General，不区分大小写，不区分重音，不区分假名，不区分全半角
Latin1_General_CI_AS	Latin1-General，不区分大小写，区分重音，不区分假名，不区分全半角
Latin1_General_CI_AS_KS	Latin1-General，不区分大小写，区分重音，区分假名，不区分全半角
Latin1_General_CS_AS	Latin1-General，区分大小写，区分重音，不区分假名，不区分全半角
Modern_Spanish_CI_AS	Modern_Spanish，不区分大小写，区分重音，不区分假名，不区分全半角
SQL_1xCompat_CP850_CI_AS	对于 Unicode 数据为 Latin1-General，不区分大小写，区分重音，不区分假名，不区分全半角，对于非 Unicode 数据，为代码页 850 上的 SQL Server 排序顺序 49
SQL_Latin1_General_CP1_CI_AI	对于 Unicode 数据为 Latin1-General，不区分大小写，不区分重音，不区分假名，不区分全半角；对于非 Unicode 数据，为代码页 1252 上的 SQL Server 排序顺序 54

排序规则	描述
SQL_Latin1_General_CP1_CI_AS	对于 Unicode 数据为 Latin1-General，不区分大小写，区分重音，不区分假名，不区分全半角；对于非 Unicode 数据，为代码页 1252 上的 SQL Server 排序顺序 52
SQL_Latin1_General_CP1_CS_AS	对于 Unicode 数据为 Latin1-General，区分大小写，区分重音，不区分假名，不区分全半角；对于非 Unicode 数据，为代码页 1252 上的 SQL Server 排序顺序 51
SQL_Latin1_General_CP1250_CI_AS	对于 Unicode 数据为 Latin1-General，不区分大小写，区分重音，不区分假名，不区分全半角；对于非 Unicode 数据，为代码页 1250 上的 SQL Server 排序顺序 82
SQL_Latin1_General_CP1250_CS_AS	对于 Unicode 数据为 Latin1-General，区分大小写，区分重音，不区分假名，不区分全半角；对于非 Unicode 数据，为代码页 1250 上的 SQL Server 排序顺序 81
SQL_Latin1_General_CP1251_CI_AS	对于 Unicode 数据为 Latin1-General，不区分大小写，区分重音，不区分假名，不区分全半角；对于非 Unicode 数据，为代码页 1251 上的 SQL Server 排序顺序 106
SQL_Latin1_General_CP1251_CS_AS	对于 Unicode 数据为 Latin1-General，区分大小写，区分重音，不区分假名，不区分全半角；对于非 Unicode 数据，为代码页 1251 上的 SQL Server 排序顺序 105
SQL_Latin1_General_CP1253_CI_AI	对于 Unicode 数据为 Latin1-General，不区分大小写，不区分重音，不区分假名，不区分全半角；对于非 Unicode 数据，为代码页 1253 上的 SQL Server 排序顺序 124

排序规则	描述
SQL_Latin1_General_CP1253_CI_AS	对于 Unicode 数据为 Latin1-General，不区分大小写，区分重音，不区分假名，不区分全半角；对于非 Unicode 数据，为代码页 1253 上的 SQL Server 排序顺序 114
SQL_Latin1_General_CP1253_CS_AS	对于 Unicode 数据为 Latin1-General，区分大小写，区分重音，不区分假名，不区分全半角；对于非 Unicode 数据，为代码页 1253 上的 SQL Server 排序顺序 113
SQL_Latin1_General_CP1254_CI_AS	对于 Unicode 数据为 Turkish，不区分大小写，区分重音，不区分假名，不区分全半角，对于非 Unicode 数据，为代码页 1254 上的 SQL Server 排序顺序 130
SQL_Latin1_General_CP1254_CS_AS	对于 Unicode 数据为 Turkish，区分大小写，区分重音，不区分假名，不区分全半角，对于非 Unicode 数据，为代码页 1254 上的 SQL Server 排序顺序 129
SQL_Latin1_General_CP1255_CI_AS	对于 Unicode 数据为 Latin1-General，不区分大小写，区分重音，不区分假名，不区分全半角；对于非 Unicode 数据，为代码页 1255 上的 SQL Server 排序顺序 138
SQL_Latin1_General_CP1255_CS_AS	对于 Unicode 数据为 Latin1-General，区分大小写，区分重音，不区分假名，不区分全半角；对于非 Unicode 数据，为代码页 1255 上的 SQL Server 排序顺序 137
SQL_Latin1_General_CP1256_CI_AS	对于 Unicode 数据为 Latin1-General，不区分大小写，区分重音，不区分假名，不区分全半角；对于非 Unicode 数据，为代码页 1256 上的 SQL Server 排序顺序 146

排序规则	描述
SQL_Latin1_General_CP1256_CS_AS	对于 Unicode 数据为 Latin1-General，区分大小写，区分重音，不区分假名，不区分全半角；对于非 Unicode 数据，为代码页 1256 上的 SQL Server 排序顺序 145
SQL_Latin1_General_CP1257_CI_AS	对于 Unicode 数据为 Latin1-General，不区分大小写，区分重音，不区分假名，不区分全半角；对于非 Unicode 数据，为代码页 1257 上的 SQL Server 排序顺序 154
SQL_Latin1_General_CP1257_CS_AS	对于 Unicode 数据为 Latin1-General，区分大小写，区分重音，不区分假名，不区分全半角；对于非 Unicode 数据，为代码页 1257 上的 SQL Server 排序顺序 153
SQL_Latin1_General_CP437_BIN	对于 Unicode 数据为 Latin1-General，二进制排序，对于非 Unicode 数据为代码页 437 上的 SQL Server 排序顺序 30
SQL_Latin1_General_CP437_BIN2	对于 Unicode 数据为 Latin1-General，二进制代码点比较排序，对于非 Unicode 数据为代码页 437 上的 SQL Server 排序顺序 30
SQL_Latin1_General_CP437_CI_AI	对于 Unicode 数据为 Latin1-General，不区分大小写，不区分重音，不区分假名，不区分全半角；对于非 Unicode 数据，为代码页 437 上的 SQL Server 排序顺序 34
SQL_Latin1_General_CP437_CI_AS	对于 Unicode 数据为 Latin1-General，不区分大小写，区分重音，不区分假名，不区分全半角，对于非 Unicode 数据，为代码页 437 上的 SQL Server 排序顺序 32

排序规则	描述
SQL_Latin1_General_CP437_CS_AS	对于 Unicode 数据为 Latin1-General，区分大小写，区分重音，不区分假名，不区分全半角；对于非 Unicode 数据，为代码页 437 上的 SQL Server 排序顺序 31
SQL_Latin1_General_CP850_BIN	对于 Unicode 数据为 Latin1-General，二进制排序，对于非 Unicode 数据为代码页 850 上的 SQL Server 排序顺序 40
SQL_Latin1_General_CP850_BIN2	对于 Unicode 数据为 Latin1-General，二进制代码点比较排序，对于非 Unicode 数据为代码页 850 上的 SQL Server 排序顺序 40
SQL_Latin1_General_CP850_CI_AI	对于 Unicode 数据为 Latin1-General，不区分大小写，不区分重音，不区分假名，不区分全半角；对于非 Unicode 数据，为代码页 850 上的 SQL Server 排序顺序 44
SQL_Latin1_General_CP850_CI_AS	对于 Unicode 数据为 Latin1-General，不区分大小写，区分重音，不区分假名，不区分全半角，对于非 Unicode 数据，为代码页 850 上的 SQL Server 排序顺序 42
SQL_Latin1_General_CP850_CS_AS	对于 Unicode 数据为 Latin1-General，区分大小写，区分重音，不区分假名，不区分全半角；对于非 Unicode 数据，为代码页 850 上的 SQL Server 排序顺序 41
SQL_Latin1_General_Pref_CP1_CI_AS	对于 Unicode 数据为 Latin1-General，不区分大小写，区分重音，不区分假名，不区分全半角；对于非 Unicode 数据，为代码页 1252 上的 SQL Server 排序顺序 53

排序规则	描述
SQL_Latin1_General_Pref_CP437_CI_AS	对于 Unicode 数据为 Latin1-General，不区分大小写，区分重音，不区分假名，不区分全半角，对于非 Unicode 数据，为代码页 437 上的 SQL Server 排序顺序 33
SQL_Latin1_General_Pref_CP850_CI_AS	对于 Unicode 数据为 Latin1-General，不区分大小写，区分重音，不区分假名，不区分全半角，对于非 Unicode 数据，为代码页 850 上的 SQL Server 排序顺序 43
Thai_CI_AS	Thai，不区分大小写，区分重音，不区分假名，不区分全半角

RDS Custom for SQL Server 数据库实例的本地时区

原定设置情况下，已设置 RDS Custom for SQL Server 数据库实例的时区。当前默认值为协调世界时 (UTC)。您可改为将数据库实例的时区设置为本地时区以匹配应用程序的时区。

首次创建数据库实例时将设置时区。您可以使用 Amazon Web Services Management Console、Amazon RDS API CreateDBInstance 操作或 Amazon CLI create-db-instance 命令创建数据库实例。

如果数据库实例是多可用区部署的一部分，则当您进行故障转移时，您的时区将保留您设置的本地时区。

在请求时间点还原时，请指定要还原到的时间。时间以您的当地时区显示。有关更多信息，请参阅“将 Amazon RDS 的数据库实例还原到指定时间”。

下面是在数据库实例上设置本地时区时的限制：

- 您可以在实例创建期间为数据库实例配置时区，但不能修改现有 RDS Custom for SQL Server 数据库实例的时区。
- 如果修改了现有 RDS Custom for SQL Server 数据库实例的时区，则 RDS Custom 会将数据库实例状态更改为 `unsupported-configuration`，并发送事件通知。
- 您无法将快照从一个时区的数据库实例还原至另一个时区的数据库实例。

- 强烈建议您不要将备份文件从一个时区还原到另一个时区。如果您将备份文件从一个时区还原至为另一个时区，则必须审查您的查询和应用程序来查看时区更改的效果。有关更多信息，请参阅"使用本机备份和还原导入和导出 SQL Server 数据库"。

支持的时区

您可以将本地时区设置为下表中列出的值之一。

时区	标准时间偏差	描述	备注
阿富汗标准时间	(UTC+04:30)	喀布尔	此时区不遵守夏令时。
阿拉斯加标准时间	(UTC–09:00)	阿拉斯加州	
阿留申群岛标准时间	(UTC–10:00)	阿留申群岛	
阿尔泰标准时间	(UTC+07:00)	巴尔瑙尔，戈尔诺-阿尔泰斯克	
阿拉伯标准时间	(UTC+03:00)	科威特，利雅得	此时区不遵守夏令时。
阿拉伯标准时间	(UTC+04:00)	阿布扎比、马斯喀特	
阿拉伯标准时间	(UTC+03:00)	巴格达	此时区不遵守夏令时。
阿根廷标准时间	(UTC–03:00)	布宜诺斯艾利斯市	此时区不遵守夏令时。
阿斯特拉罕标准时间	(UTC+04:00)	阿斯特拉罕，乌里扬诺夫斯克	
大西洋标准时间	(UTC–04:00)	大西洋时间 (加拿大)	
澳大利亚中部标准时间	(UTC+09:30)	达尔文	此时区不遵守夏令时。
澳大利亚中西部标准时间	(UTC+08:45)	尤克拉	

时区	标准时间偏差	描述	备注
澳大利亚东部标准时间	(UTC+10:00)	堪培拉、墨尔本、悉尼	
阿塞拜疆标准时间	(UTC+04:00)	巴库	
亚速尔群岛标准时间	(UTC–01:00)	亚速尔群岛	
巴伊亚标准时间	(UTC–03:00)	萨尔瓦多	
孟加拉国标准时间	(UTC+06:00)	达卡	此时区不遵守夏令时。
白俄罗斯标准时间	(UTC+03:00)	明斯克	此时区不遵守夏令时。
布干维尔岛标准时间	(UTC+11:00)	布干维尔岛	
加拿大中部标准时间	(UTC–06:00)	萨斯喀彻温省	此时区不遵守夏令时。
佛得角标准时间	(UTC–01:00)	佛得角群岛	此时区不遵守夏令时。
高加索标准时间	(UTC+04:00)	埃里温	
澳大 利亚中部标准时间	(UTC+09:30)	阿德莱德	
中美洲标准时间	(UTC–06:00)	中美洲	此时区不遵守夏令时。
中亚标准时间	(UTC+06:00)	阿斯塔纳	此时区不遵守夏令时。
巴西中部标准时间	(UTC–04:00)	库亚巴	
中欧标准时间	(UTC+01:00)	贝尔格莱德、布拉迪斯拉发、布达佩斯、卢布尔雅那、布拉格	

时区	标准时间偏差	描述	备注
中欧标准时间	(UTC+01:00)	萨拉热窝、斯科普里、华沙、萨格勒布	
太平洋中部标准时间	(UTC+11:00)	索罗门群岛、新喀里多尼亚	此时区不遵守夏令时。
中部标准时间	(UTC–06:00)	中部时间 (美国和加拿大)	
中部标准时间（墨西哥）	(UTC–06:00)	瓜达拉哈拉、墨西哥城、蒙特雷	
查塔姆群岛标准时间	(UTC+12:45)	查塔姆群岛	
中国标准时间	(UTC+08:00)	北京、重庆、香港特别行政区、乌鲁木齐	此时区不遵守夏令时。
古巴标准时间	(UTC–05:00)	哈瓦那	
国际日期变更线标准时间	(UTC–12:00)	国际日期变更线西	此时区不遵守夏令时。
东非标准时间	(UTC+03:00)	内罗毕	此时区不遵守夏令时。
澳大利亚东部标准时间	(UTC+10:00)	布里斯班	此时区不遵守夏令时。
东欧标准时间	(UTC+02:00)	基希讷乌	
东部南美洲标准时间	(UTC–03:00)	巴西利亚	
复活节岛标准时间	(UTC–06:00)	复活节岛	
东部标准时间	(UTC–05:00)	东部时间 (美国和加拿大)	
东部标准时间（墨西哥）	(UTC–05:00)	切图马尔	

时区	标准时间偏差	描述	备注
埃及标准时间	(UTC+02:00)	开罗	
叶卡捷琳堡标准时间	(UTC+05:00)	叶卡捷琳堡	
斐济标准时间	(UTC+12:00)	斐济	
FLE 标准时间	(UTC+02:00)	赫尔辛基，基辅，里加，索非亚，塔林，维尔纽斯	
格鲁吉亚标准时间	(UTC+04:00)	第比利斯	此时区不遵守夏令时。
GMT 标准时间	(UTC)	都柏林、爱丁堡、里斯本、伦敦	此时区与格林威治标准时间不同。此时区遵守夏令时。
格陵兰标准时间	(UTC–03:00)	格陵兰	
格林威治标准时间	(UTC)	蒙罗维亚、雷克雅未克	此时区不遵守夏令时。
GTB 标准时间	(UTC+02:00)	雅典、布加勒斯特	
海地标准时间	(UTC–05:00)	海地	
夏威夷标准时间	(UTC–10:00)	夏威夷州	
印度标准时间	(UTC+05:30)	钦奈、加尔各答、孟买、新德里	此时区不遵守夏令时。
伊朗标准时间	(UTC+03:30)	德黑兰	
以色列标准时间	(UTC+02:00)	耶路撒冷	
约旦标准时间	(UTC+02:00)	阿曼	
加里宁格勒标准时间	(UTC+02:00)	加里宁格勒	

时区	标准时间偏差	描述	备注
堪察加标准时间	(UTC+12:00)	彼得罗巴甫洛夫斯克-堪察加 – 旧	
韩国标准时间	(UTC+09:00)	首尔	此时区不遵守夏令时。
利比亚标准时间	(UTC+02:00)	的黎波里	
莱恩群岛标准时间	(UTC+14:00)	圣诞岛	
豪勋爵岛标准时间	(UTC+10:30)	豪勋爵岛	
马加丹标准时间	(UTC+11:00)	马加丹	此时区不遵守夏令时。
麦哲伦标准时间	(UTC–03:00)	蓬塔阿雷纳斯	
马克萨斯群岛标准时间	(UTC–09:30)	马克萨斯群岛	
毛里求斯标准时间	(UTC+04:00)	路易港	此时区不遵守夏令时。
中东标准时间	(UTC+02:00)	贝鲁特	
蒙得维的亚标准时间	(UTC–03:00)	蒙得维的亚	
摩洛哥标准时间	(UTC+01:00)	卡萨布兰卡	
山地标准时间	(UTC–07:00)	山区时间 (美国和加拿大)	
山地标准时间（墨西哥）	(UTC–07:00)	奇瓦瓦、拉巴斯、马萨特兰	
缅甸标准时间	(UTC+06:30)	仰光	此时区不遵守夏令时。
中亚北部标准时间	(UTC+07:00)	新西伯利亚	

时区	标准时间偏差	描述	备注
纳米比亚标准时间	(UTC+02:00)	温得和克	
尼泊尔标准时间	(UTC+05:45)	加德满都	此时区不遵守夏令时。
新西兰标准时间	(UTC+12:00)	奥克兰、惠灵顿	
纽芬兰标准时间	(UTC–03:30)	纽芬兰省	
诺福克岛标准时间	(UTC+11:00)	诺福克岛	
北亚东部标准时间	(UTC+08:00)	伊尔库茨克	
北亚标准时间	(UTC+07:00)	克拉斯诺亚尔斯克	
朝鲜标准时间	(UTC+09:00)	平壤	
鄂木斯克标准时间	(UTC+06:00)	鄂木斯克	
太平洋 SA 标准时间	(UTC–03:00)	圣地亚哥	
太平洋标准时间	(UTC–08:00)	太平洋时间 (美国和加拿大)	
太平洋标准时间（墨西哥）	(UTC–08:00)	下加利福尼亚州	
巴基斯坦标准时间	(UTC+05:00)	伊斯兰堡，卡拉奇	此时区不遵守夏令时。
巴拉圭标准时间	(UTC–04:00)	亚松森	
罗马标准时间	(UTC+01:00)	布鲁塞尔，哥本哈根，马德里，巴黎	
俄罗斯时区 10	(UTC+11:00)	乔库尔达赫	
俄罗斯时区 11	(UTC+12:00)	阿纳德尔，堪察加彼得罗巴甫洛夫斯克	

时区	标准时间偏差	描述	备注
俄罗斯时区 3	(UTC+04:00)	伊热夫斯克，萨马拉	
俄罗斯标准时间	(UTC+03:00)	莫斯科、圣彼德堡、伏尔加格勒	此时区不遵守夏令时。
南美洲东部标准时间	(UTC–03:00)	卡宴，福塔雷萨	此时区不遵守夏令时。
SA 太平洋标准时间	(UTC–05:00)	波哥大、利马、基多、里约布兰科	此时区不遵守夏令时。
南美洲西部标准时间	(UTC–04:00)	乔治敦，拉巴斯，马瑙斯，圣胡安	此时区不遵守夏令时。
圣皮埃尔标准时间	(UTC–03:00)	圣皮埃尔和密克隆群岛	
萨哈林标准时间	(UTC+11:00)	萨哈林	
萨摩亚群岛标准时间	(UTC+13:00)	萨摩亚群岛	
圣多美标准时间	(UTC+01:00)	圣多美	
萨拉托夫标准时间	(UTC+04:00)	萨拉托夫	
东南亚标准时间	(UTC+07:00)	曼谷、河内、雅加达	此时区不遵守夏令时。
新加坡标准时间	(UTC+08:00)	吉隆坡、新加坡	此时区不遵守夏令时。
南非标准时间	(UTC+02:00)	哈拉雷，比勒陀利亚	此时区不遵守夏令时。
斯里兰卡标准时间	(UTC+05:30)	斯里哈亚华登尼普拉	此时区不遵守夏令时。
苏丹标准时间	(UTC+02:00)	喀土穆	

时区	标准时间偏差	描述	备注
叙利亚标准时间	(UTC+02:00)	大马士革	
台北标准时间	(UTC+08:00)	台北	此时区不遵守夏令时。
塔斯马尼亚岛标准时间	(UTC+10:00)	霍巴特	
托坎廷斯标准时间	(UTC–03:00)	阿拉瓜伊纳	
东京标准时间	(UTC+09:00)	大坂、札幌、东京	此时区不遵守夏令时。
托木斯克标准时间	(UTC+07:00)	托木斯克	
汤加标准时间	(UTC+13:00)	努库阿洛法	此时区不遵守夏令时。
外贝加尔标准时间	(UTC+09:00)	赤塔市	
土耳其标准时间	(UTC+03:00)	伊斯坦布尔	
特克斯和凯科斯群岛标准时间	(UTC–05:00)	特克斯和凯科斯群岛	
乌兰巴托标准时间	(UTC+08:00)	乌兰巴托	此时区不遵守夏令时。
美国东部标准时间	(UTC–05:00)	印第安纳 (东部)	
美国山地标准时间	(UTC–07:00)	亚利桑那州	此时区不遵守夏令时。
UTC	UTC	协调世界时	此时区不遵守夏令时。
UTC–02	(UTC–02:00)	协调世界时–02	此时区不遵守夏令时。
UTC–08	(UTC–08:00)	协调世界时–08	

时区	标准时间偏差	描述	备注
UTC–09	(UTC–09:00)	协调世界时–09	
UTC–11	(UTC–11:00)	协调世界时–11	此时区不遵守夏令时。
UTC+12	(UTC+12:00)	协调世界时+12	此时区不遵守夏令时。
UTC+13	(UTC+13:00)	协调世界时+13	
委内瑞拉标准时间	(UTC–04:00)	加拉加斯	此时区不遵守夏令时。
符拉迪沃斯托克标准时间	(UTC+10:00)	符拉迪沃斯托克	
伏尔加格勒标准时间	(UTC+04:00)	伏尔加格勒	
澳大利亚西部标准时间	(UTC+08:00)	珀斯	此时区不遵守夏令时。
中非西部标准时间	(UTC+01:00)	中非西部	此时区不遵守夏令时。
西欧标准时间	(UTC+01:00)	阿姆斯特丹、柏林、伯尔尼、罗马、斯德哥尔摩、维也纳	
西蒙古标准时间	(UTC+07:00)	科布多	
西亚标准时间	(UTC+05:00)	阿什哈巴德，塔什干	此时区不遵守夏令时。
西岸标准时间	(UTC+02:00)	加沙，希伯伦	
西太平洋标准时间	(UTC+10:00)	关岛，莫尔兹比港	此时区不遵守夏令时。
雅库茨克标准时间	(UTC+09:00)	雅库茨克	

在 RDS Custom for SQL Server 中使用服务主密钥

RDS Custom for SQL Server 支持使用服务主密钥（SMK）。RDS Custom 在 RDS Custom for SQL Server 数据库实例的整个生命周期中保留相同的 SMK。通过保留相同的 SMK，您的数据库实例可以使用通过 SMK 加密的对象，例如链接服务器密码和凭证。如果您使用多可用区部署，RDS Custom 还会在主和辅助数据库实例之间同步和维护 SMK。

主题

区域和版本可用性

对于 RDS Custom 上提供的所有 SQL Server 版本，在所有提供 RDS Custom for SQL Server 的区域中都支持使用 SMK。有关适用于 RDS Custom for SQL Server 的 Amazon RDS 的版本和区域可用性的更多信息，请参阅支持 RDS Custom for SQL Server 的区域和数据库引擎。

支持的特征

将 SMK 与 RDS Custom for SQL Server 一起使用时，支持以下功能：

- 透明数据加密 (TDE)
- 列级加密
- 数据库邮件
- 链接服务器
- SQL Server Integration Services (SSIS)

使用 TDE

SMK 支持配置透明数据加密（TDE），这种加密在数据写入存储之前对数据进行加密，并在从存储读取数据时自动解密数据。与 RDS for SQL Server 不同，在 RDS Custom for SQL Server 数据库实例上配置 TDE 不需要使用选项组。相反，创建证书和数据库加密密钥后，可以运行以下命令在数据库级别开启 TDE：

```
ALTER DATABASE [myDatabase] SET ENCRYPTION ON;
```

有关将 TDE 与 RDS for SQL Server 一起使用的更多信息，请参阅SQL Server 中的透明数据加密支持。

有关 Microsoft SQL Server 中的 TDE 的详细信息，请参阅 Microsoft 文档中的透明数据加密。

配置功能

有关配置将 SMK 和 RDS Custom for SQL Server 结合使用的功能的详细步骤，您可以参阅 Amazon RDS 数据库博客中的以下博客：

- 链接服务器：Configuring linked servers on RDS Custom for SQL Server。
- SSIS：Migrate SSIS packages to RDS Custom for SQL Server。
- TDE：Secure your data using TDE on RDS Custom for SQL Server。

要求和限制

在 RDS Custom for SQL Server 数据库实例中使用 SMK 时，请记住以下要求和限制：

- 如果您在数据库实例上重新生成 SMK，则应立即执行手动数据库快照。我们建议尽可能避免重新生成 SMK。
- 必须保留服务器证书和数据库主密钥密码的备份。如果不维护这些内容的备份，可能会导致数据丢失。
- 如果您配置 SSIS，则应使用 SSM 文档将 RDS Custom for SQL Server 数据库实例加入到域中，以防扩展计算或主机更换。
- 启用 TDE 或列加密后，数据库备份会自动加密。当您执行快照还原或时间点恢复时，将还原源数据库实例的 SMK 以解密进行还原的数据，并生成一个新的 SMK 来重新加密已还原实例上的数据。

有关 Microsoft SQL Server 中服务主密钥的更多信息，请参阅 Microsoft 文档中的 SQL Server 和数据库加密密钥。

为 Amazon RDS Custom for SQL Server 设置环境

在为 Amazon RDS Custom for SQL Server 数据库实例创建和管理数据库实例之前，请确保执行以下任务。

目录

> **Note**
>
> 有关如何设置先决条件和启动 Amazon RDS Custom for SQL Server 的分步教程，请参阅通过 CloudFormation 模板开始使用 Amazon RDS Custom for SQL Server（网络设置）和了解创建 Amazon RDS Custom for SQL Server 实例需满足的先决条件。

设置 RDS Custom for SQL Server 的先决条件

在创建 RDS Custom for SQL Server 数据库实例之前，请确保您的环境符合本主题中描述的要求。也可以使用 CloudFormation 模板在您的Amazon Web Services 账户中设置先决条件。有关更多信息，请参阅使用 Amazon CloudFormation 进行配置

RDS Custom for SQL Server 要求您配置以下先决条件：

- 配置创建实例所需的 Amazon Identity and Access Management（IAM）权限。这是向 RDS 发出 `create-db-instance` 请求所需的 Amazon Identity and Access Management（IAM）用户或角色。
- 配置 RDS Custom for SQL Server 数据库实例所需的必备资源：
  - 配置加密 RDS Custom 实例所需的 Amazon KMS 密钥。RDS Custom 在创建实例时需要使用客户托管密钥进行加密。KMS 密钥 ARN、ID、别名 ARN 或别名在创建 RDS Custom 数据库实例的请求中作为 `kms-key-id` 参数传递。
  - 在 RDS Custom for SQL Server 数据库实例中配置所需的权限。RDS Custom 在创建实例时会将实例配置文件附加到数据库实例，并使用它在数据库实例中进行自动化。在 RDS Custom 创建请求中，实例配置文件名称设置为 `custom-iam-instance-profile`。可以从 Amazon Web Services Management Console创建实例配置文件或手动创建实例配置文件。有关更多信息，请参阅使用 Amazon Web Services Management Console自动创建实例配置文件和手动创建您的 IAM 角色和实例配置文件。
  - 根据对 RDS Custom for SQL Server 的要求配置网络设置。RDS Custom 实例位于您在创建实例时提供的子网中（配置有数据库子网组）。这些子网必须允许 RDS Custom 实例与实现 RDS 自动化所需的服务进行通信。

> **Note**
>
> 要满足上述要求，请确保没有任何限制账户级别权限的服务控制策略（SCP）。
> 如果您使用的账户属于 Amazon 组织，则该账户可能具有限制账户级别权限的服务控制策略（SCP）。确保 SCP 不会限制对于您使用以下过程创建的用户和角色的权限。

有关 SCP 的更多信息，请参阅《Amazon Organizations 用户指南》中的服务控制策略 (SCP)。使用 Amazon CLI 命令 describe-organization 检查您的账户是否属于 Amazon 组织。有关 Amazon Organizations 的更多信息，请参阅《Amazon Organizations 用户指南》中的什么是 Amazon Organizations。

有关 RDS Custom for SQL Server 的一般要求，请参阅 RDS Custom for SQL Server 的一般要求。

使用 Amazon Web Services Management Console自动创建实例配置文件

RDS Custom 要求您创建和配置实例配置文件以启动任何 RDS Custom for SQL Server 数据库实例。使用 Amazon Web Services Management Console只需一个步骤即可创建和附加新的实例配置文件。此选项位于创建数据库、还原快照和还原到时间点控制台页面的"RDS Custom 安全性"部分。选择创建新的实例配置文件以提供实例配置文件名称后缀。Amazon Web Services Management Console将创建新的实例配置文件，该配置文件具有执行 RDS Custom 自动化任务所需的权限。要自动创建新的实例配置文件，您的登录 Amazon Web Services Management Console用户必须具有 `iam:CreateInstanceProfile`、`iam:AddRoleToInstanceProfile`、`iam:CreateRole` 和 `iam:AttachRolePolicy` 权限。

> **Note**
>
> 此选项只在 Amazon Web Services Management Console中可用。如果您使用的是 CLI 或 SDK，请使用 RDS Custom 提供的 CloudFormation 模板或手动创建实例配置文件。有关更多信息，请参阅手动创建您的 IAM 角色和实例配置文件。

步骤 1：为您的 IAM 主体授予所需的权限

确保您有足够的访问权限来创建 RDS Custom 实例。可以使用控制台或 CLI 创建 RDS Custom for SQL Server 数据库实例的 IAM 用户或 IAM 角色（称为 IAM 主体）必须实施以下任一策略，才能成功创建数据库实例：

- `AdministratorAccess` 策略
- 具有以下其他权限的 `AmazonRDSFullAccess` 策略：

```
iam:SimulatePrincipalPolicy
cloudtrail:CreateTrail
cloudtrail:StartLogging
s3:CreateBucket
```

```
s3:PutBucketPolicy
s3:PutBucketObjectLockConfiguration
s3:PutBucketVersioning
kms:CreateGrant
kms:DescribeKey
kms:Decrypt
kms:ReEncryptFrom
kms:ReEncryptTo
kms:GenerateDataKeyWithoutPlaintext
kms:GenerateDataKey
ec2:DescribeImages
ec2:RunInstances
ec2:CreateTags
```

RDS Custom 在创建实例期间使用这些权限。这些权限在您的账户中配置执行 RDS Custom 操作所需的资源。

有关 `kms:CreateGrant` 权限的更多信息，请参阅 [Amazon KMS key 管理]。

以下示例 JSON 策略授予所需的权限。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ValidateIamRole",
            "Effect": "Allow",
            "Action": "iam:SimulatePrincipalPolicy",
            "Resource": "*"
        },
        {
            "Sid": "CreateCloudTrail",
            "Effect": "Allow",
            "Action": [
                "cloudtrail:CreateTrail",
                "cloudtrail:StartLogging"
            ],
            "Resource": "arn:aws:cloudtrail:*:*:trail/do-not-delete-rds-custom-*"
        },
        {
            "Sid": "CreateS3Bucket",
            "Effect": "Allow",
```

```
            "Action": [
                "s3:CreateBucket",
                "s3:PutBucketPolicy",
                "s3:PutBucketObjectLockConfiguration",
                "s3:PutBucketVersioning"
            ],
            "Resource": "arn:aws:s3:::do-not-delete-rds-custom-*"
        },
        {
            "Sid": "CreateKmsGrant",
            "Effect": "Allow",
            "Action": [
                "kms:CreateGrant",
                "kms:DescribeKey"
            ],
            "Resource": "*"
        }
    ]
}
```

IAM 主体需要以下额外权限，才能使用自定义引擎版本（CEV）：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ConfigureKmsKeyEncryptionPermission",
            "Effect": "Allow",
            "Action": [
                "kms:CreateGrant",
                "kms:DescribeKey",
                "kms:Decrypt",
                "kms:ReEncryptFrom",
                "kms:ReEncryptTo",
                "kms:GenerateDataKeyWithoutPlaintext",
                "kms:GenerateDataKey"
            ],
            "Resource": "arn:aws:kms:region:account_id:key/key_id"
        },
        {
            "Sid": "CreateEc2Instance",
            "Effect": "Allow",
            "Action": [
```

```
                    "ec2:DescribeImages",
                    "ec2:RunInstances",
                    "ec2:CreateTags"
                ],
                "Resource": "*"
            }
        ]
}
```

将 *account_id* 替换为您用来创建实例的账户 ID。将 *region* 替换为您用来创建实例的 Amazon Web Services 区域。将 *key_id* 替换为客户自主管理型密钥 ID。可以根据需要添加多个密钥。

有关启动 EC2 实例所需的资源级权限的更多信息，请参阅启动实例（RunInstances）。

此外，IAM 主体需要对 IAM 角色具有 `iam:PassRole` 权限。必须将其附加到在请求的 `custom-iam-instance-profile` 参数中传递的实例配置文件，以创建 RDS Custom 数据库实例。实例配置文件及其附加的角色稍后将在步骤 2：配置网络、实例配置文件和加密中创建。

> **Note**
> 确保之前列出的权限不受服务控制策略 (SCP)、权限边界或与 IAM 主体关联的会话策略的限制。

步骤 2：配置网络、实例配置文件和加密

您可以使用以下任一过程来配置 IAM 实例配置文件角色、虚拟私有云 (VPC) 和 Amazon KMS 对称加密密钥：

- 使用 Amazon CloudFormation 进行配置（推荐）
- 手动配置

> **Note**
> 如果您的账户属于任一 Amazon Organizations，请确保实例配置文件角色所需的权限不受服务控制策略（SCP）限制。
> 此主题中的网络配置在与不公开访问的数据库实例配合使用时效果最佳。您无法从 VPC 外部直接连接到此类数据库实例。

使用 Amazon CloudFormation 进行配置

为了简化设置，您可以使用 Amazon CloudFormation 模板文件创建 CloudFormation 堆栈。CloudFormation 模板根据 RDS Custom 的要求创建所有网络配置、实例配置文件和加密资源。

要了解如何创建堆栈，请参阅《Amazon CloudFormation 用户指南》中的在 Amazon CloudFormation 控制台上创建堆栈。

有关如何使用 Amazon CloudFormation 模板启动 Amazon RDS Custom for SQL Server 的教程，请参阅 Amazon 数据库博客 中的通过 Amazon CloudFormation 模板开始使用 Amazon RDS Custom for SQL Server。

主题

- CloudFormation 所需的参数
- 下载 Amazon CloudFormation 模板文件
- 使用 CloudFormation 配置资源

CloudFormation 所需的参数

使用 CloudFormation 配置 RDS Custom 必备资源需要以下参数：

参数组	参数名称	默认值	描述
可用性配置	为先决条件设置选择可用性配置	多可用区	指定是在单可用区还是多可用区配置中，为 RDS Custom 实例设置先决条件。如果您在配置中至少需要一个多可用区数据库实例，则应使用多可用区配置
网络配置	为 VPC 指定 IPv4 CIDR 块	10.0.0.0/16	为 VPC 指定 IPv4 CIDR 块（或 IP 地址范围）。此 VPC 配置为创建和使用 RDS Custom 数据库实例。

参数组	参数名称	默认值	描述
	为两个私有子网中的第一个指定 IPv4 CIDR 块	10.0.128.0/20	为您的第一个私有子网指定 IPv4 CIDR 块（或 IP 地址范围）。这是可在其中创建 RDS Custom 数据库实例的两个子网之一。这是一个私有子网，无法访问互联网。
	为两个私有子网中的第二个指定 IPv4 CIDR 块	10.0.144.0/20	为第二个私有子网指定 IPv4 CIDR 块（或 IP 地址范围）。这是可在其中创建 RDS Custom 数据库实例的两个子网之一。这是一个私有子网，无法访问互联网。
	为公有子网指定 IPv4 CIDR 块	10.0.0.0/20	为您的公有子网指定 IPv4 CIDR 块（或 IP 地址范围）。这是可以在其中创建 EC2 实例与 RDS Custom 数据库实例连接的子网之一。这是一个可以访问互联网的公有子网。

参数组	参数名称	默认值	描述
RDP 访问配置	您的源的 IPv4 CIDR 块	-	指定您的源的 IPv4 CIDR 块（或 IP 地址范围）。这是您与公有子网中的 EC2 实例建立 RDP 连接的 IP 范围。如果未设置，则不配置与 EC2 实例的 RDP 连接。
	设置 RDP 对 RDS Custom for SQL Server 实例的访问权限	否	指定是否建立从 EC2 实例到 RDS Custom for SQL Server 实例的 RDP 连接。默认情况下，不配置从 EC2 实例到数据库实例的 RDP 连接。

CloudFormation 创建的资源

使用默认设置成功创建 CloudFormation 堆栈会在您的 Amazon Web Services 账户中创建以下资源：

- 对称加密 KMS 密钥，用于加密 RDS Custom 管理的数据。
- 使用 `AmazonRDSCustomInstanceProfileRolePolicy` 将实例配置文件与 IAM 角色相关联，以提供 RDS Custom 所需的权限。有关更多信息，请参阅《Amazon 托管策略参考指南》中的 AmazonRDSCustomServiceRolePolicy。
- 将 CIDR 范围指定为 CloudFormation 参数的 VPC。默认值为 `10.0.0.0/16`。
- 在参数中指定了 CIDR 范围的两个私有子网，以及 Amazon Web Services 区域中的两个不同可用区。子网 CIDR 的默认值为 `10.0.128.0/20` 和 `10.0.144.0/20`。
- 一个在参数中指定了 CIDR 范围的公有子网。子网 CIDR 的默认值为 10.0.0.0/20。EC2 实例位于此子网中，可用于连接到 RDS Custom 实例。
- 为 VPC 设置了 DHCP 选项，域名解析为 Amazon 域名系统 (DNS) 服务器。
- 与两个私有子网关联但无法访问互联网的路由表。
- 与公有子网关联且可以访问互联网的路由表。

- 与 VPC 关联且允许互联网访问公有子网的互联网网关。
- 网络访问控制列表（ACL），与两个私有子网关联并且在 VPC 内只能访问 HTTPS 和数据库端口。
- 要与 RDS Custom 实例关联的 VPC 安全组。限制对 RDS Custom 所需的 Amazon Web Services 服务端点的出站 HTTPS 访问，以及对 EC2 实例安全组的入站数据库端口的访问。
- 要与公有子网中的 EC2 实例关联的 VPC 安全组。限制出站数据库端口对 RDS Custom 实例安全组的访问。
- 对于为 RDS Custom 所需的 Amazon Web Services 服务端点创建的 VPC 端点，要与它们关联的 VPC 安全组。
- 在其中创建 RDS Custom 实例的数据库子网组。此模板创建的两个私有子网已添加到数据库子网组中。
- 对应于 RDS Custom 所需的每个 Amazon Web Services 服务端点的 VPC 端点。

除了上述列表之外，将可用性配置设置为多可用区还会创建以下资源：

- 允许私有子网之间通信的网络 ACL 规则。
- 对与 RDS Custom 实例关联的 VPC 安全组中多可用区端口的入站和出站访问权限。
- 与进行多可用区通信所需的 Amazon 服务端点连接的 VPC 端点。

此外，设置 RDP 访问配置还会创建以下资源：

- 配置从您的源 IP 地址对公有子网的 RDP 访问权限：
  - 允许建立从您的源 IP 到公有子网的 RDP 连接的网络 ACL 规则。
  - 对从您的源 IP 连接到与 EC2 实例关联的 VPC 安全组的 RDP 端口进行入口访问。
- 配置从公有子网中的 EC2 实例到私有子网中的 RDS Custom 实例的 RDP 访问权限：
  - 允许建立从公有子网到私有子网的 RDP 连接的网络 ACL 规则。
  - 对从与 EC2 实例关联的 VPC 安全组连接到与 RDS Custom 实例关联的 VPC 安全组的 RDP 端口进行入站访问

使用以下过程为 RDS Custom for SQL Server 创建 CloudFormation 堆栈。

下载 Amazon CloudFormation 模板文件

下载模板文件

1. 打开链接 custom-sqlserver-onboard.zip 的上下文（右键单击）菜单，然后选择 Save Link As（将链接另存为）。
2. 将文件保存并提取到您的计算机中。

使用 CloudFormation 配置资源

使用 CloudFormation 配置资源

1. 通过以下网址打开 CloudFormation 控制台：https://console.aws.amazon.com/cloudformation。
2. 要启动 Create Stack（创建堆栈）向导，请选择 Create Stack（创建堆栈）。

   此时会显示 Create stack（创建快照）。
3. 对于 Prerequisite - Prepare template（先决条件 - 准备模板），请选择 Template is ready（模板已就绪）。
4. 对于 Select Template（选择模板），执行以下操作：

   a. 对于 Template source(模板来源)，选择 Upload a template file(上载模板文件)。
   b. 对于选择文件，请导航到并选择正确的文件。
5. 选择下一步。

   此时将出现 Specify stack details（指定堆栈详细信息）页面。
6. 对于堆栈名称，输入 **`rds-custom-sqlserver`**。
7. 对于 Parameters（参数），请执行以下操作：

   a. 要保留原定设置选项，请选择 Next（下一步)。
   b. 要更改选项，请选择相应的可用性配置、网络配置和 RDP 访问配置，然后选择下一步。

      在更改参数之前，请仔细阅读每个参数的说明。

> **Note**
>
> 如果您选择在此 CloudFormation 堆栈中至少创建一个多可用区实例，请确保 CloudFormation 堆栈参数为先决条件设置选择可用性配置设置为 `Multi-AZ`。如果

> 您将 CloudFormation 堆栈创建为单可用区，请在创建第一个多可用区实例之前将 CloudFormation 堆栈更新为多可用区配置。

8. 在配置堆栈选项页面上，请选择下一步。
9. 在 Review rds-custom-sqlserver（查看 rds-custom-sqlserver）页面上，请执行以下操作：

   a. 对于 Capabilities（功能），请选中 I acknowledge that Amazon CloudFormation might create IAM resources with custom names（我确认 Amazon CloudFormation 可能使用自定义名称创建 IAM 资源）复选框。

   b. 选择创建堆栈。

> **Note**
>
> 请勿直接从资源页面更新从该 Amazon CloudFormation 堆栈创建的资源。这可以防止您使用 Amazon CloudFormation 模板将后续更新应用于这些资源。

CloudFormation 创建 RDS Custom for SQL Server 所需的资源。如果堆栈创建失败，请通读 Events（事件）选项卡，以查看哪项资源创建失败及其状态原因。

对于将作为参数传递以创建 RDS Custom for SQL Server 数据库实例的所有资源，控制台中此 CloudFormation 堆栈的 Outputs（输出）选项卡应包含有关它们的信息。确保使用 CloudFormation 为 RDS Custom 数据库实例创建的 VPC 安全组和数据库子网组。原定设置情况下，RDS 会尝试附加原定设置 VPC 安全组，该安全组可能没有您所需的访问权限。

如果您使用了 CloudFormation 来创建资源，则可以跳过手动配置。

更新 CloudFormation 堆栈

您还可以在创建后更新 CloudFormation 堆栈的一些相关配置。可以更新的配置包括：

- RDS Custom for SQL Server 的可用性配置
  - 为先决条件设置选择可用性配置：更新此参数可在单可用区和多可用区配置之间切换。如果您将此 CloudFormation 堆栈用于至少一个多可用区实例，则必须更新该堆栈以选择多可用区配置。
- RDS Custom for SQL Server 的 RDP 访问配置
  - 您的源的 IPv4 CIDR 块：您可以通过更新此参数来更新您的源的 IPv4 CIDR 块（或 IP 地址范围）。将此参数设置为空白会删除从源 CIDR 块到公有子网的 RDP 访问配置。

- 设置对 RDS Custom for SQL Server 的 RDP 访问：启用或禁用从 EC2 实例到 RDS Custom for SQL Server 实例的 RDP 连接。

删除 CloudFormation 堆栈

在从 CloudFormation 堆栈中删除所有使用资源的 RDS Custom 实例后，即可删除该堆栈。RDS Custom 不会跟踪 CloudFormation 堆栈，因此当存在使用堆栈资源的数据库实例时，它不会阻止删除该堆栈。在删除该堆栈时，请确保没有使用堆栈资源的 RDS Custom 数据库实例。

> **Note**
>
> 当您删除 CloudFormation 堆栈时，该堆栈创建的所有资源都会被删除，但 KMS 密钥除外。KMS 密钥会进入等待删除状态并且将在 30 天后删除。要保留 KMS 密钥，请在 30 天宽限期内执行 CancelKeyDeletion 操作。

手动配置

如果您选择手动配置资源，请执行以下任务。

> **Note**
>
> 为了简化设置，您可以使用 Amazon CloudFormation 模板文件来创建 CloudFormation 堆栈，而不是手动配置。有关更多信息，请参阅 使用 Amazon CloudFormation 进行配置。
> 也可以使用 Amazon CLI 来完成本部分的操作。如果是，请下载并安装最新的 CLI。

主题

- 请确保您有对称的加密 Amazon KMS 密钥
- 手动创建您的 IAM 角色和实例配置文件
- 手动配置您的 VPC

请确保您有对称的加密 Amazon KMS 密钥

RDS Custom 需要对称的加密 Amazon KMS key。在创建 RDS Custom for SQL Server 数据库实例时，确保提供 KMS 密钥标识符做为参数 `kms-key-id`。有关更多信息，请参阅 为 Amazon RDS Custom for SQL Server 创建并连接到数据库实例。

您有以下选项：

- 如果您的 Amazon Web Services 账户中有现有的客户自主管理型 KMS 密钥，则可以将其与 RDS Custom 一起使用。无需进一步操作。
- 如果您已经为不同的 RDS Custom 引擎创建了客户自主管理型对称加密 KMS 密钥，可以重用相同的 KMS 密钥。无需进一步操作。
- 如果您的账户中没有现有的客户自主管理型对称加密 KMS 密钥，请按照《Amazon Key Management Service 开发人员指南》的创建密钥中的说明创建一个 KMS 密钥。
- 如果您正在创建 CEV 或 RDS Custom 数据库实例，而您的 KMS 密钥位于其他 Amazon Web Services 账户 中，请务必使用 Amazon CLI。您不能将 Amazon 控制台与跨账户 KMS 密钥一起使用。

> **Important**
>
> RDS Custom 不支持 Amazon 托管式 KMS 密钥。

确保您的对称加密密钥为 IAM 实例配置文件中的 `kms:Decrypt`（IAM）角色授予对 `kms:GenerateDataKey` 和 Amazon Identity and Access Management 操作的访问权限。如果您的账户中有一个新的对称加密密钥，不需要进行任何更改。否则，请确保对称加密密钥的策略授予对这些操作的访问权限。

有关更多信息，请参阅 步骤 4：为 RDS Custom for Oracle 配置 IAM。

手动创建您的 IAM 角色和实例配置文件

可以手动创建实例配置文件并使用它来启动 RDS Custom 实例。如果计划在 Amazon Web Services Management Console中创建实例，请跳过此部分。Amazon Web Services Management Console允许您创建实例配置文件并将其附加到 RDS Custom 数据库实例。有关更多信息，请参阅 使用 Amazon Web Services Management Console自动创建实例配置文件。

手动创建实例配置文件时，请将实例配置文件名称作为 `custom-iam-instance-profile` 参数传递给 `create-db-instance` CLI 命令。RDS Custom 使用与此实例配置文件关联的角色，通过运行自动化功能来管理实例。

为 RDS Custom for SQL Server 创建 IAM 实例配置文件和 IAM 角色

1. 创建名为 `AWSRDSCustomSQLServerInstanceRole` 的 IAM 角色，并带有信任策略，以便 Amazon EC2 可代入此角色。
2. 将 Amazon 托管策略 `AmazonRDSCustomInstanceProfileRolePolicy` 添加到 `AWSRDSCustomSQLServerInstanceRole`。
3. 为名为 `AWSRDSCustomSQLServerInstanceProfile` 的 RDS Custom for SQL Server 创建 IAM 实例配置文件。
4. 将 `AWSRDSCustomSQLServerInstanceRole` 添加到实例配置文件。

创建 AWSRDSCustomSQLServerInstanceRole IAM 角色

以下示例将创建 `AWSRDSCustomSQLServerInstanceRole` 角色。信任策略将允许 Amazon EC2 代入该角色。

```
aws iam create-role \
    --role-name AWSRDSCustomSQLServerInstanceRole \
    --assume-role-policy-document '{
        "Version": "2012-10-17",
          "Statement": [
            {
              "Action": "sts:AssumeRole",
              "Effect": "Allow",
              "Principal": {
                  "Service": "ec2.amazonaws.com"
              }
            }
          ]
        }'
```

将访问策略添加到 AWSRDSCustomSQLServerInstanceRole

要提供所需的权限，请将 Amazon 托管策略 `AmazonRDSCustomInstanceProfileRolePolicy` 附加到 `AWSRDSCustomSQLServerInstanceRole`。`AmazonRDSCustomInstanceProfileRolePolicy` 允许 RDS Custom 实例发送和接收消息，并执行各种自动化操作。

> **Note**
>
> 确保访问策略中的权限不受 SCP 或与实例配置文件角色关联的权限边境所限制。

以下示例将 Amazon 托管策略 `AWSRDSCustomSQLServerIamRolePolicy` 附加到 `AWSRDSCustomSQLServerInstanceRole` 角色。

```
aws iam attach-role-policy \
    --role-name AWSRDSCustomSQLServerInstanceRole \
    --policy-arn arn:aws:iam::aws:policy/AmazonRDSCustomInstanceProfileRolePolicy
```

创建您的 RDS Custom for SQL Server 实例配置文件

实例配置文件是包含单个 IAM 角色的容器。RDS Custom 使用实例配置文件将角色传递给实例。

如果使用 Amazon Web Services Management Console创建 Amazon EC2 角色，该控制台会自动创建实例配置文件，将其命名为与所创建角色相同的名称。请按如下方式创建您的实例配置文件，将其命名为 `AWSRDSCustomSQLServerInstanceProfile`。

```
aws iam create-instance-profile \
    --instance-profile-name AWSRDSCustomSQLServerInstanceProfile
```

请将 AWSRDSCustomSQLServerInstanceRole 添加到您的 RDS Custom for SQL Server 实例配置文件中

将 `AWSRDSCustomInstanceRoleForRdsCustomInstance` 角色添加到先前创建的 `AWSRDSCustomSQLServerInstanceProfile` 配置文件中。

```
aws iam add-role-to-instance-profile \
    --instance-profile-name AWSRDSCustomSQLServerInstanceProfile \
    --role-name AWSRDSCustomSQLServerInstanceRole
```

手动配置您的 VPC

就像 Amazon EC2 实例或 Amazon RDS 实例一样，您的 RDS Custom 数据库实例位于基于 Amazon VPC 服务的虚拟私有云 (VPC) 中。您将提供并配置自己的 VPC。因此，您将能够完全控制实例联网设置。

RDS Custom 将通信从数据库实例发送到其他 Amazon Web Services 服务。确保可以从您在其中创建 RDS Custom 数据库实例的子网访问以下服务：

- Amazon CloudWatch (`com.amazonaws.region.monitoring`)
- Amazon CloudWatch Logs (`com.amazonaws.region.logs`)
- Amazon CloudWatch Events (`com.amazonaws.region.events`)
- Amazon EC2 (`com.amazonaws.region.ec2` 和 `com.amazonaws.region.ec2messages`)
- Amazon EventBridge (`com.amazonaws.region.events`)
- Amazon S3 (`com.amazonaws.region.s3`)
- Amazon Secrets Manager (`com.amazonaws.region.secretsmanager`)
- Amazon Systems Manager (`com.amazonaws.region.ssm` 和 `com.amazonaws.region.ssmmessages`)

如果创建多可用区部署

- Amazon Simple Queue Service (`com.amazonaws.region.sqs`)

如果 RDS Custom 无法与必需的服务通信，它会发布以下事件：

```
Database instance in incompatible-network. SSM Agent connection not available. Amazon
 RDS can't connect to the dependent Amazon services.
```

```
Database instance in incompatible-network. Amazon RDS can't connect to dependent Amazon
 services. Make sure port 443 (HTTPS) allows outbound connections, and try again.
 "Failed to connect to the following services: s3 events"
```

为避免 `incompatible-network` 错误，请确保您的 RDS Custom 数据库实例与 Amazon Web Services 服务之间通信所涉及的 VPC 组件满足以下要求：

- 数据库实例可以在端口 443 上建立到其他 Amazon Web Services 服务的出站连接。
- VPC 允许对源自 RDS Custom 数据库实例的请求进行传入响应。
- RDS Custom 可以正确解析每个 Amazon Web Services 服务的端点的域名。

如果您已经为其他 RDS Custom 数据库引擎配置了 VPC，可以重用该 VPC 并跳过此过程。

主题

- [配置您的 VPC 安全组]

- 为依赖项 Amazon Web Services 服务配置端点
- 配置实例元数据服务

配置您的 VPC 安全组

安全组充当 VPC 实例的虚拟防火墙，以控制入站和出站流量。RDS Custom 数据库实例在其网络接口上附加了一个用于保护该实例的安全组。请确保该安全组允许通过 HTTPS 在 RDS Custom 和其他 Amazon Web Services 服务之间传输流量。可以将此安全组作为实例创建请求中的 `vpc-security-group-ids` 参数进行传递。

为 RDS Custom 配置安全组

1. 登录到Amazon Web Services Management Console并打开 Amazon VPC 控制台，网址：https://console.aws.amazon.com/vpc。
2. 允许 RDS Custom 使用原定设置安全组，或创建您自己的安全组。

   有关详细说明，请参阅通过创建安全组提供对 VPC 中的数据库实例的访问。
3. 确保您的安全组允许在端口 443 上建立出站连接。RDS Custom 需要此端口来与依赖项 Amazon Web Services 服务进行通信。
4. 如果您有私有 VPC 并且使用 VPC 端点，请确保与数据库实例关联的安全组允许在端口 443 上建立到 VPC 端点的出站连接。还要确保与 VPC 端点关联的安全组允许在端口 443 上建立来自数据库实例的入站连接。

   如果不允许传入连接，RDS Custom 实例将无法连接到 Amazon Systems Manager 和 Amazon EC2 端点。有关更多信息，请参阅《Amazon Systems Manager 用户指南》中的创建虚拟私有云端点。
5. 对于 RDS Custom for SQL Server 多可用区实例，请确保与数据库实例关联的安全组允许在端口 1120 上建立与该安全组本身的入站和出站连接。这是在多可用区 RDS Custom for SQL Server 数据库实例上建立对等主机连接所必需的。

有关安全组的更多信息，请参阅 Amazon VPC 开发人员指南中的您的 VPC 的安全组。

为依赖项 Amazon Web Services 服务配置端点

我们建议您按照以下说明为 VPC 中的每项服务添加端点。但是，您也可以使用任何其他解决方案，只要使您的 VPC 能够与 Amazon 服务端点通信即可。例如，您可以使用网络地址转换 (NAT) 或 Amazon Direct Connect。

为与 RDS Custom 配合使用的 Amazon Web Services 服务配置端点

1. 通过 https://console.aws.amazon.com/vpc/ 打开 Amazon VPC 控制台。
2. 在导航栏中，使用区域选择器来选择 Amazon Web Services 区域。
3. 在导航窗格中，选择端点。在主窗格中，选择创建端点。
4. 对于 Service category（服务类别），选择 Amazon Web Services 服务。
5. 对于 Service Name（服务名称），请选择表中显示的端点。
6. 对于 VPC，选择您的 VPC。
7. 对于子网，请从每个可用区中选择一个要包括的子网。

   VPC 端点可以跨越多个可用区。Amazon 为您选择的每个子网中的 VPC 端点创建一个弹性网络接口。每个网络接口有一个域名系统 (DNS) 主机名和私有 IP 地址。
8. 对于 Security group（安全组），请选择或创建一个安全组。

   您可以使用安全组来控制对您的端点的访问，就像您使用防火墙一样。确保安全组允许端口 443 上来自数据库实例的入站连接。有关安全组的更多信息，请参阅 Amazon VPC 用户指南 中的您的 VPC 的安全组。
9. 或者，您可以将策略附加到 VPC 端点。端点策略可以控制对您要连接到的 Amazon Web Services 服务的访问权限。原定设置策略允许所有请求通过端点。如果您使用自定义策略，请确保策略中允许来自数据库实例的请求。
10. 选择创建端点。

下表说明了如何查找您的 VPC 用于出站通信所需的端点列表。

服务	端点格式	备注和链接
Amazon Systems Manager	采用以下端点格式： • `ssm.region.amazonaws.com` • `ssmmessages.region.amazonaws.com`	有关每个区域中端点的列表，请参阅《Amazon Web Services 一般参考》中的 Amazon Systems Manager 端点和限额。
Amazon Secrets Manager	采用端点格式 `secretsmanager.region.amazonaws.com`。	有关每个区域中端点的列表，请参阅《Amazon Web Services 一般参考》中的 Amazon Secrets Manager 端点和限额。

服务	端点格式	备注和链接
Amazon CloudWatch	采用以下端点格式： • 对于 CloudWatch 指标，请使用 `monitoring. region.amazonaws.com` • 对于 CloudWatch Events，请使用 `events.region.amazonaws.com` • 对于 CloudWatch Logs，请使用 `logs.region.amazonaws.com`	有关每个区域中的端点列表，请参阅： • 《Amazon Web Services 一般参考》中的 Amazon CloudWatch 端点和限额 • 《Amazon Web Services 一般参考》中的 Amazon CloudWatch Logs 端点和限额 • 《Amazon Web Services 一般参考》中的 Amazon CloudWatch Events 端点和限额
Amazon EC2	采用以下端点格式： • `ec2.region.amazonaws.com` • `ec2messages. region.amazonaws.com`	有关每个区域中端点的列表，请参阅《Amazon Web Services 一般参考》中的 Amazon Elastic Compute Cloud 端点和限额。
Amazon S3	采用端点格式 `s3.region.amazonaws.com` 。	有关每个区域中端点的列表，请参阅《Amazon Web Services 一般参考》中的 Amazon Simple Storage Service 端点和限额。 要了解 Amazon S3 的网关端点的更多信息，请参阅《Amazon VPC 开发人员指南》中的 Amazon S3 端点。 要了解如何创建访问点，请参阅《Amazon VPC 开发人员指南》中的创建访问点。 要了解如何为 Amazon S3 创建网关端点，请参阅网关 VPC 端点。

服务	端点格式	备注和链接
Amazon Simple Queue Service	采用端点格式 sqs.*region*.amazonaws.com	要查看每个区域中端点的列表，请参阅 Amazon Simple Storage Service 端点和限额

配置实例元数据服务

请确保您的实例可以执行以下操作：

- 可使用 Instance Metadata Service Version 2 (IMDSv2) 来访问实例元数据服务。
- 允许通过端口 80 (HTTP) 到 IMDS 链接 IP 地址的出站通信。
- 从 `http://169.254.169.254` IMDSv2 链接请求实例元数据。

有关更多信息，请参阅《Amazon EC2 用户指南》中的使用 IMDSv2。

跨实例限制

当您按照上述步骤创建实例配置文件时，系统会使用 Amazon 托管策略 `AmazonRDSCustomInstanceProfileRolePolicy` 向 RDS Custom 提供实现实例管理和监控自动化所需的权限。该托管策略可确保这些权限仅允许访问 RDS Custom 运行自动化功能所需的资源。建议使用该托管策略来支持新功能并满足安全要求，这些要求会自动应用于现有的实例配置文件，无需人工干预。有关更多信息，请参阅Amazon 托管策略：AmazonRDSCustomInstanceProfileRolePolicy。

`AmazonRDSCustomInstanceProfileRolePolicy` 托管策略将实例配置文件限制为具有跨账户访问权限，但它可能允许在同一账户内跨 RDS Custom 实例访问某些 RDS Custom 托管资源。根据您的要求，您可以使用权限边界进一步限制跨实例访问。权限边界策略定义基于身份的策略可以授予实体的最大权限，但它本身并不授予权限。有关更多信息，请参阅评估有边界的有效权限。

例如，以下边界策略限制实例配置文件角色对特定 Amazon KMS 密钥的访问权限，并限制跨使用不同 Amazon KMS 密钥的实例对 RDS Custom 托管式资源的访问权限。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyOtherKmsKeyAccess",
            "Effect": "Deny",
```

```
            "Action": "kms:*",
            "NotResource": "arn:aws:kms:region:acct_id:key/KMS_key_ID"
        },
        {
            "Sid": "NoBoundarySetByDefault",
            "Effect": "Allow",
            "Action": "*",
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> 请确保权限边界不会阻止 `AmazonRDSCustomInstanceProfileRolePolicy` 向 RDS Custom 授予的任何权限。

对于 RDS Custom for SQL Server 使用自带媒体

RDS Custom for SQL Server 支持两种许可模式：附带许可证（LI）和自带媒体（BYOM）。

利用 BYOM，您可以执行以下操作：

1. 在 Amazon EC2 Windows AMI 上提供并安装您自己的 Microsoft SQL Server 二进制文件以及支持的累积更新（CU）。
2. 将 AMI 另存为黄金镜像，该镜像是可用于创建自定义引擎版本（CEV）的模板。
3. 从您的黄金镜像创建 CEV。
4. 使用 CEV 创建新的 RDS Custom for SQL Server 数据库实例。

然后，Amazon RDS 会为您管理您的数据库实例。

> **Note**
> 如果您还有一个附带许可证（LI）的 RDS Custom for SQL Server 数据库实例，则无法将此数据库实例中的 SQL Server 软件与 BYOM 一起使用。您必须将自己的 SQL Server 二进制文件放入 BYOM 中。

对 RDS Custom for SQL Server 使用 BYOM 的要求

使用 RDS Custom for SQL Server 的自定义引擎版本的相同一般要求也适用于 BYOM。有关更多信息，请参阅 RDS Custom for SQL Server CEV 的要求。

使用 BYOM 时，请确保满足以下附加要求：

- 使用以下支持的版本之一：SQL Server 2022 或 2019 企业版、标准版或开发版。
- 将 SQL Server sysadmin（SA）服务器角色权限授予 `NT AUTHORITY\SYSTEM`。
- 保持 EC2 Windows Server 操作系统配置为使用 UTC 时间。

  默认情况下，基础 Amazon EC2 Windows 实例设置为 UTC 时区。有关查看和更改 Windows 实例的时间的更多信息，请参阅《Amazon EC2 用户指南》中的为 Windows 实例设置时间。
- 打开 TCP 端口 1433 和 UDP 端口 1434 以允许 SSM 连接。

对 RDS Custom for SQL Server 使用 BYOM 的限制

RDS Custom for SQL Server 的相同一般限制也适用于 BYOM。有关更多信息，请参阅 Amazon RDS Custom for SQL Server 的要求和限制。

对于 BYOM，以下附加限制适用：

- 仅支持原定设置 SQL Server 实例（MSSQLSERVER）。不支持已命名的 SQL Server 实例。RDS Custom for SQL Server 仅检测和监控原定设置 SQL Server 实例。
- 每个 AMI 仅支持 SQL Server 的单一安装。不支持多次安装不同的 SQL Server 版本。
- BYOM 不支持 SQL Server Web 版本。
- BYOM 不支持 SQL Server 各版本的评估版。安装 SQL Server 时，请不要选中使用评估版的复选框。
- 功能可用性和支持因每个数据库引擎的特定版本以及 Amazon Web Services 区域而异。有关更多信息，请参阅RDS Custom for SQL Server CEV 的区域可用性 和RDS Custom for SQL Server CEV 的版本支持。

使用 BYOM 创建 RDS Custom for SQL Server 数据库实例

要使用 BYOM 准备和创建 RDS Custom for SQL Server 数据库实例，请参阅使用自带媒体（BYOM）准备 CEV。

使用适用于 RDS Custom for SQL Server 的自定义引擎版本

RDS Custom for SQL Server 的自定义引擎版本（CEV）是包括 Microsoft SQL Server 的亚马逊云机器镜像（AMI）。

CEV 工作流程的基本步骤如下：

1. 选择 Amazon EC2 Windows AMI 以用作 CEV 的基础镜像。您可以选择使用预安装的 Microsoft SQL Server，也可以自带媒体以自行安装 SQL Server。
2. 在操作系统（OS）上安装其他软件，并自定义操作系统和 SQL Server 的配置以满足您的企业需求。
3. 将 AMI 另存为黄金镜像
4. 从黄金镜像创建自定义引擎版本（CEV）。
5. 使用 CEV 创建新的 RDS Custom for SQL Server 数据库实例。

然后，Amazon RDS 会为您管理这些数据库实例。

CEV 允许您维护操作系统和数据库的首选基准配置。使用 CEV 可确保主机配置（如任何第三方代理安装或其他操作系统自定义）在 RDS Custom for SQL Server 数据库实例上持续存在。使用 CEV，您可以使用相同配置快速部署 RDS Custom for SQL Server 数据库实例集。

主题

正在准备创建适用于 RDS Custom for SQL Server 的 CEV

您可以使用包含预安装、附带许可证（LI）的 Microsoft SQL Server 的亚马逊机器映像（AMI）来创建 CEV，也可以使用安装您自己的 SQL Server 安装媒体（BYOM）的 AMI 来创建 CEV。

准备 CEV

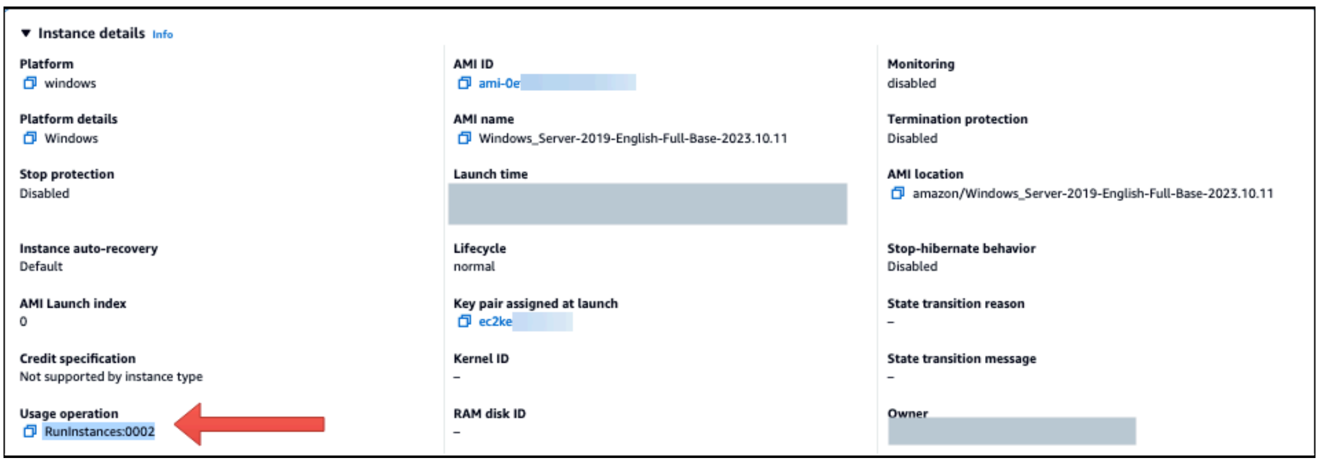
使用以下过程，通过自带媒体（BYOM）或预安装的 Microsoft SQL Server（LI）创建 CEV。

使用自带媒体（BYOM）准备 CEV

以下步骤以 Windows Server 2019 Base 的 AMI 作为示例。

使用 BYOM 创建 CEV

1. 从 Amazon EC2 控制台控制中，选择启动实例。
2. 对于名称，输入实例的名称。
3. 在“快速入门”下，选择 Windows。
4. 选择 Microsoft Windows Server 2019 Base。
5. 选择适当的实例类型、键对、网络和存储设置，然后启动实例。
6. 启动或创建 EC2 实例后，请确保从步骤 4 中选择了正确的 Windows AMI：
    a. 在 Amazon EC2 控制台选择 EC2 实例。
    b. 在详细信息部分，检查使用情况操作并确保将其设置为 RunInstances:0002。



7. 登录 EC2 实例并将您的 SQL Server 安装媒体复制到该实例。

> **Note**
>
> 如果您使用 SQL Server 开发版构建 CEV，则可能需要使用 Microsoft Visual Studio 订阅获取安装媒体。

8. 安装 SQL Server。务必执行以下操作：
    a. 查看对 RDS Custom for SQL Server 使用 BYOM 的要求和RDS Custom for SQL Server CEV 的版本支持。

b. 将实例根目录设置为默认值 `C:\Program Files\Microsoft SQL Server\`。不要更改此目录。

c. 将 SQL Server 数据库引擎账户名称设置为 `NT Service\MSSQLSERVER` 或 `NT AUTHORITY\NETWORK SERVICE`。

d. 将 SQL Server 启动模式设置为手动。

e. 将 SQL Server 身份验证模式选择为混合。

f. 保留默认数据目录和 TempDB 位置的当前设置。

9. 将 SQL Server sysadmin（SA）服务器角色权限授予 `NT AUTHORITY\SYSTEM`：

```
USE [master]
GO
EXEC master..sp_addsrvrolemember @loginame = N'NT AUTHORITY\SYSTEM' , @rolename =
 N'sysadmin'
GO
```

10. 安装其他软件，或自定义操作系统和数据库配置以满足您的要求。

11. 在 EC2 实例上运行 Sysprep。有关更多信息，请参阅使用 Windows Sysprep 创建 Amazon EC2 AMI。

12. 保存包含您已安装的 SQL Server 版本、其他软件和自定义项的 AMI。这将是您的黄金镜像。

13. 通过提供您创建的映像的 AMI ID 来创建新的 CEV。有关详细步骤，请参阅创建适用于 RDS Custom for SQL Server 的 CEV。

14. 使用 CEV 创建新的 RDS Custom for SQL Server 数据库实例。有关详细步骤，请参阅从 CEV 创建 RDS Custom for SQL Server 数据库实例。

使用预安装的 SQL Server（LI）准备 CEV

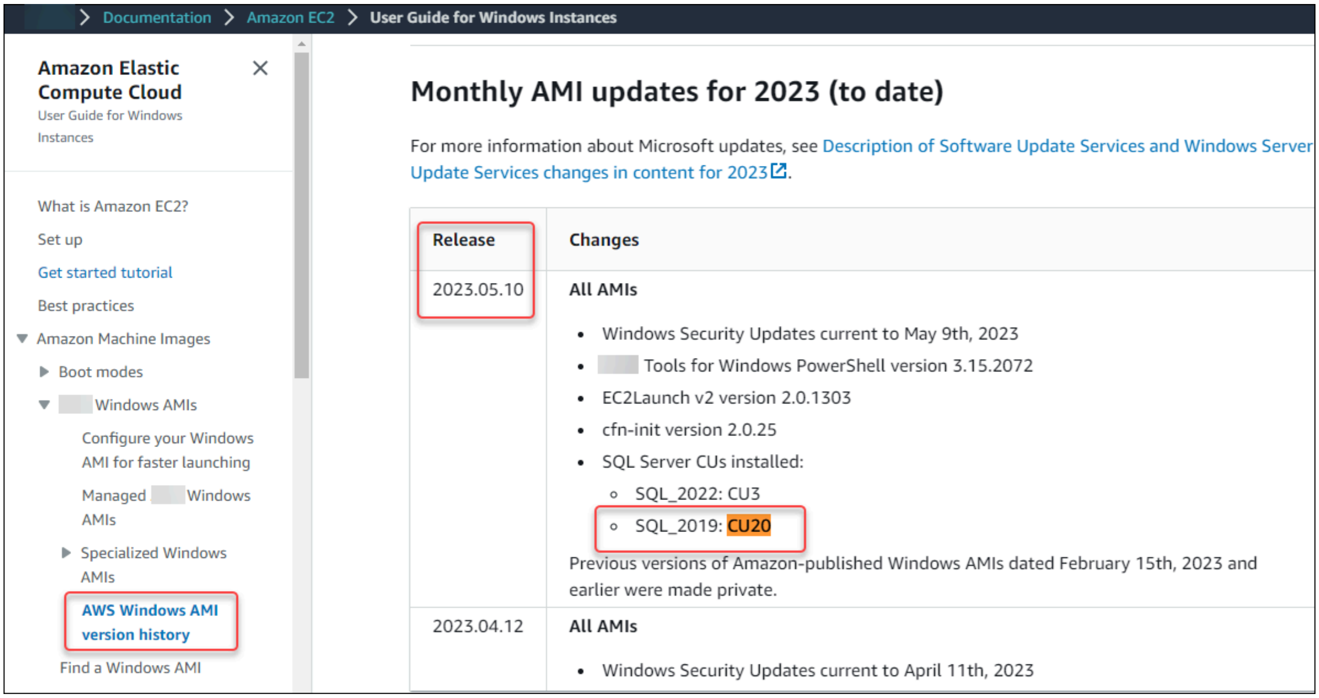
使用预安装的 Microsoft SQL Server（LI）创建 CEV 的以下步骤以 SQL Server CU20 版本号为 `2023.05.10` 的 AMI 作为示例。创建 CEV 时，请选择具有最新版本号的 AMI。这可以确保您使用的是支持的 Windows Server 和 SQL Server 版本 [具有最新的累积更新（CU）]。

使用预安装的 Microsoft SQL Server（LI）创建 CEV

1. 选择最新可用的 Amazon EC2 Windows 亚马逊机器映像（AMI），其中带有附带许可证（LI）的 Microsoft Windows Server 和 SQL Server。

   a. 在 Windows AMI 版本历史记录中搜索 CU20。

b. 记下版本号。对于 SQL Server 2019 CU20，版本号为 `2023.05.10`。



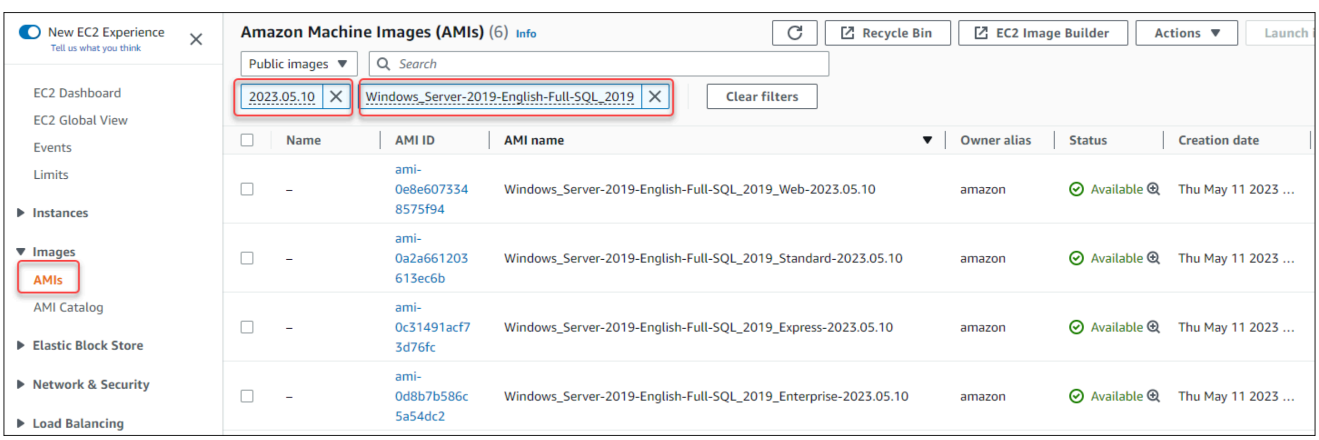
c. 通过以下网址打开 Amazon EC2 控制台：https://console.aws.amazon.com/ec2/。

d. 在 Amazon EC2 控制台的左侧导航窗格中，选择 Images（映像），然后选择 AMI。

e. 选择公有映像。

f. 在搜索框中输入 `2023.05.10`。随即出现 AMI 的列表。

g. 在搜索框中输入 `Windows_Server-2019-English-Full-SQL_2019` 以筛选结果。应显示以下结果。



h. 选择要使用的 AMI 以及 SQL Server 版本。

2. 从您选择的 AMI 创建或启动 EC2 实例。

3. 登录 EC2 实例并安装其他软件，或自定义操作系统和数据库配置以满足您的要求。

4. 在 EC2 实例上运行 Sysprep。有关使用 Sysprep 准备 AMI 的更多信息，请参阅使用 Sysprep 创建标准化的亚马逊机器映像（AMI）。
5. 保存包含您已安装的 SQL Server 版本、其他软件和自定义项的 AMI。这将是您的黄金镜像。
6. 通过提供您创建的映像的 AMI ID 来创建新的 CEV。有关创建 CEV 的详细步骤，请参阅 创建适用于 RDS Custom for SQL Server 的 CEV。
7. 使用 CEV 创建新的 RDS Custom for SQL Server 数据库实例。有关详细步骤，请参阅从 CEV 创建 RDS Custom for SQL Server 数据库实例。

RDS Custom for SQL Server CEV 的区域可用性

以下 Amazon Web Services 区域中提供对 RDS Custom for SQL Server 的自定义引擎版本（CEV）的支持：

- 美国东部（俄亥俄州）
- 美国东部（弗吉尼亚州北部）
- 美国西部（俄勒冈州）
- 美国西部（加利福尼亚北部）
- 亚太地区（孟买）
- 亚太地区（大阪）
- 亚太地区（首尔）
- 亚太地区（新加坡）
- 亚太地区（悉尼）
- 亚太地区（东京）
- 加拿大（中部）
- 欧洲地区（法兰克福）
- 欧洲地区（爱尔兰）
- 欧洲地区（伦敦）
- 欧洲地区（巴黎）
- 欧洲地区（斯德哥尔摩）
- 南美洲（圣保罗）

RDS Custom for SQL Server CEV 的版本支持

以下 Amazon EC2 Windows AMI 支持创建适用于 RDS Custom for SQL Server 的 CEV：

- 对于使用预安装媒体的 CEV，带有附带许可证（LI）的 Microsoft Windows Server 2019（操作系统）和 SQL Server 2022 或 2019 的 Amazon EC2 Windows AMI
- 对于使用自带媒体（BYOM）的 CEV，带有 Microsoft Windows Server 2019（操作系统）的 Amazon EC2 Windows AMI

以下操作系统（OS）和数据库版本支持创建适用于 RDS Custom for SQL Server 的 CEV：

- 对于使用预装媒体的 CEV：
  - 带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或 Web 版。
  - 带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或 Web 版。
- 对于使用自带媒体（BYOM）的 CEV，
  - 带有 CU9、CU13、CU14-GDR、CU15-GDR、CU16、CU17 和 CU18 的 SQL Server 2022 企业版、标准版或开发版。
  - 带有 CU8、CU17、CU18、CU20、CU24、CU26、CU28-GDR、CU29-GDR、CU30 和 CU32 的 SQL Server 2019 企业版、标准版或开发版。
- 对于使用预安装媒体或自带媒体（BYOM）的 CEV，Windows Server 2019 是唯一支持的操作系统。

有关更多信息，请参阅Amazon Windows AMI 版本历史记录。

RDS Custom for SQL Server CEV 的要求

以下要求应用于创建适用于 RDS Custom for SQL Server 的 CEV：

- 用于创建 CEV 的 AMI 必须基于 RDS Custom for SQL Server 支持的操作系统和数据库配置。有关支持的配置的更多信息，请参阅 Amazon RDS Custom for SQL Server 的要求和限制。
- CEV 必须具有唯一的名称。您无法创建与现有 CEV 同名的 CEV。
- 必须使用 SQL Server 的以下命名模式命名 CEV：主要版本 + 次要版本 + 自定义字符串。主要版本 + 次要版本必须与 AMI 提供的 SQL Server 版本相匹配。例如，您可以将带有 SQL Server 2019 CU17 的 AMI 命名为 15.00.4249.2.my_cevtest。

- 您必须使用 Sysprep 准备 AMI。有关使用 Sysprep 准备 AMI 的更多信息，请参阅使用 Sysprep 创建标准化的亚马逊机器映像（AMI）。
- 您负责维护 AMI 的生命周期。从 CEV 创建的 RDS Custom for SQL Server 数据库实例不存储 AMI 的副本。它保持一个指向您用于创建 CEV 的 AMI 的指针。AMI 必须存在，RDS Custom for SQL Server 数据库实例才能保持可运行状态。

RDS Custom for SQL Server CEV 的限制

以下限制适用于 RDS Custom for SQL Server 的自定义引擎版本：

- 如果存在与 CEV 关联的资源（如数据库实例或数据库快照），则无法将其删除。
- 要创建 RDS Custom for SQL Server 数据库实例，CEV 的状态必须为 `pending-validation`、`available`、`failed` 或 `validating`。如果 CEV 状态为 `incompatible-image-configuration`，则无法使用此 CEV 创建 RDS Custom for SQL Server 数据库实例。
- 要修改 RDS Custom for SQL Server 数据库实例以使用新的 CEV，CEV 的状态必须为 `available`。
- 无法从现有 RDS Custom for SQL Server 数据库实例创建 AMI 或 CEV。
- 您无法修改现有 CEV 以使用其他 AMI。但是，您可以修改 RDS Custom for SQL Server 数据库实例以使用不同的 CEV。有关更多信息，请参阅 修改 RDS Custom for SQL Server 数据库实例。
- 不支持使用与创建数据库实例时提供的 KMS 密钥不同的客户自主管理型 KMS 密钥来对 AMI 或 CEV 进行加密。
- 不支持 CEV 的跨区域副本。
- 不支持 CEV 的跨账户副本。
- 删除 CEV 后，无法还原或恢复它。但是，您可以从同一 AMI 创建新的 CEV。
- RDS Custom for SQL Server 数据库实例将您的 SQL Server 数据库文件存储在 D:\ 驱动器中。与 CEV 关联的 AMI 应将 Microsoft SQL Server 系统数据库文件存储在 C:\ 驱动器中。
- RDS Custom for SQL Server 数据库实例保留您对 SQL Server 所做的配置更改。在从 CEV 创建的正在运行的 RDS Custom for SQL Server 数据库实例上对操作系统所做的任何配置更改都不会保留。如果您需要对操作系统进行永久配置更改并将其保留为新的基准配置，请创建新的 CEV 并修改数据库实例以使用新的 CEV。

> **Important**
>
> 修改 RDS Custom for SQL Server 数据库实例以使用新的 CEV 是一项离线操作。您可以立即执行修改，也可以安排在每周维护时段内进行修改。

- 当您修改 CEV 时，Amazon RDS 不会将这些修改推送到任何关联的 RDS Custom for SQL Server 数据库实例。必须修改每个 RDS Custom for SQL Server 数据库实例，才能使用新的或更新的 CEV。有关更多信息，请参阅 [修改 RDS Custom for SQL Server 数据库实例]。
- > **Important**
  >
  > 如果删除 CEV 使用的 AMI，则任何可能需要更换主机的修改（例如规模计算）都将失败。然后，RDS Custom for SQL Server 数据库实例将放在 RDS 支持边界之外。建议您避免删除与 CEV 关联的任何 AMI。

创建适用于 RDS Custom for SQL Server 的 CEV

您可以使用 Amazon Web Services Management Console或 Amazon CLI 创建自定义引擎版本（CEV）。然后，您可以使用 CEV 创建 RDS Custom for SQL Server 实例。

确保 Amazon 机器映像（AMI）与您的 CEV 位于同一 Amazon 账户和区域中。否则，创建 CEV 的过程将失败。

有关更多信息，请参阅 [为 Amazon RDS Custom for SQL Server 创建并连接到数据库实例]。

> **Important**
>
> 对于使用预安装的 SQL Server 创建的 AMI 和使用自带媒体（BYOM）创建的 AMI，创建 CEV 的步骤相同。

控制台

要创建 CEV

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Custom engine versions（自定义引擎版本）。

Custom engine versions（自定义引擎版本）页面显示了当前存在的所有 CEV。如果您尚未创建任何 CEV，则该表为空。

3. 选择 Create custom engine version（创建自定义引擎版本）。
4. 对于 Engine type（引擎类型），选择 Microsoft SQL Server。
5. 对于版本，请选择要使用的数据库引擎版本。
6. 对于 Major version（主要版本），选择在 AMI 上安装的主引擎主版本。
7. 在 Version details（版本详细信息）中，在 Custom engine version name（自定义引擎版本名称）中输入有效的名称。

   名称格式为 *major-engine-version*.*minor-engine-version*.*customized_string*。您可以使用 1–50 位字母数字字符、下划线、短划线和句点。例如，您可以输入名称 **15.00.4249.2.my_cevtest**。

   或者，选择输入 CEV 的描述。
8. 对于 Installation Media（安装介质），浏览或输入您想要从中创建 CEV 的 AMI ID。
9. 在 Tags （标签）部分中，添加任何标签以标识 CEV。
10. 选择 Create custom engine version（创建自定义引擎版本）。

Custom engine versions（自定义引擎版本）页面显示。您的 CEV 显示状态 pending-validation（待验证）

Amazon CLI

要使用 Amazon CLI 创建 CEV，请运行 create-custom-db-engine-version 命令。

以下选项为必填：

- `--engine`
- `--engine-version`
- `--image-id`

您还可以指定以下选项：

- `--description`
- `--region`

- `--tags`

下面的示例创建了一个名为 `15.00.4249.2.my_cevtest` 的 CEV。确保 CEV 的名称以主要引擎版本号开头。

Example

对于 Linux、macOS 或 Unix：

```
aws rds create-custom-db-engine-version \
    --engine custom-sqlserver-ee \
    --engine-version 15.00.4249.2.my_cevtest \
    --image-id ami-0r93cx31t5r596482 \
    --description "Custom SQL Server EE 15.00.4249.2 cev test"
```

以下部分输出显示了引擎、参数组和其他信息。

```
"DBEngineVersions": [
    {
    "Engine": "custom-sqlserver-ee",
    "MajorEngineVersion": "15.00",
    "EngineVersion": "15.00.4249.2.my_cevtest",
    "DBEngineDescription": "Microsoft SQL Server Enterprise Edition for RDS Custom for
 SQL Server",
    "DBEngineVersionArn": "arn:aws:rds:us-east-1:<my-account-id>:cev:custom-sqlserver-
ee/15.00.4249.2.my_cevtest/a1234a1-123c-12rd-bre1-1234567890",
    "DBEngineVersionDescription": "Custom SQL Server EE 15.00.4249.2 cev test",

    "Image": [
        "ImageId": "ami-0r93cx31t5r596482",
        "Status": "pending-validation"
     ],
    "CreateTime": "2022-11-20T19:30:01.831000+00:00",
    "SupportsLogExportsToCloudwatchLogs": false,
    "SupportsReadReplica": false,
    "Status": "pending-validation",
    "SupportsParallelQuery": false,
    "SupportsGlobalDatabases": false,
    "TagList": []
    }
]
```

如果创建 CEV 的过程失败，则 RDS Custom for SQL Server 会发出 `RDS-EVENT-0198` 以及该消息 `Creation failed for custom engine version major-engine-version.cev_name`。该消息包括有关失败的详细信息，例如，事件打印丢失的文件。要查找有关 CEV 创建问题的故障排除建议，请参阅 RDS Custom for SQL Server 的 CEV 错误故障排除。

从 CEV 创建 RDS Custom for SQL Server 数据库实例

成功创建 CEV 后，CEV status（CEV 状态）将显示 `pending-validation`。现在，您可以使用 CEV 创建新的 RDS Custom for SQL Server 数据库实例。要从 CEV 创建新的 RDS Custom for SQL Server 数据库实例，请参阅 创建 RDS Custom for SQL Server 数据库实例。

CEV 的生命周期

CEV 生命周期包括以下状态。

CEV 状态	描述	故障排除建议
`pending-validation`	CEV 已创建，正在等待关联 AMI 的验证。CEV 将保持为 `pending-validation`，直到从中创建 RDS Custom for SQL Server 数据库实例。	如果没有现有任务，则从 CEV 创建一个新的 RDS Custom for SQL Server 数据库实例。创建 RDS Custom for SQL Server 数据库实例时，系统尝试验证 CEV 的关联 AMI。
`validating`	基于新的 CEV 创建 RDS Custom for SQL Server 数据库实例的任务正在进行中。创建 RDS Custom for SQL Server 数据库实例时，系统尝试	等待现有 RDS Custom for SQL Server 数据库实例的创建任务完成。您可以使用 RDS EVENTS 控制台查看详细的事件消息以进行故障排除。

CEV 状态	描述	故障排除建议
	验证 CEV 的关联 AMI。	
`available`	已成功验证 CEV。成功从 CEV 创建 RDS Custom for SQL Server 数据库实例后，CEV 将进入 `available` 状态。	CEV 不需要任何额外的验证。它可用于创建其他 RDS Custom for SQL Server 数据库实例或修改现有实例。
`inactive`	CEV 已修改为非活动状态。	您无法使用此 CEV 创建或升级 RDS Custom 数据库实例。此外，您无法使用此 CEV 还原数据库快照以创建新的 RDS Custom 数据库实例。有关如何将状态更改为 `ACTIVE` 的信息，请参阅 [修改适用于 RDS Custom for SQL Server 的 CEV]。
`failed`	此 CEV 的创建数据库实例步骤在可验证 AMI 之前失败。或者，CEV 使用的底层 AMI 不处于可用状态。	排查系统无法创建数据库实例的根本原因。查看详细的错误消息并尝试再次创建新的数据库实例。确保 CEV 使用的底层 AMI 处于可用状态。

CEV 状态	描述	故障排除建议
`incompatible-image-configuration`	验证 AMI 时出错。	查看错误的技术详细信息。您无法再次尝试使用此 CEV 验证 AMI。查看以下部分：建议： • 确保使用所需的 SQL Server 命名模式命名 CEV：主要版本 + 次要版本 + 自定义字符串。 • 确保 CEV 名称中的 SQL Server 版本与 AMI 提供的版本相匹配。 • 确保操作系统编译版本满足最低要求的编译版本。 • 确保操作系统主版本满足最低要求的主要版本。 使用正确的信息创建新的 CEV。 如果需要，使用支持的 AMI 创建一个新的 EC2 实例，并在其上运行 Sysprep 进程。

修改适用于 RDS Custom for SQL Server 的 CEV

您可以使用 Amazon Web Services Management Console 或 Amazon CLI 修改 CEV。您可以修改 CEV 描述或其可用性状态。您的 CEV 具有以下状态值之一：

- `available` – 您可以使用此 CEV 创建新的 RDS Custom 数据库实例或升级数据库实例。这是新创建的 CEV 的原定设置状态。
- `inactive` – 您无法使用此 CEV 创建或升级 RDS Custom 数据库实例。使用此 CEV，您无法还原数据库快照以创建新的 RDS Custom 数据库实例。

您可以将 CEV 状态从 `available` 更改为 `inactive` 或从 `inactive` 更改为 `available`。您可以将状态更改为 `INACTIVE`，以防止意外使用 CEV 或使已停止的 CEV 有资格再次使用。

控制台

要修改 CEV

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Custom engine versions（自定义引擎版本）。
3. 选择要修改其描述或状态的 CEV。
4. 对于 Actions（操作），选择 Modify（修改）。
5. 进行以下任何更改：
   - 对于 CEV status settings（CEV 状态设置），选择新的可用状态。
   - 对于 Version description（版本描述），输入新的描述。
6. 选择 Modify CEV（修改 CEV）。

   如果 CEV 正在使用中，控制台将显示 You can't modify the CEV status（您不能修改 CEV 状态）。修复问题，然后重试。

Custom engine versions（自定义引擎版本）页面显示。

Amazon CLI

要通过使用 Amazon CLI 修改 CEV，请运行 modify-custom-db-engine-version 命令。您可以通过运行 describe-db-engine-versions 命令查找要修改的 CEV。

以下选项为必填：

- `--engine`
- `--engine-version` *`cev`*，其中 *`cev`* 是要修改的自定义引擎版本的名称
- `--status` *`status`*，其中 *`status`* 是您希望分配给 CEV 的可用状态。

以下示例将名为 `15.00.4249.2.my_cevtest` 的 CEV 从当前状态更改为 `inactive`。

Example

对于 Linux、macOS 或 Unix：

```
aws rds modify-custom-db-engine-version \
```

```
    --engine custom-sqlserver-ee \
    --engine-version 15.00.4249.2.my_cevtest \
    --status inactive
```

对于 Windows：

```
aws rds modify-custom-db-engine-version ^
    --engine custom-sqlserver-ee ^
    --engine-version 15.00.4249.2.my_cevtest ^
    --status inactive
```

要修改 RDS Custom for SQL Server 数据库实例以使用新的 CEV，请参阅修改 RDS Custom for SQL Server 数据库实例以使用新 CEV。

修改 RDS Custom for SQL Server 数据库实例以使用新 CEV

您可以修改现有的 RDS Custom for SQL Server 数据库实例以使用不同的 CEV。您可以进行的更改包括：

- 更改 CEV
- 更改数据库实例类
- 更改备份保留期和备份时段
- 更改维护时段

控制台

要修改 RDS Custom for SQL Server 数据库实例

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。
3. 选择要修改的数据库实例。
4. 选择修改。
5. 根据需要进行以下更改：

    a. 对于 DB engine version（数据库引擎版本），请选择不同的 CEV。
    b. 更改 DB instance class（数据库实例类）的值。有关支持的类，请参阅 RDS Custom for SQL Server 的数据库实例类支持。

c. 更改 Backup retention period（备份保留期）的值。
d. 对于 Backup window（备份时段），设置 Start time（开始时间）和 Duration（持续时间）的值。
e. 对于 DB instance maintenance window（数据库实例维护时段），设置 Start day（开始日）、Start time（开始时间）和 Duration（持续时间）的值。

6. 选择继续。
7. 选择 Apply immediately（立即应用）或 Apply during the next scheduled maintenance window（在下一个计划的维护时段内应用）。
8. 选择修改数据库实例。

> **Note**
>
> 将数据库实例从一个 CEV 修改为另一个 CEV 时，例如，在升级次要版本时，SQL Server 系统数据库（包括其数据和配置）将在当前 RDS Custom for SQL Server 数据库实例中持久存在。

Amazon CLI

要使用 Amazon CLI 修改数据库实例以使用不同的 CEV，请运行 modify-db-instance 命令。

以下选项为必填：

- `--db-instance-identifier`
- `--engine-version` *`cev`*，其中 *`cev`* 是要将数据库实例更改为的自定义引擎版本的名称。

以下示例将名为 `my-cev-db-instance` 的数据库实例修改为使用名为 `15.00.4249.2.my_cevtest_new` 的 CEV 并立即应用更改。

Example

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier my-cev-db-instance \
    --engine-version 15.00.4249.2.my_cevtest_new \
    --apply-immediately
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier my-cev-db-instance ^
    --engine-version 15.00.4249.2.my_cevtest_new ^
    --apply-immediately
```

查看 Amazon RDS Custom for SQL Server 的 CEV 详细信息

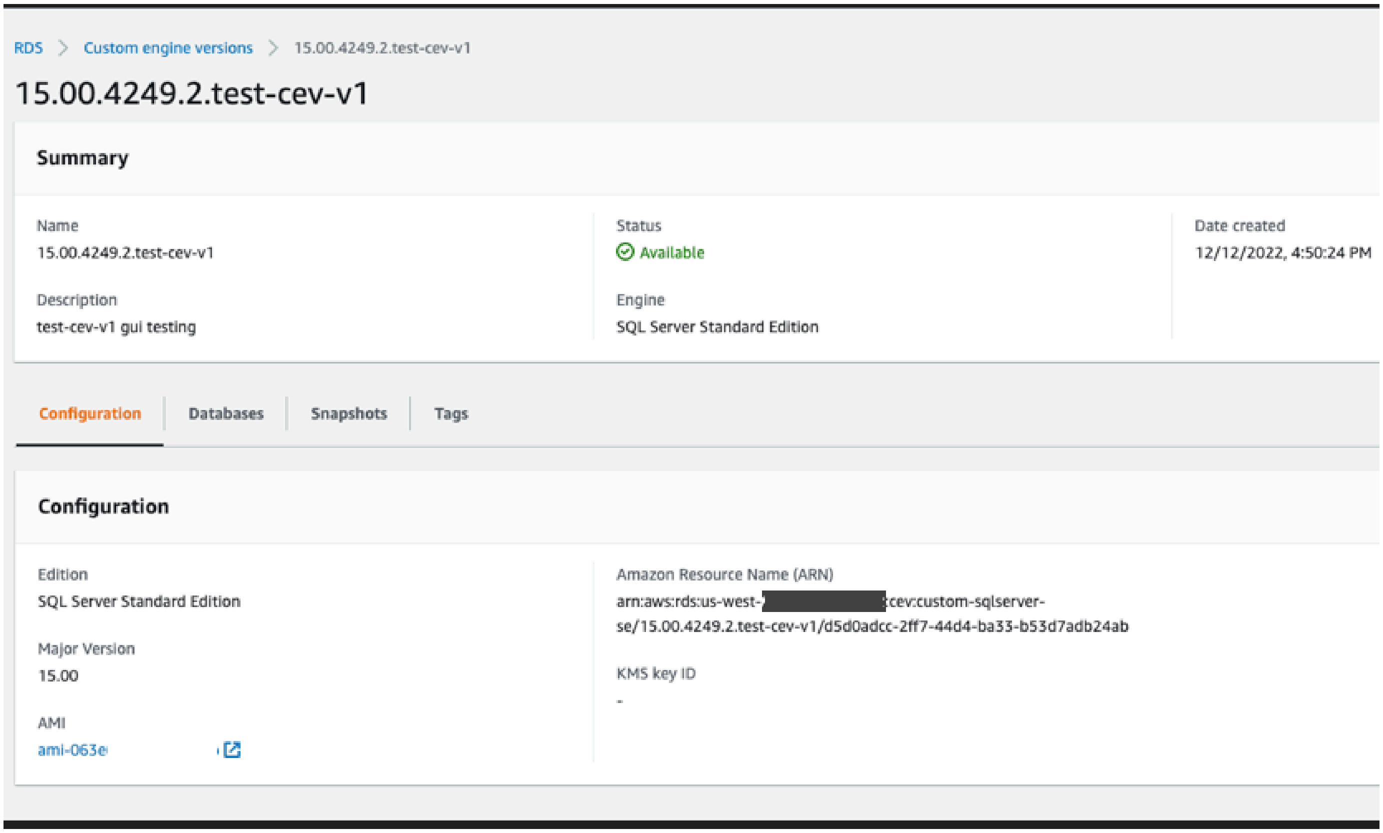
您可以使用 Amazon Web Services Management Console或 Amazon CLI 查看有关 CEV 的详细信息。

控制台

查看 CEV 详细信息

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Custom engine versions（自定义引擎版本）。

   Custom engine versions（自定义引擎版本）页面显示了当前存在的所有 CEV。如果您尚未创建任何 CEV，则该页面为空。
3. 选择要查看的 CEV 的名称。
4. 选择 Configuration（配置）以查看详细信息。



Amazon CLI

要通过使用 Amazon CLI 查看有关 CEV 的详细信息，请运行 describe-db-engine-versions 命令。

您还可以指定以下选项：

- `--include-all`，以查看具有任何生命周期状态的所有 CEV。如果没有 `--include-all` 选项，则只会返回处于 `available` 生命周期状态的 CEV。

```
aws rds describe-db-engine-versions --engine custom-sqlserver-ee --engine-version
 15.00.4249.2.my_cevtest --include-all
{
    "DBEngineVersions": [
        {
            "Engine": "custom-sqlserver-ee",
            "MajorEngineVersion": "15.00",
            "EngineVersion": "15.00.4249.2.my_cevtest",
            "DBParameterGroupFamily": "custom-sqlserver-ee-15.0",
            "DBEngineDescription": "Microsoft SQL Server Enterprise Edition for custom
 RDS",
```

```
            "DBEngineVersionArn": "arn:aws:rds:us-east-1:{my-account-id}:cev:custom-
sqlserver-ee/15.00.4249.2.my_cevtest/a1234a1-123c-12rd-bre1-1234567890",
            "DBEngineVersionDescription": "Custom SQL Server EE 15.00.4249.2 cev test",
            "Image": {
                "ImageId": "ami-0r93cx31t5r596482",
                "Status": "pending-validation"
            },
            "DBEngineMediaType": "AWS Provided",
            "CreateTime": "2022-11-20T19:30:01.831000+00:00",
            "ValidUpgradeTarget": [],
            "SupportsLogExportsToCloudwatchLogs": false,
            "SupportsReadReplica": false,
            "SupportedFeatureNames": [],
            "Status": "pending-validation",
            "SupportsParallelQuery": false,
            "SupportsGlobalDatabases": false,
            "TagList": [],
            "SupportsBabelfish": false
        }
    ]
}
```

您可以使用筛选条件查看具有特定生命周期状态的 CEV。例如，要查看生命周期状态为 `pending-validation`、`available` 或 `failed` 的 CEV：

```
aws rds describe-db-engine-versions engine custom-sqlserver-ee
                region us-west-2 include-all query 'DBEngineVersions[?Status ==
 pending-validation ||
                Status == available || Status == failed]'
```

删除适用于 RDS Custom for SQL Server 的 CEV

您可以使用 Amazon Web Services Management Console 或 Amazon CLI 删除 CEV。通常，这需要花费几分钟时间。

删除 CEV 之前，请确保它未被以下任何项目使用：

- RDS Custom 数据库实例
- RDS Custom 数据库实例的快照
- RDS Custom 数据库实例的自动备份

控制台

要删除 CEV

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Custom engine versions（自定义引擎版本）。
3. 选择要删除其描述或状态的 CEV。
4. 对于操作，选择删除。

   Delete ***cev_name***?（删除 cev_name？）对话框显示。
5. 输入 **delete me**，然后选择删除。

   在 Custom engine versions（自定义引擎版本）页面中，横幅显示您的 CEV 正在被删除。

Amazon CLI

要通过使用 Amazon CLI 删除 CEV，请运行 delete-custom-db-engine-version 命令。

以下选项为必填：

- `--engine custom-sqlserver-ee`
- `--engine-version cev`，其中 *cev* 是要删除的自定义引擎版本的名称

以下示例会删除名为 `15.00.4249.2.my_cevtest` 的 CEV。

Example

对于 Linux、macOS 或 Unix：

```
aws rds delete-custom-db-engine-version \
    --engine custom-sqlserver-ee \
    --engine-version 15.00.4249.2.my_cevtest
```

对于 Windows：

```
aws rds delete-custom-db-engine-version ^
    --engine custom-sqlserver-ee ^
    --engine-version 15.00.4249.2.my_cevtest
```

为 Amazon RDS Custom for SQL Server 创建并连接到数据库实例

您可以创建 RDS Custom 数据库实例，然后使用 Amazon Systems Manager 或远程桌面协议 (RDP) 连接该实例。

> **Important**
>
> 务必先完成 为 Amazon RDS Custom for SQL Server 设置环境 部分中的任务，然后才能创建或连接到 RDS Custom for SQL Server 数据库实例。
>
> 您可以在创建 RDS Custom 数据库实例时标记它们，但不要创建或修改 RDS Custom 自动化所需的 `AWSRDSCustom` 标签。有关更多信息，请参阅 为 RDS Custom for SQL Server 资源添加标签。
>
> 首次创建 RDS Custom for SQL Server 数据库实例时，您可能会收到以下错误：The service-linked role is in the process of being created.（服务相关角色正在创建过程中）。请稍后重试。如果您这样做，请等待几分钟，然后重新尝试创建数据库实例。

主题

- 创建 RDS Custom for SQL Server 数据库实例
- RDS Custom 服务相关角色
- 使用 Amazon Systems Manager 连接到 RDS Custom 数据库实例
- 使用 RDP 连接到 RDS Custom 数据库实例

创建 RDS Custom for SQL Server 数据库实例

使用 Amazon Web Services Management Console 或 Amazon CLI 创建 Amazon RDS Custom for SQL Server 数据库实例。该过程与创建 Amazon RDS 数据库实例的过程相似。

有关更多信息，请参阅 创建 Amazon RDS 数据库实例。

控制台

要创建 RDS Custom for SQL Server 数据库实例

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases（数据库）。

3. 选择创建数据库。
4. 选择Standard create (标准创建) 作为数据库创建方法。
5. 在 Engine options（引擎选项）中，为引擎版本选择 Microsoft SQL Server。
6. 对于 Database management type（数据库管理类型），选择 Amazon RDS Custom。
7. 在 Edition（版本）部分中，选择要使用的数据库引擎版本。
8. （可选）如果您打算从 CEV 创建数据库实例，请选中 Use custom engine version (CEV) [使用自定义引擎版本（CEV）] 复选框。在下拉列表中选择您的 CEV。
9. 对于数据库版本，保留默认值版本。
10. 对于 Templates (模板)，选择 Production (生产)。
11. 在 Settings（设置）部分中，为 DB instance identifier（数据库实例标识符）输入一个唯一的名称。
12. 要输入主密码，请执行以下操作：

    a. 在设置部分中，打开凭证设置。
    b. 清除自动生成密码复选框。
    c. 更改 Master username（主用户名）值，并在 Master password（主密码）和 Confirm password（确认密码）中输入相同的密码。

    预设情况下，新的 RDS Custom 数据库实例对主用户使用自动生成的密码。
13. 在 DB instance size（数据库实例大小）部分中，为 DB instance class（数据库实例类）选择一个值。

    有关支持的类，请参阅 RDS Custom for SQL Server 的数据库实例类支持。
14. 选择 Storage（存储）设置。
15. 对于 RDS Custom 安全性，执行以下操作：

    a. 对于 IAM 实例配置文件，可以使用两个选项来为 RDS Custom for SQL Server 数据库实例选择实例配置文件。

        1. 选择创建新的实例配置文件并提供实例配置文件的名称后缀。有关更多信息，请参阅 使用 Amazon Web Services Management Console自动创建实例配置文件。
        2. 选择现有的实例配置文件。从下拉列表中，选择以 `AWSRDSCustom` 开头的实例配置文件。
    b. 对于加密，选择 Enter a key ARN（输入一个密钥 ARN）以列出可用的 Amazon KMS 密钥。然后从列表中选择您的密钥。

RDS Custom 需要 Amazon KMS 密钥。有关更多信息，请参阅 请确保您有对称的加密 Amazon KMS 密钥。

16. 对于其余部分，请指定首选的 RDS Custom 数据库实例设置。有关每项设置的信息，请参阅 数据库实例的设置。以下设置不会显示在控制台中，也不受支持：

    - 处理器功能
    - Storage autoscaling (存储自动扩展)
    - 可用性与持久性
    - Database authentication（数据库身份验证）中的 Password and Kerberos authentication（密码和 Kerberos 身份验证）选项（仅支持 Password authentication（密码身份验证））
    - Additional configuration（其他配置）中的 Database options（数据库选项）组
    - Performance Insights
    - 日志导出
    - 启用自动次要版本升级
    - 删除保护

    Backup retention period（备份保留期）受支持，但您不能选择 0 天。

17. 选择 Create database（创建数据库）。

    View credential details（查看凭证详细信息）按钮显示在 Databases（数据库）页面中。

    要查看 RDS Custom 数据库实例的主用户名和密码，请选择 View credential details（查看凭证详细信息）。

    要以主用户身份连接到数据库实例，请使用显示的用户名和密码。

    > **Important**
    >
    > 您无法再次查看主用户密码。如果您不记录它，您可能需要更改它。要在 RDS Custom 数据库实例可用后更改主用户密码，请修改数据库实例。有关修改数据库实例的更多信息，请参阅 管理 Amazon RDS Custom for SQL Server 数据库实例。

18. 选择 Databases（数据库）以查看 RDS Custom 数据库实例列表。

19. 选择您刚创建的 RDS Custom 数据库实例。

在 RDS 控制台上，将显示新 RDS Custom 数据库实例的详细信息。

- 数据库实例具有 creating（正在创建）状态，直到 RDS Custom 数据库实例完成创建并可供使用。当状态变为 Available (可用) 时，您可以连接到该数据库实例。根据所分配的实例类和存储的不同，新数据库实例可能需要数分钟时间才能变得可用。
- Role（角色）具有值 Instance (RDS Custom)（实例 (RDS Custom)）。
- RDS Custom automation mode（RDS Custom 自动化模式）具有值 Full automation（完全自动化）。此设置意味着数据库实例可以提供自动监控和实例恢复。

Amazon CLI

您可以通过使用 [create-db-instance](https://docs.aws.amazon.com/cli/latest/reference/rds/create-db-instance.html) Amazon CLI 命令创建 RDS Custom 数据库实例。

以下选项为必填：

- `--db-instance-identifier`
- `--db-instance-class`（有关支持的实例类的列表，请参阅 RDS Custom for SQL Server 的数据库实例类支持）
- `--engine`（`custom-sqlserver-ee`、`custom-sqlserver-se`、或 `custom-sqlserver-web`）
- `--kms-key-id`
- `--custom-iam-instance-profile`

以下示例将创建名为 `my-custom-instance` 的 RDS Custom for SQL Server 数据库实例。备份保留期为 3 天。

> **Note**
>
> 要从自定义引擎版本（CEV）创建数据库实例，请为 `--engine-version` 参数提供现有 CEV 名称。例如，`--engine-version 15.00.4249.2.my_cevtest`

Example

对于 Linux、macOS 或 Unix：

```
aws rds create-db-instance \
```

```
    --engine custom-sqlserver-ee \
    --engine-version 15.00.4073.23.v1 \
    --db-instance-identifier my-custom-instance \
    --db-instance-class db.m5.xlarge \
    --allocated-storage 20 \
    --db-subnet-group mydbsubnetgroup \
    --master-username myuser \
    --master-user-password mypassword \
    --backup-retention-period 3 \
    --no-multi-az \
    --port 8200 \
    --kms-key-id mykmskey \
    --custom-iam-instance-profile AWSRDSCustomInstanceProfileForRdsCustomInstance
```

对于 Windows：

```
aws rds create-db-instance ^
    --engine custom-sqlserver-ee ^
    --engine-version 15.00.4073.23.v1 ^
    --db-instance-identifier my-custom-instance ^
    --db-instance-class db.m5.xlarge ^
    --allocated-storage 20 ^
    --db-subnet-group mydbsubnetgroup ^
    --master-username myuser ^
    --master-user-password mypassword ^
    --backup-retention-period 3 ^
    --no-multi-az ^
    --port 8200 ^
    --kms-key-id mykmskey ^
    --custom-iam-instance-profile AWSRDSCustomInstanceProfileForRdsCustomInstance
```

Note

作为安全最佳实践，请指定除此处所示提示以外的密码。

使用 `describe-db-instances` 命令以获取有关您的实例的详细信息。

```
aws rds describe-db-instances --db-instance-identifier my-custom-instance
```

以下部分输出显示了引擎、参数组和其他信息。

```
{
    "DBInstances": [
        {
            "PendingModifiedValues": {},
            "Engine": "custom-sqlserver-ee",
            "MultiAZ": false,
            "DBSecurityGroups": [],
            "DBParameterGroups": [
                {
                    "DBParameterGroupName": "default.custom-sqlserver-ee-15",
                    "ParameterApplyStatus": "in-sync"
                }
            ],
            "AutomationMode": "full",
            "DBInstanceIdentifier": "my-custom-instance",
            "TagList": []
        }
    ]
}
```

RDS Custom 服务相关角色

服务相关角色 为 Amazon RDS Custom 提供针对您的 Amazon Web Services 账户中的资源的访问权限。这样可以让您更轻松地使用 RDS Custom，因为您不必手动添加必要的权限。RDS Custom 定义其服务相关角色的权限，除非另外定义，否则只有 RDS Custom 可以代入该角色。定义的权限包括信任策略和权限策略，以及不能附加到任何其他 IAM 实体的权限策略。

创建 RDS Custom 数据库实例时，将创建并使用 Amazon RDS 和 RDS Custom 服务相关角色（如果它们尚未存在）。有关更多信息，请参阅 将服务相关角色用于 Amazon RDS。

首次创建 RDS Custom for SQL Server 数据库实例时，您可能会收到以下错误：The service-linked role is in the process of being created.（服务相关角色正在创建过程中）。请稍后重试。如果您这样做，请等待几分钟，然后重新尝试创建数据库实例。

使用 Amazon Systems Manager 连接到 RDS Custom 数据库实例

创建 RDS Custom 数据库实例后，您可以使用 Amazon Systems Manager 会话管理器。会话管理器是一项 Systems Manager 功能，可让您通过基于浏览器的 shell 或通过 Amazon CLI 管理 Amazon EC2 实例。有关更多信息，请参阅 Amazon Systems Manager 会话管理器。

控制台

使用会话管理器连接到您的数据库实例

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，请选择 Databases（数据库），然后选择要连接到的 RDS Custom 数据库实例。
3. 选择配置。
4. 请注意您的数据库实例的 Resource ID（资源 ID）值。例如，资源 ID 可能为 `db-ABCDEFGHIJKLMNOPQRS0123456`。
5. 通过以下网址打开 Amazon EC2 控制台：https://console.aws.amazon.com/ec2/。
6. 在导航窗格中，选择实例。
7. 查找 EC2 实例的名称，然后选择与其关联的实例 ID。例如，实例 ID 可能为 `i-abcdefghijklm01234`。
8. 选择连接。
9. 选择 Session Manager（会话管理器）。
10. 选择连接。

    系统将为您的会话打开一个窗口。

Amazon CLI

您可以使用 Amazon CLI 连接到 RDS Custom 数据库实例。此方法需要 Amazon CLI 的会话管理器插件。要了解如何安装插件，请参阅安装 Amazon CLI 的会话管理器插件。

要查找 RDS Custom 数据库实例的数据库资源 ID，请使用 `describe-db-instances`。

```
aws rds describe-db-instances \
    --query 'DBInstances[*].[DBInstanceIdentifier,DbiResourceId]' \
    --output text
```

以下示例输出显示了 RDS Custom 实例的资源 ID。前缀为 `db-`。

```
db-ABCDEFGHIJKLMNOPQRS0123456
```

要查找数据库实例的 EC2 实例 ID，请使用 `aws ec2 describe-instances`。以下示例将 `db-ABCDEFGHIJKLMNOPQRS0123456` 用于资源 ID。

```
aws ec2 describe-instances \
    --filters "Name=tag:Name,Values=db-ABCDEFGHIJKLMNOPQRS0123456" \
    --output text \
    --query 'Reservations[*].Instances[*].InstanceId'
```

以下示例输出显示了 EC2 实例 ID。

```
i-abcdefghijklm01234
```

使用 `aws ssm start-session` 命令，以在 `--target` 参数中提供 EC2 实例 ID。

```
aws ssm start-session --target "i-abcdefghijklm01234"
```

成功的连接如下所示。

```
Starting session with SessionId: yourid-abcdefghijklm1234
[ssm-user@ip-123-45-67-89 bin]$
```

使用 RDP 连接到 RDS Custom 数据库实例

创建 RDS Custom 数据库实例后，您可以使用 RDP 客户端连接到此实例。该过程与连接到 Amazon EC2 实例的过程相同。有关更多信息，请参阅连接到 Windows 实例。

要连接到数据库实例，您需要与该实例关联的密钥对。RDS Custom 会为您创建密钥对。密钥对名称使用前缀 `do-not-delete-rds-custom-DBInstanceIdentifier`。Amazon Secrets Manager 将私有密钥存储为密钥。

完成以下步骤中的任务：

1. 配置您的数据库实例以允许 RDP 连接.
2. 检索您的私有密钥.
3. 使用 RDP 实用工具连接到 EC2 实例.

配置您的数据库实例以允许 RDP 连接

要允许 RDP 连接，请配置 VPC 安全组并在主机上设置防火墙规则。

配置您的 VPC 安全组

请确保与您数据库实例关联的 VPC 安全组允许在端口 3389 上进行传输控制协议 (TCP) 的入站连接。要了解如何配置 VPC 安全组，请参阅 配置您的 VPC 安全组。

在主机上设置防火墙规则

要允许端口 3389 上的 TCP 入站连接，请在主机上设置防火墙规则。以下示例显示如何执行此操作。

建议使用特定的 `-Profile` 值：`Public`、`Private` 或者 `Domain`。使用 `Any` 指这三个值。您还可以指定用逗号分隔的一组值。有关设置防火墙规则的更多信息，请参阅 Microsoft 文档中的 Set-NetFirewallRule。

使用 Systems Manager 会话管理器设置防火墙规则

1. 连接到会话管理器，如 使用 Amazon Systems Manager 连接到 RDS Custom 数据库实例 中所示。
2. 运行以下命令。

```
Set-NetFirewallRule -DisplayName "Remote Desktop - User Mode (TCP-In)" -Direction
 Inbound -LocalAddress Any -Profile Any
```

使用 Systems Manager CLI 命令设置防火墙规则

1. 使用以下命令在主机上打开 RDP。

```
OPEN_RDP_COMMAND_ID=$(aws ssm send-command --region $AWS_REGION \
    --instance-ids $RDS_CUSTOM_INSTANCE_EC2_ID \
    --document-name "AWS-RunPowerShellScript" \
    --parameters '{"commands":["Set-NetFirewallRule -DisplayName \"Remote Desktop -
 User Mode (TCP-In)\" -Direction Inbound -LocalAddress Any -Profile Any"]}' \
    --comment "Open RDP port" | jq -r ".Command.CommandId")
```

2. 使用输出中返回的命令 ID 获取上一个命令的状态。要使用以下查询返回命令 ID，请确保安装了 jq 插件。

```
aws ssm list-commands \
    --region $AWS_REGION \
    --command-id $OPEN_RDP_COMMAND_ID
```

检索您的私有密钥

使用 Amazon Web Services Management Console 或 Amazon CLI 检索您的私有密钥。

控制台

要检索私有密钥

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，请选择 Databases（数据库），然后选择要连接到的 RDS Custom 数据库实例。
3. 选择配置选项卡。
4. 请注意您的数据库实例的 DB instance ID（数据库实例 ID），例如 `my-custom-instance`。
5. 通过以下网址打开 Amazon EC2 控制台：https://console.aws.amazon.com/ec2/。
6. 在导航窗格中，选择实例。
7. 查找 EC2 实例的名称，然后选择与其关联的实例 ID。

   在此示例中，实例 ID 为 `i-abcdefghijklm01234`。
8. 在 Details（详细信息）中，查找 Key pair name（密钥对名称）。密钥对名称包括数据库标识符。在此示例中，密钥对名称为 `do-not-delete-rds-custom-my-custom-instance-0d726c`。
9. 在实例摘要中，查找 Public IPv4 DNS（公有 IPv4 DNS）。在本示例中，公有 DNS 可能是 `ec2-12-345-678-901.us-east-2.compute.amazonaws.com`。
10. 打开位于 https://console.aws.amazon.com/secretsmanager/ 的 Amazon Secrets Manager 控制台。
11. 选择与密钥对同名的密钥。
12. 选择 Retrieve secret value (检索密钥值)。

Amazon CLI

要检索私有密钥

1. 通过调用 `aws rds describe-db-instances` 命令获取您的 RDS Custom 数据库实例的列表。

```
aws rds describe-db-instances \
    --query 'DBInstances[*].[DBInstanceIdentifier,DbiResourceId]' \
```

```
    --output text
```

2. 例如，从示例输出中选择数据库实例标识符，例如 do-not-delete-rds-custom-*my-custom-instance*。
3. 通过调用 aws ec2 describe-instances 命令查找您的数据库实例的 EC2 实例 ID。以下示例使用 EC2 实例名称来描述数据库实例。

```
aws ec2 describe-instances \
    --filters "Name=tag:Name,Values=do-not-delete-rds-custom-my-custom-instance" \
    --output text \
    --query 'Reservations[*].Instances[*].InstanceId'
```

以下示例输出显示了 EC2 实例 ID。

```
i-abcdefghijklm01234
```

4. 通过指定 EC2 实例 ID 查找密钥名称，如以下示例所示。

```
aws ec2 describe-instances \
    --instance-ids i-abcdefghijklm01234 \
    --output text \
    --query 'Reservations[*].Instances[*].KeyName'
```

以下示例输出显示了使用前缀 do-not-delete-rds-custom-*DBInstanceIdentifier* 的密钥名称。

```
do-not-delete-rds-custom-my-custom-instance-0d726c
```

使用 RDP 实用工具连接到 EC2 实例

执行《Amazon EC2 用户指南》中的使用 RDP 连接到 Windows 实例中的过程。此过程假定您创建了包含您的私有密钥的 .pem 文件。

管理 Amazon RDS Custom for SQL Server 数据库实例

Amazon RDS Custom for SQL Server 支持 Amazon RDS 数据库实例的常见管理任务子集。在下文中，您可以找到使用 Amazon Web Services Management Console和 Amazon CLI 的受支持的 RDS Custom for SQL Server 管理任务的说明。

主题

暂停和恢复 RDS Custom 自动化

RDS Custom 自动为 RDS Custom for SQL Server 数据库实例提供监控和实例恢复功能。如果您需要自定义实例，请执行以下操作：

1. 在指定的时间内暂停 RDS Custom 自动化。暂停可确保您的自定义项不会干扰 RDS Custom 自动化。
2. 根据需要自定义 RDS Custom for SQL Server 数据库实例。
3. 请执行以下任一操作：
   - 手动恢复自动化。
   - 等待暂停期结束。在这种情况下，RDS Custom 会自动恢复监控和实例恢复。

> **Important**
>
> 修改 RDS Custom for SQL Server 数据库实例时，暂停和恢复自动化是唯一受支持的自动化任务。

控制台

要暂停或恢复 RDS Custom 自动化

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，请选择 Databases（数据库），然后选择要修改的 RDS Custom 数据库实例。
3. 选择修改。将显示修改数据库实例页面。
4. 对于 RDS Custom automation mode（RDS Custom 自动化模式），请选择以下选项之一：

   - Paused（已暂停）将暂停 RDS Custom 数据库实例的监控和实例恢复。为 Automation mode duration（自动化模式持续时间）输入您想要的暂停持续时间。最小值为 60 分钟（默认值）。最大值为 1440 分钟。
   - Full automation（完全自动化）将恢复自动化。
5. 请选择 Continue（继续），以查看修改摘要。

   将显示一条消息，表示 RDS Custom 将立即应用更改。
6. 如果更改正确无误，请选择 Modify DB Instance（修改数据库实例）。也可以选择 Back (返回) 编辑您的更改，或选择 Cancel (取消) 取消更改。

   在 RDS 控制台上，将显示修改的详细信息。如果您暂停了自动化，RDS Custom 数据库实例的 Status（状态）将指示 Automation paused（自动化已暂停）。
7. （可选）在导航窗格中，选择 Databases (数据库)，然后选择 RDS Custom 数据库实例。

   在 Summary（摘要）窗格中，RDS Custom automation mode（RDS Custom 自动化模式）表示自动化状态。如果自动化已暂停，则值为 Paused（已暂停）。自动化将在 ***num*** 分钟内恢复。

Amazon CLI

要暂停或恢复 RDS Custom 自动化，请使用 `modify-db-instance` Amazon CLI 命令。使用必需的参数 `--db-instance-identifier` 识别数据库实例。使用以下参数控制自动化模式：

- `--automation-mode` 指定数据库实例的暂停状态。有效值为 `all-paused`，它将暂停自动化，以及 `full`，它将恢复自动化。
- `--resume-full-automation-mode-minutes` 指定暂停的持续时间。默认值为 60 分钟。

> **Note**
>
> 无论您是指定 `--no-apply-immediately` 还是 `--apply-immediately`，RDS Custom 都会尽快地异步应用修改。

在命令响应中，`ResumeFullAutomationModeTime` 以 UTC 时间戳表示恢复时间。当自动化模式为 `all-paused` 时，您可以使用 `modify-db-instance` 恢复自动化模式或延长暂停时间。不支持其他 `modify-db-instance` 选项。

以下示例将 `my-custom-instance` 的自动化暂停 90 分钟。

Example

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier my-custom-instance \
    --automation-mode all-paused \
    --resume-full-automation-mode-minutes 90
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier my-custom-instance ^
    --automation-mode all-paused ^
    --resume-full-automation-mode-minutes 90
```

以下示例将暂停持续时间再延长 30 分钟。在 `ResumeFullAutomationModeTime` 中显示的原始时间中增加 30 分钟。

Example

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier my-custom-instance \
    --automation-mode all-paused \
    --resume-full-automation-mode-minutes 30
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier my-custom-instance ^
    --automation-mode all-paused ^
    --resume-full-automation-mode-minutes 30
```

以下示例恢复 `my-custom-instance` 的完全自动化。

Example

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier my-custom-instance \
    --automation-mode full \
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier my-custom-instance ^
    --automation-mode full
```

在以下部分示例输出中，待处理 `AutomationMode` 值为 `full`。

```
{
    "DBInstance": {
        "PubliclyAccessible": true,
        "MasterUsername": "admin",
        "MonitoringInterval": 0,
        "LicenseModel": "bring-your-own-license",
        "VpcSecurityGroups": [
            {
                "Status": "active",
                "VpcSecurityGroupId": "0123456789abcdefg"
            }
        ],
        "InstanceCreateTime": "2020-11-07T19:50:06.193Z",
        "CopyTagsToSnapshot": false,
        "OptionGroupMemberships": [
            {
                "Status": "in-sync",
                "OptionGroupName": "default:custom-oracle-ee-19"
            }
```

```
        ],
        "PendingModifiedValues": {
            "AutomationMode": "full"
        },
        "Engine": "custom-oracle-ee",
        "MultiAZ": false,
        "DBSecurityGroups": [],
        "DBParameterGroups": [
            {
                "DBParameterGroupName": "default.custom-oracle-ee-19",
                "ParameterApplyStatus": "in-sync"
            }
        ],
        ...
        "ReadReplicaDBInstanceIdentifiers": [],
        "AllocatedStorage": 250,
        "DBInstanceArn": "arn:aws:rds:us-west-2:012345678912:db:my-custom-instance",
        "BackupRetentionPeriod": 3,
        "DBName": "ORCL",
        "PreferredMaintenanceWindow": "fri:10:56-fri:11:26",
        "Endpoint": {
            "HostedZoneId": "ABCDEFGHIJKLMNO",
            "Port": 8200,
            "Address": "my-custom-instance.abcdefghijk.us-west-2.rds.amazonaws.com"
        },
        "DBInstanceStatus": "automation-paused",
        "IAMDatabaseAuthenticationEnabled": false,
        "AutomationMode": "all-paused",
        "EngineVersion": "19.my_cev1",
        "DeletionProtection": false,
        "AvailabilityZone": "us-west-2a",
        "DomainMemberships": [],
        "StorageType": "gp2",
        "DbiResourceId": "db-ABCDEFGHIJKLMNOPQRSTUVW",
        "ResumeFullAutomationModeTime": "2020-11-07T20:56:50.565Z",
        "KmsKeyId": "arn:aws:kms:us-west-2:012345678912:key/
aa111a11-111a-11a1-1a11-1111a11a1a1a",
        "StorageEncrypted": false,
        "AssociatedRoles": [],
        "DBInstanceClass": "db.m5.xlarge",
        "DbInstancePort": 0,
        "DBInstanceIdentifier": "my-custom-instance",
        "TagList": []
```

```
    }
```

修改 RDS Custom for SQL Server 数据库实例

修改 RDS Custom for SQL Server 数据库实例的过程与为 Amazon RDS 执行此操作类似，但您可以进行的更改仅限于以下内容：

- 更改数据库实例类
- 更改备份保留期和备份时段
- 更改维护时段
- 当新版本可用时升级数据库引擎版本
- 更改分配的存储、预调配 IOPS 和存储类型
- 允许和删除多可用区部署

以下限制适用于修改 RDS Custom for SQL Server 数据库实例：

- 不支持自定义数据库选项和参数组。
- 手动附加到 RDS Custom 数据库实例的任何存储卷都在支持外围之外。

  有关更多信息，请参阅 RDS Custom 支持外围。

控制台

要修改 RDS Custom for SQL Server 数据库实例

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。
3. 选择要修改的数据库实例。
4. 选择修改。
5. 根据需要进行以下更改：

   a. 对于数据库引擎版本，选择新版本。
   b. 更改 DB instance class（数据库实例类）的值。有关支持的类，请参阅RDS Custom for SQL Server 的数据库实例类支持

c. 更改 Backup retention period（备份保留期）的值。

d. 对于 Backup window（备份时段），设置 Start time（开始时间）和 Duration（持续时间）的值。

e. 对于 DB instance maintenance window（数据库实例维护时段），设置 Start day（开始日）、Start time（开始时间）和 Duration（持续时间）的值。

6. 选择继续。

7. 选择 Apply immediately（立即应用）或 Apply during the next scheduled maintenance window（在下一个计划的维护时段内应用）。

8. 选择修改数据库实例。

Amazon CLI

要修改 RDS Custom for SQL Server 数据库实例，请使用 modify-db-instance Amazon CLI 命令。根据需要设置以下参数：

- `--db-instance-class` – 有关支持的类，请参阅RDS Custom for SQL Server 的数据库实例类支持
- `--engine-version` – 要升级到的数据库引擎的版本号。
- `--backup-retention-period` – 保留自动备份的持续时间，为 0-35 天。
- `--preferred-backup-window` – 创建自动备份的每日时间范围。
- `--preferred-maintenance-window` – 可能进行系统维护的每周时间范围（按世界协调时间计算）。
- `--apply-immediately` – 使用 `--apply-immediately` 可立即应用存储更改。

  或使用 `--no-apply-immediately`（默认设置）以在下一个维护时段内应用更改。

修改 RDS Custom for SQL Server 数据库实例的存储

修改 RDS Custom for SQL Server 数据库实例的存储与修改 Amazon RDS 数据库实例的存储类似，但您只能执行以下操作：

- 增加分配的存储大小。
- 更改存储类型。您可以使用可用的存储类型，如通用型或预调配 IOPS。gp3、io1 和 io2 Block Express 存储类型支持预调配 IOPS。
- 如果您使用的是支持预调配 IOPS 的卷类型，则更改预调配 IOPS。

以下限制适用于修改 RDS Custom for SQL Server 数据库实例的存储：

- RDS Custom for SQL Server 的最小分配存储大小为 20 GiB。io1、gp2 和 gp3 的最大存储限制为 16 TiB，而 io2 支持 64 TiB。
- 与 Amazon RDS 一样，您无法减少分配的存储。这是 Amazon Elastic Block Store (Amazon EBS) 卷的限制。有关更多信息，请参阅 使用 Amazon RDS 数据库实例的存储。
- RDS Custom for SQL Server 数据库实例不支持存储自动扩缩。
- 您手动连接到 RDS Custom 数据库实例的任何存储卷都不会考虑进行存储扩缩。仅考虑对 RDS 提供的默认数据卷（即 D 驱动器）进行存储扩缩。

  有关更多信息，请参阅 RDS Custom 支持外围。
- 扩展存储通常不会导致数据库实例的任何中断或性能下降。在修改数据库实例的存储大小后，数据库实例的状态为 storage-optimization。
- 存储优化可能耗时数小时。六（6）小时后或在实例上完成存储优化后，您才能进行进一步的存储修改。有关更多信息，请参阅 使用 Amazon RDS 数据库实例的存储。
- 只能通过 Amazon RDS 控制台或 Amazon CLI 修改存储。直接从 Amazon EC2、Amazon EBS 控制台或 CLI 修改存储可能会导致预期存储设置与实际存储设置不一致。对于多可用区实例，这可能会使两台主机上的卷大小不一致，从而导致 RDS Custom 的配置状态不受支持。

有关存储的更多信息，请参阅 Amazon RDS 数据库实例存储。

有关存储修改的一般信息，请参阅使用 Amazon RDS 数据库实例的存储。

控制台

修改 RDS Custom for SQL Server 数据库实例的存储

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。
3. 选择要修改的数据库实例。
4. 选择修改。
5. 根据需要进行以下更改：

   a. 为 Allocated Storage (分配的存储) 输入新值。它必须大于当前值，且范围从 20GiB 到 16TiB。

    b. 更改 Storage type（存储类型）的值。您可以从可用的通用型或预调配 IOPS 存储类型中进行选择。gp3、io1 和 io2 Block Express 存储类型支持预调配 IOPS。
    c. 如果您指定支持预调配 IOPS 的卷类型，则可以定义预调配 IOPS 值。

6. 选择继续。
7. 选择 Apply immediately（立即应用）或 Apply during the next scheduled maintenance window（在下一个计划的维护时段内应用）。
8. 选择修改数据库实例。

Amazon CLI

要修改 RDS Custom for SQL Server 数据库实例的存储，请使用 [modify-db-instance](https://docs.aws.amazon.com/cli/latest/reference/rds/modify-db-instance.html) Amazon CLI 命令。根据需要设置以下参数：

- `--allocated-storage` – 要为数据库实例分配的存储量（以 GiB 为单位）。它必须大于当前值，并且范围从 20 到 16,384GiB。
- `--storage-type` – 存储类型，例如 gp2、gp3、io1 或 io2。
- `--iops` – 数据库实例的预调配 IOPS。您只能为支持预调配 IOPS 的存储类型（gp3、io1 和 io2）指定此项。
- `--apply-immediately` – 使用 `--apply-immediately` 可立即应用存储更改。

  或使用 `--no-apply-immediately`（默认设置）以在下一个维护时段内应用更改。

以下示例将 my-custom-instance 的存储大小更改为 200GiB，将存储类型更改为 io1，将预调配 IOPS 更改为 3000。

Example

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier my-custom-instance \
    --storage-type io1 \
    --iops 3000 \
    --allocated-storage 200 \
    --apply-immediately
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier my-custom-instance ^
    --storage-type io1 ^
    --iops 3000 ^
    --allocated-storage 200 ^
    --apply-immediately
```

为 RDS Custom for SQL Server 资源添加标签

您可以将 RDS Custom 资源与 Amazon RDS 资源一样标记，但有一些重要的区别：

- 不要创建或修改 RDS Custom 自动化所需的 `AWSRDSCustom` 标签。如果您这样做，可能会破坏自动化。
- Name 标签将添加到 RDS Custom 资源，前缀为 `do-not-delete-rds-custom`。任何客户传递的键值都将被覆盖。
- 在创建过程中添加到 RDS Custom 数据库实例的标签将传播到所有其他相关 RDS Custom 资源。
- 当您在创建数据库实例后将标签添加到 RDS Custom 资源时，标签不会传播。

有关资源标记的一般信息，请参阅为 Amazon RDS 资源添加标签。

删除 RDS Custom for SQL Server 数据库实例

要删除 RDS Custom for SQL Server 数据库实例，请执行以下操作：

- 提供数据库实例的名称。
- 选择或清除拍摄数据库实例的最终数据库快照的选项。
- 选择或清除保留自动备份的选项。

您可以使用控制台或 CLI 删除 RDS Custom for SQL Server 数据库实例。删除数据库实例所需的时间可能会有所不同，具体取决于备份保留期（也即，要删除的备份数）、删除的数据量以及是否拍摄最终快照。

> **Warning**
>
> 删除 RDS Custom for SQL Server 数据库实例将永久删除 EC2 实例和关联的 Amazon EBS 卷。任何时候都不要终止或删除这些资源，否则，删除和最终快照创建可能会失败。

> **Note**
>
> 如果数据库实例具有 `creating`、`failed`、`incompatible-create`、`incompatible-restore` 或 `incompatible-network` 状态，则无法为其创建最终数据库快照。有关更多信息，请参阅 查看Amazon RDS 数据库实例状态。

> **Important**
>
> 当您选择拍摄最终快照时，我们建议您当正在删除数据库实例时避免向数据库实例写入数据。启动数据库实例删除后，不能保证数据更改会被最终快照捕获。

控制台

要删除 RDS Custom 数据库实例

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases（数据库），然后选择要删除的 RDS Custom for SQL Server 数据库实例。RDS Custom for SQL Server 数据库实例显示角色 Instance (RDS Custom for SQL Server) [实例（RDS Custom for SQL Server）]。
3. 对于操作，选择删除。
4. 要拍摄最终快照，请选择 Create final snapshot（创建最终快照），并为 Final snapshot name（最终快照名称）提供名称。
5. 要保留自动备份，请选择 Retain automated backups (保留自动备份)。
6. 在框中输入 **`delete me`**。
7. 选择删除。

Amazon CLI

您可以通过使用 delete-db-instance Amazon CLI 命令删除 RDS Custom for SQL Server 数据库实例。使用必需的参数 `--db-instance-identifier` 识别数据库实例。其余参数与 Amazon RDS 数据库实例的参数相同。

以下示例删除名为 `my-custom-instance` 的 RDS Custom for SQL Server 数据库实例，拍摄最终快照，并保留自动备份。

Example

对于 Linux、macOS 或 Unix：

```
aws rds delete-db-instance \
    --db-instance-identifier my-custom-instance \
    --no-skip-final-snapshot \
    --final-db-snapshot-identifier my-custom-instance-final-snapshot \
    --no-delete-automated-backups
```

对于 Windows：

```
aws rds delete-db-instance ^
    --db-instance-identifier my-custom-instance ^
    --no-skip-final-snapshot ^
    --final-db-snapshot-identifier my-custom-instance-final-snapshot ^
    --no-delete-automated-backups
```

要拍摄最终快照，需要且必须指定 `--final-db-snapshot-identifier` 选项。

要跳过最终快照，请在命令中指定 `--skip-final-snapshot` 选项而不是 `--no-skip-final-snapshot` 和 `--final-db-snapshot-identifier` 选项。

要删除自动备份，请在命令中指定 `--delete-automated-backups` 选项而不是 `--no-delete-automated-backups` 选项。

启动和停止 RDS Custom for SQL Server 数据库实例

您可以启动和停止 RDS Custom for SQL Server 数据库实例。RDS for SQL Server 数据库实例的一般要求和限制同样适用于停止和启动 RDS Custom for SQL Server 数据库实例。有关更多信息，请参阅暂时停止 Amazon RDS 数据库实例。

以下注意事项也适用于启动和停止 RDS Custom for SQL Server 数据库实例：

- 当数据库实例为 `STOPPED` 时，不支持修改 RDS Custom for SQL Server 数据库实例的 EC2 实例属性。
- 只有在将 RDS Custom for SQL Server 数据库实例配置为单可用区时，您才能停止和启动该数据库实例。您无法停止多可用区配置中的 RDS Custom for SQL Server 数据库实例。
- 当您停止 RDS Custom for SQL Server 数据库实例时，将创建 `SYSTEM` 快照。当您再次启动 RDS Custom for SQL Server 数据库实例时，将自动删除此快照。

- 如果您在 RDS Custom for SQL Server 数据库实例停止时删除 EC2 实例，那么当您再次启动 RDS Custom for SQL Server 数据库实例时，`C:` 驱动器将被替换。
- 当您停止 RDS custom for SQL Server 数据库实例时，只要您不修改实例类型，`C:\` 驱动器、主机名和您的自定义配置就会持久存在。
- 以下操作将导致 RDS Custom 将数据库实例放置在支持外围之外，并且您仍将收取数据库实例小时数的费用：
  - 在 Amazon RDS 停止时启动底层 EC2 实例。要解决这个问题，您可以调用 `start-db-instance` Amazon RDS API，或者停止 EC2 以便 RDS Custom 实例返回到 `STOPPED`。
  - RDS Custom for SQL Server 数据库实例为 `ACTIVE` 时停止底层 EC2 实例。

有关停止和启动数据库实例的更多详细信息，请参阅暂时停止 Amazon RDS 数据库实例和启动之前停止的 Amazon RDS 数据库实例。

将 Microsoft Active Directory 用于 RDS Custom for SQL Server

RDS Custom for SQL Server 允许将您的实例加入自托管式 Active Directory（AD）或 Amazon Managed Microsoft AD。无论您的 AD 托管在哪里，例如本地数据中心、Amazon EC2 或任何其他云服务提供商，都是如此。

要对用户和服务进行身份验证，您可以在 RDS Custom for SQL Server 数据库实例上使用 NTLM 或 Kerberos 身份验证，而无需使用中间域和林信任。当用户尝试使用自加入的 Active Directory 在您的 RDS Custom for SQL Server 数据库实例上进行身份验证时，身份验证请求将转发到您指定的自托管式 AD 或 Amazon Managed Microsoft AD。

在以下各节中，您可以找到有关将自托管式 Active Directory 和 Amazon Managed Active Directory 用于 RDS Custom for SQL Server 的信息。

主题

区域和版本可用性

RDS Custom for SQL Server 支持自托管式 AD 和 Amazon Managed Microsoft AD，在所有支持 RDS Custom for SQL Server 的区域中使用 NTLM 或 Kerberos。有关更多信息，请参阅 支持 RDS Custom 的区域和数据库引擎。

配置自托管式 AD 或本地 AD

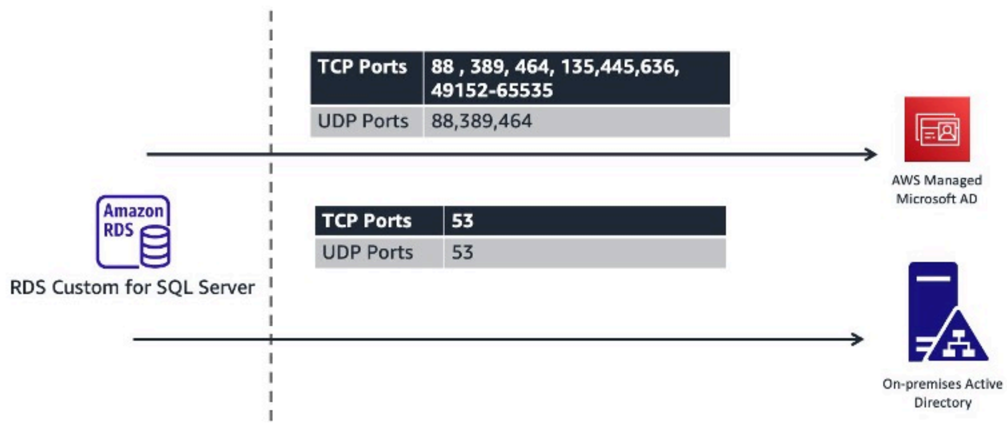
要将本地或自托管式 Microsoft AD 加入 RDS Custom for SQL Server 数据库实例，您的活动域必须按以下方式进行配置：

- 在自托管式或本地 AD 中定义与 RDS Custom for SQL Server 数据库实例关联的 VPC 中的子网。确认您的 VPC 中的子网与 AD 站点中的子网之间没有任何冲突。
- 您 AD 域控制器的域功能级别为 Windows Server 2008 R2 或更高版本。
- 您的 AD 域名不能采用单标签域（SLD）格式。RDS Custom for SQL Server 不支持 SLD 域。
- AD 的完全限定域名（FQDN）不能超过 47 个字符。

配置您的网络连接

按以下方式配置您的自托管式或本地 AD 网络连接：

- 在 Amazon VPC（其中运行 RDS Custom for SQL Server 实例）和您的 AD 之间设置连接。使用 Amazon Direct Connect、Amazon VPN Amazon Transit Gateway 和 VPC 对等连接。
- RDS Custom for SQL Server 安全组和网络 ACL 允许端口上的流量流向您的自托管式或本地 AD。有关更多信息，请参阅 网络配置端口规则。



配置 DNS 解析

设置以下要求以使用自托管式或本地 AD 配置 DNS 解析：

- 在 VPC 内配置 DNS 解析，以解析自托管式 Active Directory 的完全限定域名（FQDN）。FQDN 的一个例子是 `corp.example.local`。要配置 DNS 解析，请将 VPC DNS 解析程序配置为使用 Amazon Route 53 出站端点和解析程序规则转发对某些域的查询。有关更多信息，请参阅配置 Route 53 Resolver 出站端点以解析 DNS 记录。
- 对于同时利用 VPC 和本地资源的工作负载，您必须解析本地托管的 DNS 记录。本地资源可能需要解析托管在 Amazon 上的名称。

要创建混合云设置，请使用解析程序端点和条件转发规则解析本地资源与自定义 VPC 之间的 DNS 查询。有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的解析 VPC 与网络之间的 DNS 查询。

> **Important**
>
> 修改 RDS Custom for SQL Server 上网络接口的 DNS 解析程序设置会导致启用了 DNS 的 VPC 端点不再正常运行。无法访问互联网的私有子网中的实例需要启用了 DNS 的 VPC 端点。

使用 Amazon Directory Service 配置 Microsoft Active Directory

Amazon Managed Microsoft AD 在 Amazon 中创建完全托管的 Microsoft Active Directory，由 Windows Server 2019 提供支持并在 2012 R2 林和域功能级别运行。Amazon Directory Service 在 Amazon VPC 的不同子网中创建域控制器，使您的目录即使在出现故障时也具有很高的可用性。

要使用 Amazon Managed Microsoft AD 创建目录，请参阅《Amazon Directory Service 管理指南》中的 Amazon Managed Microsoft AD 入门。

配置您的网络连接

在目录和数据库实例之间启用跨 VPC 流量

要在同一 VPC 中查找目录和数据库实例，请跳过该步骤，然后转到网络配置端口规则中的下一步。

要在不同的 VPC 中查找目录和数据库实例，请使用 VPC 对等连接或 Amazon Transit Gateway 配置跨 VPC 流量。有关使用 VPC 对等连接的更多信息，请参阅《Amazon VPC 对等连接指南》中的什么是 VPC 对等连接？和《Amazon VPC Transit Gateways》中的什么是 Amazon Transit Gateway？。

使用 VPC 对等连接启用跨 VPC 流量

1. 设置适合的 VPC 路由规则，以便确保网络流量可以双向流动。
2. 允许数据库实例的安全组从目录的安全组接收入站流量。有关更多信息，请参阅 网络配置端口规则。
3. 网络访问控制列表（ACL）不得阻止流量。

如果该目录由不同的 Amazon Web Services 账户拥有，则您必须共享该目录。要与 Amazon Web Services 账户共享 RDS Custom for SQL Server 实例所在的目录，请按照《Amazon Directory Service 管理指南》中的教程：共享 Amazon Managed Microsoft AD 以便无缝地加入 EC2 域操作。

在 Amazon Web Services 账户之间共享目录

1. 使用数据库实例的账户登录 Amazon Directory Service 控制台，并检查在处理之前域是否具有 `SHARED` 状态。
2. 使用数据库实例的账户登录 Amazon Directory Service 控制台后，记下目录 ID 值。您可以使用此 ID 将数据库实例加入域。

配置 DNS 解析

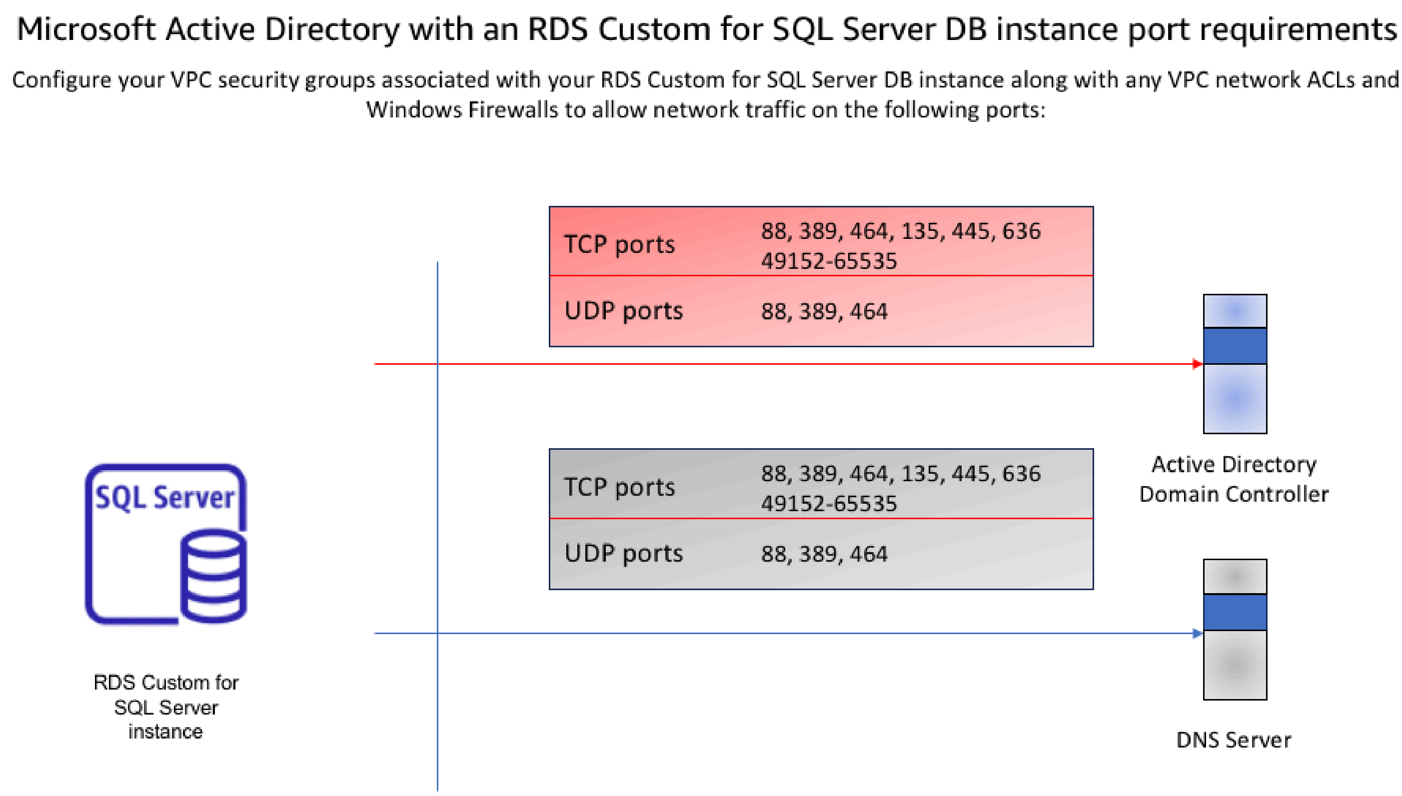
当您使用 Amazon Managed Microsoft AD 创建目录时，Amazon Directory Service 将代表您创建两个域控制器并添加 DNS 服务。

如果您已有 Amazon Managed Microsoft AD 或计划在 VPC 中启动一个（除 RDS Custom for SQL Server 数据库实例之外），请将 VPC DNS 解析程序配置为使用 Route 53 出站和解析程序规则转发对某些域的查询，请参阅配置 Route 53 Resolver 出站端点以解析 DNS 记录。

网络配置端口规则

确保您满足以下网络配置：

- 在您要在其中创建 RDS Custom for SQL Server 数据库实例的 Amazon VPC 与自托管式 Active Directory 或 Amazon Managed Microsoft AD 之间配置了连接。对于自托管式 Active Directory，使用 Amazon Direct Connect、Amazon VPN、VPC 对等连接或 Amazon Transit Gateway 来设置连接。对于 Amazon Managed Microsoft AD，使用 VPC 对等连接来设置连接。
- 确保要在其中创建 RDS Custom for SQL Server 数据库实例的子网的安全组和 VPC 网络 ACL 在端口上允许有下图所示方向的流量。



下表确定了每个端口的作用。

协议	端口	角色
TCP/UDP	53	域名系统（DNS）
TCP/UDP	88	Kerberos 身份验证
TCP/UDP	464	更改/设置密码
TCP/UDP	389	轻型目录访问协议（LDAP）
TCP	135	分布式计算环境/端点映射器（DCE/EPMAP）
TCP	445	目录服务 SMB 文件共享
TCP	636	基于 TLS/SSL 的轻型目录访问协议（LDAPS）

协议	端口	角色
TCP	49152 - 65535	RPC 的临时端口

- 通常，域 DNS 服务器位于 AD 域控制器中。您无需配置 VPC DHCP 选项集即可使用此功能。有关更多信息，请参阅《Amazon VPC 用户指南》中的 DHCP 选项集。

> **Important**
>
> 如果您使用的是 VPC 网络 ACL，则还必须允许动态端口（49152-65535）上有来自 RDS Custom for SQL Server 数据库实例的出站流量。确保这些流量规则也镜像到适用于每个 AD 域控制器、DNS 服务器和 RDS Custom for SQL Server 数据库实例的防火墙上。
> 虽然 VPC 安全组要求仅在发起网络流量的方向打开端口，但大多数 Windows 防火墙和 VPC 网络 ACL 要求双向打开端口。

网络验证

在将 RDS Custom 实例加入自托管式 AD 或Amazon Managed Microsoft AD 之前，请从与您计划启动 RDS Custom for SQL Server 实例所在的同一 VPC 中的 EC2 实例中检查以下内容。

- 检查您能否将完全限定域名（FQDN）解析为域控制器 IP。

```
nslookup corp.example.com
```

  该命令必须返回类似的输出：

```
Server:  ip-10-0-0-2.us-west-2.compute.internal
Address:  25.0.0.2

Non-authoritative answer:
Name:    corp.example.com
Addresses:  40.0.9.25 (DC1 IP)
            40.0.50.123 (DC2 IP)
```

- 解析您要启动 RDS Custom 实例所在的同一 VPC 中的 EC2 实例中的 Amazon 服务：

```
$region='input-your-aws-region'
$domainFQDN='input-your-domainFQDN'
```

```
function Test-DomainPorts {
    param (
        [string]$Domain,
        [array]$Ports
    )

    foreach ($portInfo in $Ports) {
        try {
            $conn = New-Object System.Net.Sockets.TcpClient
            $connectionResult = $conn.BeginConnect($Domain, $portInfo.Port, $null,
 $null)
            $success = $connectionResult.AsyncWaitHandle.WaitOne(1000) # 1 second
 timeout
            if ($success) {
                $conn.EndConnect($connectionResult)
                $result = $true
            } else {
                $result = $false
            }
        }
        catch {
            $result = $false
        }
        finally {
            if ($null -ne $conn) {
                $conn.Close()
            }
        }
        Write-Host "$($portInfo.Description) port open: $result"
    }
}

# Check if ports can be reached
$ports = @(
    @{Port = 53;   Description = "DNS"},
    @{Port = 88;   Description = "Kerberos"},
    @{Port = 389;  Description = "LDAP"},
    @{Port = 445;  Description = "SMB"},
    @{Port = 5985; Description = "WinRM"},
    @{Port = 636;  Description = "LDAPS"},
    @{Port = 3268; Description = "Global Catalog"},
    @{Port = 3269; Description = "Global Catalog over SSL"},
    @{Port = 9389; Description = "AD DS"}
```

```
)

function Test-DomainReachability {
    param (
        [string]$DomainName
    )

    try {
        $dnsResults = Resolve-DnsName -Name $DomainName -ErrorAction Stop
        Write-Host "Domain $DomainName is successfully resolving to following IP
 addresses: $($dnsResults.IpAddress)"
        Write-Host ""
        return $true
    }
    catch {
        Write-Host ""
        Write-Host "Error Message: $($_.Exception.Message)"
        Write-Host "Domain $DomainName reachability check failed, please Configure
 DNS resolution"
        return $false
    }
}

$domain = (Get-WmiObject Win32_ComputerSystem).Domain
if ($domain -eq 'WORKGROUP') {
    Write-Host ""
    Write-Host "Host $env:computername is still part of WORKGROUP and not part of any
 domain"
    }
else {
    Write-Host ""
    Write-Host "Host $env:computername is joined to $domain domain"
    Write-Host ""
    }


$isReachable = Test-DomainReachability -DomainName $domainFQDN
if ($isReachable) {
    write-Host "Checking if domain $domainFQDN is reachable on required ports  "
    Test-DomainPorts -Domain $domainFQDN -Ports $ports
}
else {
    Write-Host "Port check skipped. Domain not reachable"
}
```

```
# Get network adapter configuration
$networkConfig = Get-WmiObject Win32_NetworkAdapterConfiguration |
                 Where-Object { $_.IPEnabled -eq $true } |
                 Select-Object -First 1

# Check DNS server settings
$dnsServers = $networkConfig.DNSServerSearchOrder

if ($dnsServers) {
    Write-Host "`nDNS Server settings:"
    foreach ($server in $dnsServers) {
        Write-Host "  - $server"
    }
} else {
    Write-Host "`nNo DNS servers configured or unable to retrieve DNS server
 information."
}

write-host ""

# Checks reachability to dependent services
$services = "s3", "ec2", "secretsmanager", "logs", "events", "monitoring", "ssm",
 "ec2messages", "ssmmessages"

function Get-TcpConnectionAsync {
    param (
        $ServicePrefix,
        $region
    )
    $endpoint = "${ServicePrefix}.${region}.amazonaws.com"
    $tcp = New-Object Net.Sockets.TcpClient
    $result = $false

    try {
        $connectTask = $tcp.ConnectAsync($endpoint, 443)
        $timedOut = $connectTask.Wait(3000)
        $result = $tcp.Connected
    }
    catch {
        $result = $false
    }
```

```
    return $result
}

foreach ($service in $services) {
    $validationResult = Get-TcpConnectionAsync -ServicePrefix $service -Region
 $region
    Write-Host "Reachability to $service is $validationResult"
}
```

对于 `s3`、`ec2`、`secretsmanager`、`logs`、`events`、`monitoring`、`ssm`、`ec2messages` 和 `ssmmessages`，`TcpTestSucceeded` 值必须返回 `True`。

为 RDS Custom for SQL Server 实例设置 Windows 身份验证

我们建议为任何拥有已加入 AD 域的 RDS Custom for SQL Server 数据库实例的 Amazon Web Services 账户创建专用 OU 以及作用域为该 OU 的服务凭证。通过专用 OU 和服务凭证，您可以避免权限冲突并遵循最低权限原则。

Active Directory 级别的组策略可能与 Amazon 自动化和权限发生冲突。我们建议选择仅适用于您为 RDS Custom for SQL Server 创建的 OU 的 GPO。

- 要在自托管式或本地 AD 中创建 OU 和 AD 域用户，您可以以域管理员身份连接域控制器。
- 要在 Amazon Directory Service 目录中创建用户和组，必须连接到管理实例，还必须已作为有权创建用户和组的用户登录。有关更多信息，请参阅《Amazon Directory Service 管理指南》中的 Amazon Managed Microsoft AD 中的用户和组管理。
- 要从 Amazon EC2 Windows Server 实例中管理 Active Directory，您需要在 EC2 实例上安装 Active Directory Domain Services 和 Active Directory Lightweight Directory Services 工具。有关更多信息，请参阅《Amazon Directory Service 管理指南》中的为 Amazon Managed Microsoft AD 安装 Active Directory 管理工具。
- 为了便于管理，建议您将这些工具安装在单独的 EC2 实例上，而不是 RDS Custom for SQL Server 数据库实例上。

AD 域服务账户的要求如下：

- 您必须在 AD 域中有一个服务账户，该账户具有将计算机加入该域的委派权限。域服务账户是您的 AD 中的一个用户账户，该账户具有执行某些任务的委派权限。
- 在您要加入 RDS Custom for SQL Server 实例的组织单位中向域服务账户委派以下权限：

  - 验证写入 DNS 主机名的能力
  - 验证写入服务主体名称的能力
  - 创建和删除计算机对象
- 对于自托管式和本地 AD，域服务账户必须是“Amazon 域名系统委托管理员”组的成员。
- 对于 Amazon Managed Microsoft AD，域名服务账户应是“DnsAdmins”组的成员。

这些是将计算机对象加入自托管式 AD 和 Amazon Managed Microsoft AD 至少需要具备的一组权限。有关更多信息，请参阅 Microsoft Windows Server 文档中的错误：当已委派控制的非管理员用户尝试将计算机加入域控制器时，访问被拒绝。

> **Important**
>
> 创建数据库实例后，请勿移动 RDS Custom for SQL Server 在组织单位（OU）中创建的计算机对象。移动关联对象可能会导致您的 RDS Custom for SQL Server 数据库实例出现配置错误。如果您需要移动 Amazon RDS 创建的计算机对象，请通过 ModifyDBInstance 操作用计算机对象所需位置修改域参数。

主题

步骤 1：在 AD 中创建组织单位（OU）

请按照以下步骤在 AD 中创建组织单位：

在 AD 中创建 OU

1. 以域管理员身份连接到您的域 AD。

2. 打开 Active Directory 用户和计算机，然后选择要在其中创建 OU 的域。
3. 右键单击该域并选择新建，然后选择组织单位。
4. 为 OU 输入名称。

   启用保护容器免遭意外删除。
5. 选择确定。您的新 OU 将出现在您的域下。

对于 Amazon Managed Microsoft AD，此 OU 的名称基于您在创建目录时键入的 NetBIOS 名称。此 OU 属于 Amazon 并且包含您有权完全控制的所有 Amazon 相关目录对象。默认情况下，此 OU 下存在两个子 OU，名为 Computers 和 Users。RDS Custom 创建的新 OU 是基于 NetBIOS 的 OU 的子级。

步骤 2：创建 AD 域用户

域用户凭证用于 Secrets Manager 中的密钥。

在您的 AD 中创建 AD 域用户

1. 打开 Active Directory 用户和计算机，然后选择要在其中创建用户的域和 OU。
2. 右键单击用户对象并选择新建，然后选择用户。
3. 输入用户的名字、姓氏和登录名。单击下一步。
4. 输入用户的密码。请勿选择用户下次登录时必须更改密码或账户已禁用。单击下一步。
5. 单击确定。您的新用户将出现在您的域下。

步骤 3：将控制权委派给自托管式 AD 或 Amazon Managed Microsoft AD 中的 AD 用户

将控制权委派给域中的 AD 域用户

1. 打开 Active Directory 用户和计算机 MMC 管理单元，然后选择您的域。
2. 右键单击您之前创建的 OU，然后选择委派控制权。
3. 在委派控制权向导中，单击下一步。
4. 在用户或组部分，单击添加。
5. 在选择用户、计算机或组中，输入您创建的 AD 用户，然后单击检查名称。如果您的 AD 用户检查成功，请单击确定。
6. 在用户或组部分，确认您的 AD 用户已添加，然后单击下一步。

7. 在要委派的任务部分，选择创建要委派的自定义任务，然后单击下一步。
8. 在 Active Directory 对象类型部分：

   选择仅文件夹中的以下对象。

   选择计算机对象

   选择在此文件夹中创建选定对象

   选择删除此文件夹中的选定对象，然后单击下一步。
9. 在权限部分：

   将常规保持为选中状态。

   选择已验证写入 DNS 主机名。

   选择已验证写入服务主体名称，然后单击下一步。
10. 在完成控制权委派向导中，请确认您的设置，然后单击完成。

步骤 4：创建密钥

在包含要加入 Active Directory 的 RDS Custom for SQL Server 数据库实例的同一 Amazon Web Services 账户和区域中创建密钥。存储在步骤 2：创建 AD 域用户中创建的 AD 域用户的凭证。

Console

- 在 Amazon Secrets Manager 中，选择存储新密钥。
- 对于密钥类型，请选择其他密钥类型。
- 对于键/值对，请添加两个键：
  - 对于第一个键，输入 `SELF_MANAGED_ACTIVE_DIRECTORY_USERNAME`，然后输入您的 AD 用户名作为值。
  - 对于第二个键，输入 `SELF_MANAGED_ACTIVE_DIRECTORY_PASSWORD`，然后输入域中 AD 用户的密码。
- 在加密密钥中，输入创建 RDS Custom for SQL Server 实例时所用的相同 Amazon KMS 密钥。
- 对于密钥名称，选择以 `do-not-delete-rds-custom-` 开头的密钥名称，以允许您的实例配置文件访问此密钥。如果您想为密钥选择一个不同的名称，请更新 `RDSCustomInstanceProfile` 以访问密钥名称。
- （可选）对于描述，输入密钥名称的描述。

- 添加标签 `Key="AWSRDSCustom",Value="custom-sqlserver"`
- 单击保存，然后单击下一步。
- 对于配置轮换设置，保留默认值并选择下一步。
- 查看密钥的设置，然后单击存储。
- 选择新密钥，然后复制密钥 ARN 的值。下一步将用它来设置您的 Active Directory。

CLI

在 CLI 中运行以下命令来创建密钥：

```
# Linux based
aws secretsmanager create-secret \
--name do-not-delete-rds-custom-DomainUserCredentails \
--description "Active directory user credentials for managing RDS Custom" \
--secret-string "{\"SELF_MANAGED_ACTIVE_DIRECTORY_USERNAME\":\"tester\",
\"SELF_MANAGED_ACTIVE_DIRECTORY_PASSWORD\":\"xxxxxxxx\"}" \
--kms-key-id <RDSCustomKMSKey> \
--tags Key="AWSRDSCustom",Value="custom-sqlserver"

# Windows based
aws secretsmanager create-secret ^
--name do-not-delete-rds-custom-DomainUserCredentails ^
--description "Active directory user credentials for managing RDS Custom" ^
--secret-string "{\"SELF_MANAGED_ACTIVE_DIRECTORY_USERNAME\":\"tester\",
\"SELF_MANAGED_ACTIVE_DIRECTORY_PASSWORD\":\"xxxxxxxx\"}" ^
--kms-key-id <RDSCustomKMSKey> ^
--tags Key="AWSRDSCustom",Value="custom-sqlserver"
```

步骤 5：创建或修改 RDS Custom for SQL Server 数据库实例

创建或修改 RDS Custom for SQL Server 数据库实例，以便与您的目录一起使用。您可以使用控制台、CLI 或 RDS API 将数据库实例与目录关联。您可以通过下列方式之一来执行该操作：

- 使用控制台、create-db-instance CLI 命令或 CreateDBInstance RDS API 操作创建新的 SQL Server 数据库实例。

  有关说明，请参阅创建 Amazon RDS 数据库实例。
- 使用控制台、modify-db-instance CLI 命令或 ModifyDBInstance RDS API 操作修改现有 SQL Server 数据库实例。

有关说明，请参阅修改 Amazon RDS 数据库实例。

- 使用控制台、restore-db-instance-from-db-snapshot CLI 命令或 RestoreDBInstanceFromDBSnapshot RDS API 操作，从数据库快照还原 SQL Server 数据库实例。

  有关说明，请参阅还原到数据库实例。

- 使用控制台、restore-db-instance-to-point-in-time CLI 命令或 RestoreDBInstanceToPointInTime RDS API 操作，将 SQL Server 数据库实例还原到某个时间点。

  有关说明，请参阅将 Amazon RDS 的数据库实例还原到指定时间。

> **Note**
>
> 如果已手动将 RDS Custom for SQL Server 实例加入 AD，请检查 网络配置端口规则 和 网络验证 的设置，然后完成步骤 1 至步骤 4。将 `--domain-fqdn`、`--domain-ou` 和 `--domain-auth-secret-arn` 更新到您的 AD，以便在 RDS Custom 中注册域加入凭证和配置，以监控、注册 CNAME 并采取恢复操作。

使用 Amazon CLI 时，数据库实例需要以下参数才能使用您创建的目录：

- 对于 `--domain-fqdn` 参数，请使用自托管式 AD 的完全限定域名。
- 对于 `--domain-ou` 参数，请使用您在自托管式 AD 中创建的 OU。
- 对于 `--domain-auth-secret-arn` 参数，请使用您创建的密钥 ARN 的值。

> **Important**
>
> 如果您修改数据库实例，以加入自托管式 AD 域或 Amazon Managed Microsoft AD 或者从中删除，则需要重启数据库实例才能使修改生效。您可以选择立即应用更改，也可以等到下一个维护时段。选择立即应用选项会导致单可用区数据库实例停机。多可用区数据库实例集群会在完成重启之前执行失效转移。有关更多信息，请参阅 修改 Amazon RDS 数据库实例。

以下 CLI 命令创建一个新的 RDS Custom for SQL Server 数据库实例并将其加入自托管式或 Amazon Managed Microsoft AD 域。

对于 Linux、macOS 或 Unix：

```
aws rds create-db-instance  \
--engine custom-sqlserver-se \
--engine-version 15.00.4312.2.v1 \
--db-instance-identifier my-custom-instance \
--db-instance-class db.m5.large \
--allocated-storage 100 --storage-type io1 --iops 1000 \
--master-username my-master-username \
--master-user-password my-master-password \
--kms-key-id  my-RDSCustom-key-id \
--custom-iam-instance-profile AWSRDSCustomInstanceProfileForRdsCustomInstance  \
--domain-fqdn "corp.example.com" \
--domain-ou "OU=RDSCustomOU,DC=corp,DC=example,DC=com" \
--domain-auth-secret-arn "arn:aws:secretsmanager:region:account-number:secret:do-not-
delete-rds-custom-my-AD-test-secret-123456" \
--db-subnet-group-name my-DB-subnet-grp \
--vpc-security-group-ids  my-securitygroup-id \
--no-publicly-accessible \
--backup-retention-period 3 \
--port 8200 \
--region us-west-2 \
--no-multi-az
```

对于 Windows：

```
aws rds create-db-instance  ^
--engine custom-sqlserver-se ^
--engine-version 15.00.4312.2.v1 ^
--db-instance-identifier my-custom-instance ^
--db-instance-class db.m5.large ^
--allocated-storage 100 --storage-type io1 --iops 1000 ^
--master-usernamemy-master-username ^
--master-user-password my-master-password ^
--kms-key-id  my-RDSCustom-key-id ^
--custom-iam-instance-profile AWSRDSCustomInstanceProfileForRdsCustomInstance  ^
--domain-fqdn "corp.example.com" ^
--domain-ou "OU=RDSCustomOU,DC=corp,DC=example,DC=com" ^
--domain-auth-secret-arn "arn:aws:secretsmanager:region:account-number:secret:do-not-
delete-rds-custom-my-AD-test-secret-123456" ^
--db-subnet-group-name my-DB-subnet-grp ^
--vpc-security-group-ids  my-securitygroup-id ^
--no-publicly-accessible ^
--backup-retention-period 3 ^
```

```
--port 8200 ^
--region us-west-2 ^
--no-multi-az
```

> **Important**
>
> 如果您的 Amazon Managed Microsoft AD 的 NetBIOS 为 corpexample，那么它本身就会显示为 OU。之前创建的任何新 OU 都将显示为嵌套 OU。对于 Amazon Managed Microsoft AD，将 `--domain-ou` 设置为 `"OU=RDSCustomOU,OU=corpexample,DC=corp,DC=example,DC=com"`。

以下命令将现有 RDS Custom for SQL Server 数据库实例修改为使用 Active Directory 域。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier my-custom-instance \
    --domain-fqdn "corp.example.com" \
    --domain-ou "OU=RDSCustomOU,DC=corp,DC=example,DC=com" \
    --domain-auth-secret-arn "arn:aws:secretsmanager:region:account-number:secret:do-not-delete-rds-custom-my-AD-test-secret-123456" \
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier my-custom-instance ^
    --domain-fqdn "corp.example.com" ^
    --domain-ou "OU=RDSCustomOU,DC=corp,DC=example,DC=com" ^
    --domain-auth-secret-arn "arn:aws:secretsmanager:region:account-number:secret:do-not-delete-rds-custom-my-AD-test-secret-123456" ^
```

以下 CLI 命令从 Active Directory 域中删除 RDS Custom for SQL Server 数据库实例。

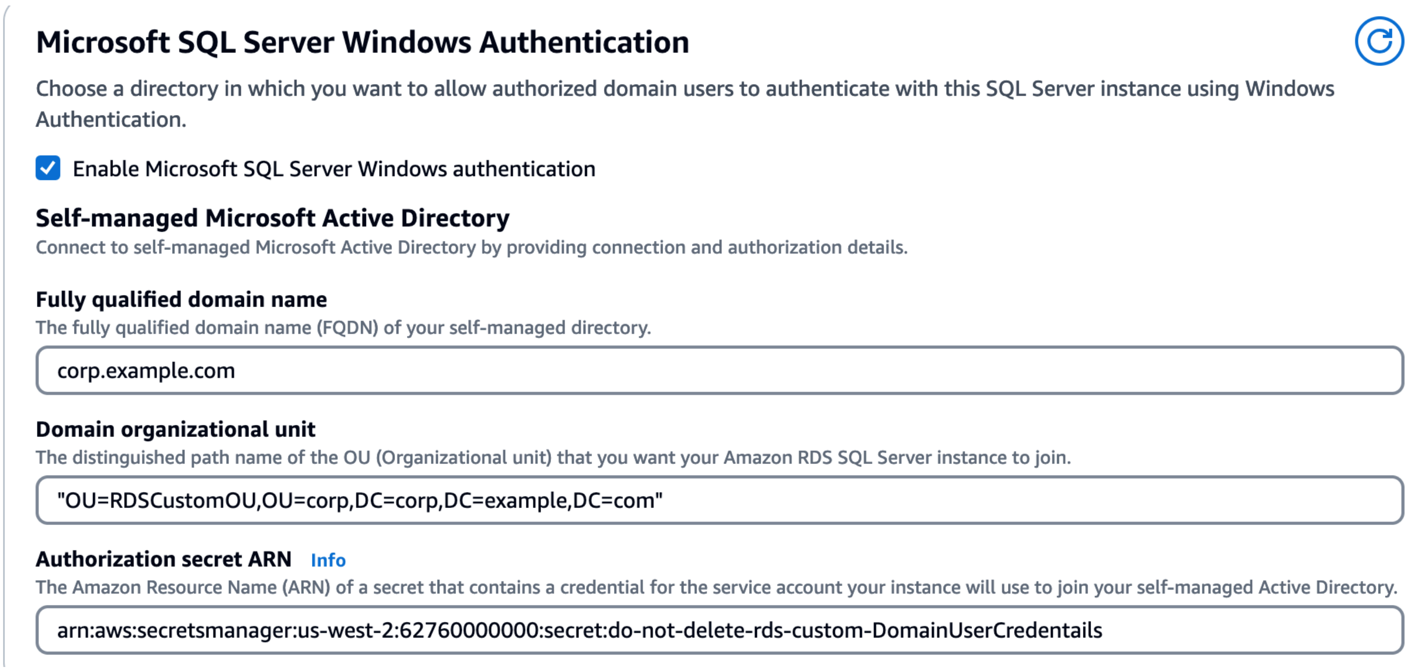
对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier my-custom-instance \
    --disable-domain
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier my-custom-instance ^
    --disable-domain
```

使用控制台创建或修改实例时，单击启用 Microsoft SQL Server Windows 身份验证以查看以下选项。



您负责确保您的域 FQDN 解析为域控制器 IP 地址。如果域控制器 IP 未解析，则域加入操作将失败，但会成功创建 RDS Custom for SQL Server 实例。有关问题排查信息，请参阅Active Directory 故障排除。

步骤 6：创建 Windows 身份验证 SQL Server 登录名

使用 Amazon RDS 主用户凭证连接 SQL Server 数据库实例，就像连接任何其他数据库实例一样。由于数据库实例已加入 AD 域，因此您可以预配置 SQL Server 登录名和用户。您可以通过 AD 域中的 AD 用户和组实用程序执行此操作。通过对这些 Windows 登录名授予和撤销标准 SQL Server 权限来管理数据库权限。

对于 AD 用户，要对 SQL Server 进行身份验证，AD 用户或其所属的 Active Directory 组必须存在 SQL Server Windows 登录名。访问权限的精细控制是通过对这些 SQL Server 登录名授予和撤销权限来处理的。AD 用户没有 SQL Server 登录名，或者用户所属 AD 组中的登录名无法访问 SQL Server 数据库实例。

创建 AD SQL Server 登录名需要 `ALTER ANY LOGIN` 权限。如果您尚未使用此权限创建任何登录名，请使用 SQL Server 身份验证以数据库实例的主用户身份进行连接，并在主用户的上下文下创建 AD SQL Server 登录名。

您可以运行数据定义语言（DDL）命令（如以下示例），为 AD 用户或组创建 SQL Server 登录名。

```
USE [master]
GO
CREATE LOGIN [mydomain\myuser] FROM WINDOWS WITH DEFAULT_DATABASE = [master],
 DEFAULT_LANGUAGE = [us_english];
GO
```

域中用户（人和应用程序）现在均可从加入域的客户端计算机使用 Windows 身份验证连接到 RDS Custom for SQL Server 实例。

步骤 7：使用 Kerberos 或 NTLM 身份验证

使用 RDS 端点进行 NTLM 身份验证

每个 Amazon RDS 数据库实例都有一个端点，每个端点都包含数据库实例的 DNS 名称和端口号。要使用 SQL 客户端应用程序连接到数据库实例，您需要该数据库实例的 DNS 名称和端口号。要使用 NTLM 身份验证进行身份验证，必须连接到 RDS 端点。

在计划内数据库维护或计划外服务中断期间，Amazon RDS 将自动失效转移到最新辅助数据库，以便操作能够在不进行手动干预的情况下快速恢复。主实例和辅助实例使用相同的端点，其物理网络地址作为失效转移过程的一部分转换到辅助实例。在失效转移时，您不必重新配置应用程序。

Kerberos 身份验证

RDS Custom for SQL Server 的基于 Kerberos 的身份验证要求连接到特定的服务主体名称（SPN）。但是，在失效转移事件发生后，应用程序可能不知道新 SPN。为了解决这个问题，RDS Custom for SQL Server 提供了一个基于 Kerberos 的端点。

基于 Kerberos 的端点遵循一种特定的格式。如果您的 RDS 端点为 `*rds-instance-name*.*account-region-hash*.*aws-region*.rds.amazonaws.com`，则相应基于 Kerberos 的端点将为 `*rds-instance-name*.*account-region-hash*.*aws-region*.awsrds.*fully qualified domain name (FQDN)*`。

例如，如果 RDS 端点为 `ad-test.cocv6zwtircu.us-east-1.rds.amazonaws.com`，域名为 `corp-ad.company.com`，则基于 Kerberos 的端点将为 `ad-test.cocv6zwtircu.us-east-1.awsrds.corp-ad.company.com`。

这个基于 Kerberos 的端点可用于使用 Kerberos 对 SQL Server 实例进行身份验证，即使在失效转移事件发生后也是如此，因为端点会自动更新为指向主 SQL Server 实例的新 SPN。

查找您的 CNAME

要查找您的 CNAME，请连接到域控制器并打开 DNS 管理器。导航到正向查找区域和您的 FQDN。

浏览 awsrds、aws-region，以及账户和区域特定的哈希值。

如果您要连接 RDS Custom EC2 实例并尝试使用 CNAME 在本地连接到数据库，则您的连接将使用 NTLM 身份验证而不是 Kerberos。

如果从远程客户端连接 CNAME 后返回了 NTLM 连接，请检查所需端口是否已列入允许列表。

要检查您的连接是否使用 Kerberos，请运行以下查询：

```
SELECT net_transport, auth_scheme
    FROM sys.dm_exec_connections
    WHERE session_id = @@SSPID;
```

在域中管理数据库实例

您可以使用控制台、Amazon CLI 或 Amazon RDS API 来管理数据库实例及其与您的域的关系。例如，您可以将数据库实例移入域、移出域或在域之间移动。

例如，使用 Amazon RDS API，您可以执行下列操作：

- 要为失败的成员资格重新尝试域加入，请使用 ModifyDBInstance API 操作并指定当前成员资格的目录 ID。
- 要为成员资格更新 IAM 角色名称，请使用 `ModifyDBInstance` API 操作并指定当前成员资格的目录 ID 和新的 IAM 角色。
- 要从域中删除数据库实例，请使用 `ModifyDBInstance` API 操作并指定 none 作为域参数。
- 要将数据库实例从一个域移至另一个域，请使用 `ModifyDBInstance` API 操作并指定新域的域标识符作为域参数。
- 要列出每个数据库实例的成员资格，请使用 DescribeDBInstances API 操作。

还原 RDS Custom for SQL Server 数据库实例并将其添加到 Active Directory 域

您可以还原数据库快照或对 SQL Server 数据库实例执行时间点故障恢复（PITR），然后将其添加到 Active Directory 域。数据库实例还原后，使用步骤 5：创建或修改 RDS Custom for SQL Server 数据库实例中介绍的过程修改此实例以将数据库实例添加到 AD 域中。

了解域成员资格

在创建或修改数据库实例后，实例将成为域的成员。Amazon 控制台将指示数据库实例的域成员资格状态。数据库实例的状态可以是以下状态之一：

- 已联接 – 实例是域的成员。
- 正在联接 – 实例正处于成为域成员的过程中。
- 待联接 – 实例成员资格待定。
- pending-maintenance-join – Amazon将在下一计划维护期间尝试使实例成为域成员。
- 待删除 – 等待从域中删除实例。
- pending-maintenance-removal – Amazon将在下一计划维护期间尝试从域中删除实例。
- 失败 – 配置问题阻碍实例联接域。在重新发出实例修改命令之前检查并修复配置。
- 正在删除 – 正从域中删除实例。

成为域成员的请求可能因网络连接问题或不正确的 IAM 角色而失败。例如，您可能会创建数据库实例或修改现有实例，并且尝试使数据库成为某个域的成员会失败。在这种情况下，您重新发出命令以创建或修改数据库实例，或者修改新创建的实例以加入域。

Active Directory 故障排除

以下是您在设置或修改 AD 时可能遇到的问题。

错误代码	描述	常见原因	故障排除建议
错误 2/0x2	系统找不到指定的文件。	使用 `–domain-ou` 参数指定的组织单位（OU）的格式或位置无效。通过 Amazon Secrets Manager 指定的域服务账户缺少加入 OU 所需的权限。	查看 `–domain-ou` 参数。确保域服务账户拥有相应的 OU 权限。
错误 5/0x5	访问被拒绝。	域服务账户的权限配置不正确，或者域中已经存在该计算机账户。	查看域中的域服务账户权限，并验证 RDS 计算机账户在域中是否重复。您可以通过在 RDS Custom for SQL Server 数

错误代码	描述	常见原因	故障排除建议
			据库实例上运行 SELECT @@SERVERNAME 来验证 RDS 计算机账户的名称。如果您使用的是多可用区，请尝试通过失效转移重启，然后再次验证 RDS 计算机账户。有关更多信息，请参阅 重启中的数据库实例。
错误 87/0x57	参数不正确。	通过 Amazon Secrets Manager 指定的域服务账户不具备相应的权限。用户配置文件也可能已损坏。	查看域服务账户的要求。
错误 234/0xEA	指定的组织单位（OU）不存在。	使用 –domain-ou 参数指定的 OU 在您的 AD 中不存在。	查看 –domain-ou 参数并确保指定的 OU 在您的 AD 中存在。
错误 1326/0x52E	用户名或密码不正确。	Amazon Secrets Manager 中提供的域服务账户凭证包含未知的用户名或错误的密码。您的 AD 中也可能已禁用域账户。	确保 Amazon Secrets Manager 中提供的凭证正确无误，并且在您的 Active Directory 中启用了域账户。
错误 1355/0x54B	指定的域不存在或无法访问。	域已关闭、指定的 DNS IP 集无法访问，或指定的 FQDN 无法访问。	查看 –domain-dns-ips 和 –domain-fqdn 参数，确保正确。查看您的 RDS Custom for SQL Server 数据库实例的网络配置，并确保您的 AD 可以访问。

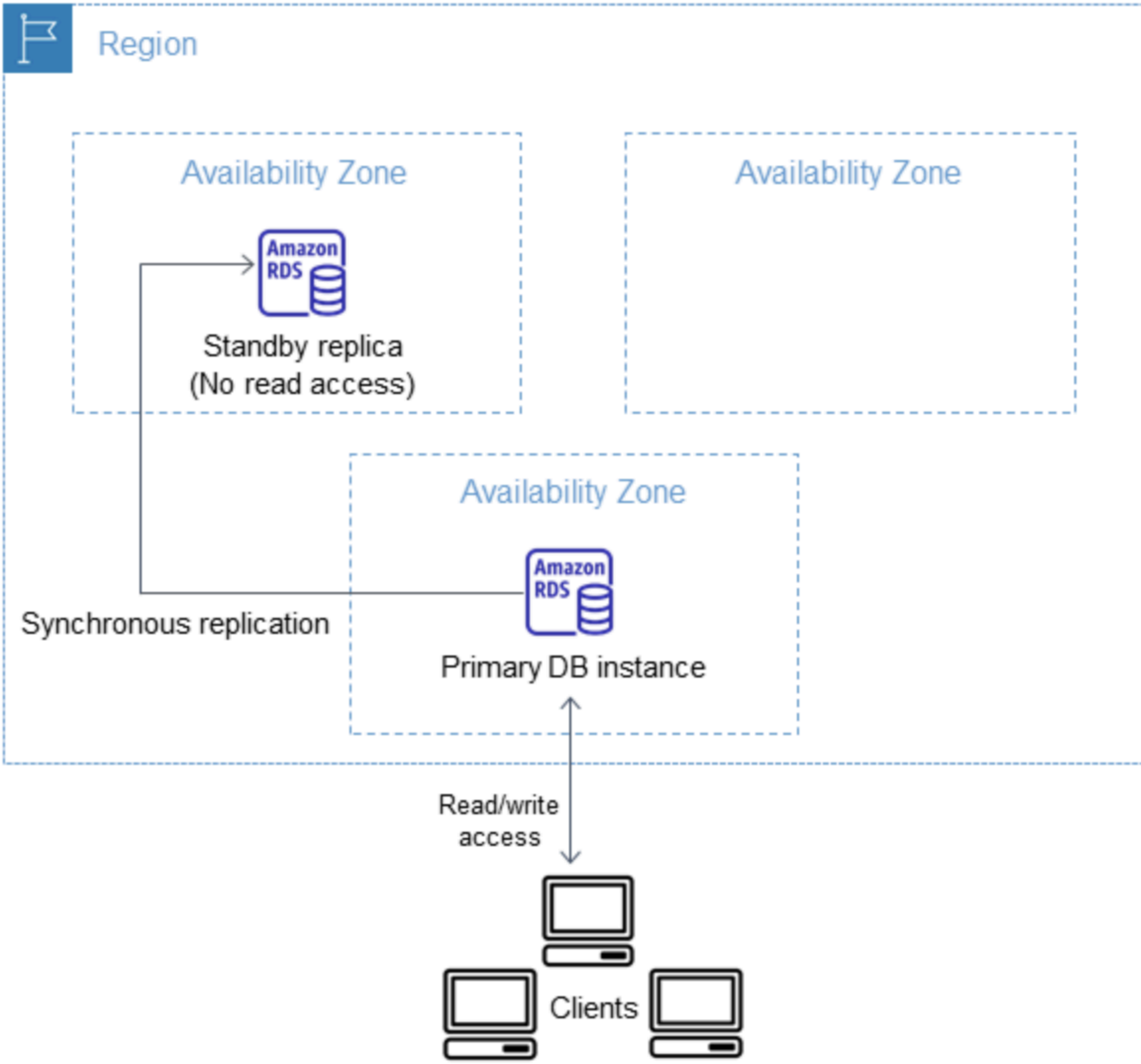
错误代码	描述	常见原因	故障排除建议
错误 1722/0x6BA	RPC 服务器不可用。	连接您的 AD 域的 RPC 服务时出现问题。这可能由于服务或网络问题导致。	验证 RPC 服务是否正在您的域控制器上运行，以及是否可以在您的域上从 RDS Custom for SQL Server 数据库实例访问 TCP 端口 135 和 `49152-65535` 。
错误 2224/0x8b0	用户账户已存在。	正在尝试添加到您 AD 的计算机账户已经存在。	通过在您的 RDS Custom for SQL Server 数据库实例上运行 `SELECT @@SERVERNAME` 来识别计算机账户，然后小心地将其从您的 AD 中删除。
错误 2242/0x8c2	此用户的密码已过期。	通过 Amazon Secrets Manager 指定的域服务账户的密码已过期。	更新用于将 RDS Custom for SQL Server 数据库实例加入 AD 的域服务账户的密码。

管理 RDS Custom for SQL Server 的多可用区部署

在 RDS Custom for SQL Server 的多可用区数据库实例部署中，Amazon RDS 会自动在不同可用区（AZ）中预调配和维护一个同步备用副本。主数据库实例可以跨可用区同步复制到备用副本以提供数据冗余。

> **Important**
>
> RDS Custom for SQL Server 的多可用区部署与 RDS for SQL Server 的多可用区部署不同。与 RDS for SQL Server 的多可用区不同，对于 RDS Custom for SQL Server，在创建多可用区数据库实例之前，您必须设置先决条件，因为 RDS Custom 在您自己的账户内运行，这需要权限。
>
> 如果您未完成先决条件，则多可用区数据库实例可能无法运行，或者自动恢复为单可用区数据库实例。有关先决条件的更多信息，请参阅RDS Custom for SQL Server 的多可用区部署的先决条件。

在计划内的系统维护期间，运行具有高可用性的数据库实例可以提高可用性。在进行计划的数据维护或发生未计划的服务中断时，Amazon RDS 会自动故障转移到最新的辅助数据库实例。此功能让数据库操作快速恢复，而不需要手动干预。主实例和备用实例使用相同的终端节点，其物理网络地址作为故障转移过程的一部分转换到辅助副本。在故障转移时，您不必重新配置应用程序。



您可以通过在创建 RDS Custom 数据库实例时指定多可用区，来创建 RDS Custom for SQL Server 多可用区部署。您可以使用控制台，通过修改数据库实例并指定多可用区选项，将现有 RDS Custom for SQL Server 数据库实例转换为多可用区部署。还可以使用 Amazon CLI 或 Amazon RDS API 指定多可用区数据库实例部署。

RDS 控制台显示备用副本的可用区（辅助可用区）。还可以使用 `describe-db-instances` CLI 命令或 `DescribeDBInstances` API 操作来查找辅助可用区。

与单可用区部署相比，具有多可用区部署的 RDS Custom for SQL Server 数据库实例的写入和提交延迟可能会增加。这种增加可能是由于数据库实例之间的同步数据复制造成的。尽管 Amazon 设计用于在可用区之间提供低延迟网络连接，但如果您的部署故障转移到备用副本，延迟可能会发生变化。

> **Note**
>
> 对于生产工作负载，我们建议您使用具有预调配 IOPS（每秒输入/输出操作数）的数据库实例类，以获得快速、一致的性能。有关数据库实例类的更多信息，请参阅 Amazon RDS Custom for SQL Server 的要求和限制。

主题

区域和版本可用性

以下 SQL Server 版本支持 RDS Custom for SQL Server 的多可用区部署：

- SQL Server 2022 和 2019：企业版、标准版、Web 版和开发版

> **Note**
>
> SQL Server 2019 CU8（15.00.4073.23）或更低版本不支持 RDS Custom for SQL Server 的多可用区部署。

RDS Custom for SQL Server 的多可用区部署在 RDS Custom for SQL Server 可用的所有区域中均可用。有关 RDS Custom for SQL Server 的多可用区部署的区域可用性的更多信息，请参阅 支持 RDS Custom for SQL Server 的区域和数据库引擎。

RDS Custom for SQL Server 的多可用区部署的限制

RDS Custom for SQL Server 的多可用区部署具有以下限制：

- 不支持跨区域多可用区部署。
- 您不能将辅助数据库实例配置为接受数据库读取活动。
- 当您在多可用区部署中使用自定义引擎版本（CEV）时，辅助数据库实例也将使用相同的 CEV。辅助数据库实例无法使用不同的 CEV。

RDS Custom for SQL Server 的多可用区部署的先决条件

如果您具有现有的 RDS Custom for SQL Server 单可用区部署，则在将其修改为多可用区部署之前，需要满足以下附加先决条件。您可以选择手动完成先决条件，也可以选择使用提供的 CloudFormation 模板完成先决条件。最新的 CloudFormation 模板同时包含适用于单可用区部署和多可用区部署的先决条件。

> **Important**
>
> 为了简化设置，我们建议您使用网络设置说明中提供的最新 Amazon CloudFormation 模板文件来创建先决条件。有关更多信息，请参阅 使用 Amazon CloudFormation 进行配置。

> **Note**
>
> 将现有的 RDS Custom for SQL Server 单可用区部署修改为多可用区部署时，必须先完成这些先决条件。如果您未完成先决条件，则多可用区设置将失败。要完成先决条件，请执行将 RDS Custom for SQL Server 的单可用区部署修改为多可用区部署中的步骤。

- 更新 RDS 安全组入站和出站规则以允许端口 1120。
- 在您的私有网络访问控制列表（ACL）中添加一条规则，以允许数据库实例 VPC 使用 TCP 端口 `0-65535`。
- 创建新的 Amazon SQS VPC 端点，以允许 RDS Custom for SQL Server 数据库实例与 SQS 通信。
- 更新实例配置文件角色中的 SQS 权限。

创建 RDS Custom for SQL Server 多可用区部署

要创建 RDS Custom for SQL Server 多可用区部署，请按照为 Amazon RDS Custom for SQL Server 创建并连接到数据库实例中的步骤操作。

> **Important**
>
> 为了简化设置，我们建议您使用网络设置说明中提供的最新 Amazon CloudFormation 模板文件。有关更多信息，请参阅 使用 Amazon CloudFormation 进行配置。

创建多可用区部署需要几分钟才能完成。

将 RDS Custom for SQL Server 的单可用区部署修改为多可用区部署

您可以将现有的 RDS Custom for SQL Server 数据库实例从单可用区部署修改为多可用区部署。修改数据库实例时，Amazon RDS 会执行多项操作：

- 拍摄主数据库实例的快照。
- 从快照中为备用副本创建新卷。这些卷在后台初始化，并在数据完全初始化后达到最大卷性能。
- 开启主数据库实例与辅助数据库实例之间的同步块级复制。

> **Important**
>
> 我们建议您避免在高峰活动期在生产数据库实例上将 RDS Custom for SQL Server 数据库实例从单可用区部署修改为多可用区部署。

Amazon 使用快照创建备用实例，以避免在从单可用区转换到多可用区时出现停机，但是在转换为多可用区期间和转换后，性能可能会受到影响。对于对写入延迟敏感的工作负载而言，这可能会产生很大的影响。尽管此功能允许快速从快照中还原大型卷，但由于同步复制，该功能可能会导致 I/O 操作的延迟增加。这种延迟可能会影响您的数据库性能。

> **Note**
>
> 如果您在 2024 年 8 月 29 日之前创建了 RDS Custom for SQL Server 数据库实例，请在修改之前修补到最新的次要版本。

- 对于 SQL Server 2019 实例，请将数据库引擎版本升级到 `15.00.4410.1.v1` 或更高版本。
- 对于 SQL Server 2022 实例，请将数据库引擎版本升级到 `16.00.4150.1.v1` 或更高版本。

主题

- 使用 CloudFormation 配置先决条件，以将单可用区部署修改为多可用区部署
- 手动配置先决条件，以将单可用区部署修改为多可用区部署
- 使用 RDS 控制台、Amazon CLI 或 RDS API 进行修改。

使用 CloudFormation 配置先决条件，以将单可用区部署修改为多可用区部署

要使用多可用区部署，必须确保应用了满足先决条件的最新 CloudFormation 模板，或者手动配置最新的先决条件。如果您已经应用了最新的 CloudFormation 先决条件模板，则可以跳过这些步骤。

使用 CloudFormation 配置 RDS Custom for SQL Server 多可用区部署先决条件

1. 通过以下网址打开 CloudFormation 控制台：https://console.aws.amazon.com/cloudformation。
2. 要启动创建堆栈向导，请选择用于创建单可用区部署的现有堆栈，然后选择更新。

   此时会显示更新堆栈页面。
3. 对于先决条件 - 准备模板，请选择替换当前模板。
4. 对于 Select Template（选择模板），执行以下操作：
   a. 下载最新的 Amazon CloudFormation 模板文件。打开链接 custom-sqlserver-onboard.zip 的上下文（右键单击）菜单，然后选择 Save Link As（将链接另存为）。
   b. 将 `custom-sqlserver-onboard.json` 文件保存并提取到您的计算机中。
   c. 对于 Template source(模板来源)，选择 Upload a template file(上载模板文件)。
   d. 对于 Choose file（选择文件），请导航到 `custom-sqlserver-onboard.json` 并选择它。
5. 选择下一步。

   此时将出现 Specify stack details（指定堆栈详细信息）页面。
6. 要保留原定设置选项，请选择 Next（下一步)。

将出现高级选项页面。

7. 要保留原定设置选项，请选择 Next（下一步)。
8. 要保留原定设置选项，请选择 Next（下一步)。
9. 在检查更改页面上，执行以下操作：

    a. 对于 Capabilities（功能），请选中 I acknowledge that Amazon CloudFormation might create IAM resources with custom names（我确认 Amazon CloudFormation 可能使用自定义名称创建 IAM 资源）复选框。
    b. 选择提交。

10. 验证更新是否成功。成功操作的状态将显示 `UPDATE_COMPLETE`。

如果更新失败，则将回滚更新过程中指定的任何新配置。现有资源将仍然可用。例如，如果添加编号为 18 和 19 的网络 ACL 规则，但存在编号相同的现有规则，则更新将返回以下错误：`Resource handler returned message: "The network acl entry identified by 18 already exists.`。在这种情况下，您可以修改现有 ACL 规则以使用小于 18 的编号，然后重试更新。

手动配置先决条件，以将单可用区部署修改为多可用区部署

> **Important**
>
> 为了简化设置，我们建议您使用网络设置说明中提供的最新 Amazon CloudFormation 模板文件。有关更多信息，请参阅 [使用 CloudFormation 配置先决条件，以将单可用区部署修改为多可用区部署](#)。

如果您选择手动配置先决条件，请执行以下任务。

1. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
2. 选择端点。创建端点页面显示。
3. 对于服务类别，选择 Amazon 服务。
4. 在服务中，搜索 *`SQS`*。
5. 在 VPC 中，选择在其中部署 RDS Custom for SQL Server 数据库实例的 VPC。
6. 在子网中，选择在其中部署 RDS Custom for SQL Server 数据库实例的子网。
7. 在安全组中，选择 *`-vpc-endpoint-sg`* 组。
8. 对于策略，选择自定义。

9. 在自定义策略中，将 *Amazon partition*、*Region*、*accountId* 和 *IAM-Instance-role* 替换为您自己的值。

```
{
        "Version": "2012-10-17",
        "Statement": [
            {
                "Condition": {
                    "StringLike": {
                        "aws:ResourceTag/AWSRDSCustom": "custom-sqlserver"
                    }
                },
                "Action": [
                    "SQS:SendMessage",
                    "SQS:ReceiveMessage",
                    "SQS:DeleteMessage",
                    "SQS:GetQueueUrl"
                ],
                "Resource": "arn:${AWS::Partition}:sqs:${AWS::Region}:
${AWS::AccountId}:do-not-delete-rds-custom-*",
                "Effect": "Allow",
                "Principal": {
                    "AWS": "arn:${AWS::Partition}:iam::${AWS::AccountId}:role/{IAM-
Instance-role}"
                }
            }
        ]
    }
```

10. 使用访问 Amazon SQS 的权限更新实例配置文件。将 *Amazon partition*、*Region* 和 *accountId* 替换为您自己的值。

```
{
    "Sid": "SendMessageToSQSQueue",
    "Effect": "Allow",
    "Action": [
      "SQS:SendMessage",
      "SQS:ReceiveMessage",
      "SQS:DeleteMessage",
      "SQS:GetQueueUrl"
```

```
    ],
    "Resource": [
      {
        "Fn::Sub": "arn:${AWS::Partition}:sqs:${AWS::Region}:${AWS::AccountId}:do-
not-delete-rds-custom-*"
      }
    ],
    "Condition": {
      "StringLike": {
        "aws:ResourceTag/AWSRDSCustom": "custom-sqlserver"
      }
    }
  }
                      >
```

11. 更新 Amazon RDS 安全组的入站和出站规则以允许端口 1120。

    a. 在安全组中，选择 *-rds-custom-instance-sg* 组。

    b. 对于入站规则，创建自定义 TCP 规则，以允许源 *-rds-custom-instance-sg* 组使用端口 *1120*。

    c. 对于出站规则，创建自定义 TCP 规则，以允许端口 *1120* 连接到目标 *-rds-custom-instance-sg* 组。

12. 在您的私有网络访问控制列表（ACL）中添加一条规则，以允许数据库实例的源子网使用 TCP 端口 `0-65535`。

> **Note**
>
> 创建入站规则和出站规则时，请记下现有的最大规则编号。您创建的新规则的规则编号必须小于 100，且与任何现有规则编号均不匹配。

    a. 在网络 ACL 中，选择 *-private-network-acl* 组。

    b. 对于入站规则，创建全部 TCP 规则，以允许将 TCP 端口 `0-65535` 用于来自 *privatesubnet1* 和 *privatesubnet2* 的源。

    c. 对于出站规则，创建全部 TCP 规则，以允许 TCP 端口 `0-65535` 连接到目标 *privatesubnet1* 和 *privatesubnet2*。

使用 RDS 控制台、Amazon CLI 或 RDS API 进行修改。

完成先决条件后，您可以使用 RDS 控制台、Amazon CLI 或 RDS API，将 RDS Custom for SQL Server 数据库实例从单可用区部署修改为多可用区部署。

控制台

将现有 RDS Custom for SQL Server 单可用区部署修改为多可用区部署

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在 Amazon RDS 控制台中，选择数据库。

   随后会显示数据库窗格。
3. 选择要修改的 RDS Custom for SQL Server 数据库实例。
4. 对于操作中，选择转换为多可用区部署。
5. 在确认页面上，选择立即应用以立即应用更改。选择此选项不会导致停机，但可能会对性能产生影响。或者，您可以选择在下一个维护时段内应用更新。有关更多信息，请参阅 使用计划修改设置。
6. 在确认页面上，选择转换为多可用区。

Amazon CLI

要使用 Amazon CLI 转换为多可用区数据库实例部署，请调用 modify-db-instance 命令并设置 `--multi-az` 选项。指定数据库实例标识符以及要修改的其他选项的值。有关各选项的信息，请参阅 数据库实例的设置。

Example

以下代码通过包含 `--multi-az` 选项来修改 `mycustomdbinstance`。将在下一维护时段使用 `--no-apply-immediately` 应用这些更改。使用 `--apply-immediately` 可立即应用更改。有关更多信息，请参阅 使用计划修改设置。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier mycustomdbinstance \
    --multi-az \
    --no-apply-immediately
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier mycustomdbinstance ^
    --multi-az  \ ^
    --no-apply-immediately
```

RDS API

要使用 RDS API 转换为多可用区数据库实例部署，请调用 ModifyDBInstance 操作并将 `MultiAZ` 参数设置为 true。

将 RDS Custom for SQL Server 多可用区部署修改为单可用区部署

您可以将现有的 RDS Custom for SQL Server 数据库实例从多可用区部署修改为单可用区部署。

控制台

将 RDS Custom for SQL Server 数据库实例从多可用区部署修改为单可用区部署。

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在 Amazon RDS 控制台中，选择数据库。

   随后会显示数据库窗格。
3. 选择要修改的 RDS Custom for SQL Server 数据库实例。
4. 对于多可用区部署，请选择否。
5. 在确认页面上，选择立即应用以立即应用更改。选择此选项不会导致停机，但可能会对性能产生影响。或者，您可以选择在下一个维护时段内应用更新。有关更多信息，请参阅 使用计划修改设置。
6. 在确认页面上，选择修改数据库实例。

Amazon CLI

要使用 Amazon CLI 将多可用区部署修改为单可用区部署，请调用 modify-db-instance 命令并包含 `--no-multi-az` 选项。指定数据库实例标识符以及要修改的其他选项的值。有关各选项的信息，请参阅 数据库实例的设置。

Example

以下代码通过包含 `--no-multi-az` 选项来修改 `mycustomdbinstance`。将在下一维护时段使用 `--no-apply-immediately` 应用这些更改。使用 `--apply-immediately` 可立即应用更改。有关更多信息，请参阅 使用计划修改设置。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier mycustomdbinstance \
    --no-multi-az  \
    --no-apply-immediately
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier mycustomdbinstance ^
    --no-multi-az \ ^
    --no-apply-immediately
```

RDS API

要使用 RDS API 将多可用区部署修改为单可用区部署，请调用 ModifyDBInstance 操作并将参数 `MultiAZ` 设置为 `false`。

RDS Custom for SQL Server 多可用区部署的失效转移过程

如果由于基础设施缺陷而导致数据库实例发生计划内或计划外的中断时，此时如果您已启用多可用区，则 Amazon RDS 会自动切换到另一个可用区中的备用副本。完成失效转移所用的时间取决于在主数据库实例变为不可用时的数据库活动和其他条件。失效转移时间通常为 60 – 120 秒。不过，事务较多或时间较长的恢复过程可能延长故障转移时间。完成失效转移后，RDS 控制台还需要一段时间才能显示新的可用区。

> **Note**
>
> 在通过失效转移重启数据库实例时，可以手动强制执行失效转移。有关重启数据库实例的更多信息，请参阅重启中的数据库实例。

Amazon RDS 会自动处理故障转移，因此，您可以尽快恢复数据库操作而无需管理干预。如果出现下表中描述的任一情况，主数据库实例会自动切换到备用副本：您可以在 RDS 事件日志中查看这些失效转移原因。

故障转移原因	描述
The operating system for the RDS Custom for SQL Server Multi-AZ DB instance is being patched in an offline operation	在操作系统补丁或安全更新的维护时段内触发了故障切换。有关更多信息，请参阅 维护数据库实例。
The primary host of the RDS Custom for SQL Server Multi-AZ DB instance is unhealthy.	多可用区数据库实例部署检测到受损的主数据库实例并进行故障转移。
The primary host of the RDS Custom for SQL Server Multi-AZ DB instance is unreachable due to loss of network connectivity.	RDS 监控检测到主数据库实例的网络可达性故障并触发了故障转移。
The RDS Custom for SQL Server Multi-AZ DB instance was modified by the customer.	数据库实例修改触发了失效转移。有关更多信息，请参阅 修改 RDS Custom for SQL Server 数据库实例。
The storage volume of the primary host of the RDS Custom for SQL Server	多可用区数据库实例部署在主数据库实例上检测到存储问题并进行故障转移。

故障转移原因	描述
Multi-AZ DB instance experienced a failure.	
The user requested a failover of the RDS Custom for SQL Server Multi-AZ DB instance.	RDS Custom for SQL Server 多可用区数据库实例已通过失效转移重启。有关更多信息，请参阅 重启中的数据库实例。
The RDS Custom for SQL Server Multi-AZ primary DB instance is busy or unresponsive.	主数据库实例没有响应。我们建议您尝试以下步骤： • 检查事件日志和 CloudWatch 日志是否存在 CPU、内存或交换空间使用过量的情况。有关更多信息，请参阅 使用 Amazon RDS 事件通知。 • 创建对于 Amazon RDS 事件触发的规则。有关更多信息，请参阅 创建对 Amazon RDS 事件触发的规则。 • 评估您的工作负载，以确定是否使用了适当的数据库实例类。有关更多信息，请参阅 数据库实例类。

要确定多可用区数据库实例是否发生故障转移，您可以执行以下操作：

- 将数据库事件订阅设置为在故障转移启动时向您发送电子邮件或 SMS 通知。有关事件的更多信息，请参阅 使用 Amazon RDS 事件通知。
- 使用 RDS 控制台或 API 操作查看数据库事件。
- 使用 RDS 控制台、CLI 或 API 操作查看 RDS Custom for SQL Server 多可用区数据库实例部署的当前状态。

使用 RDS Custom for SQL Server 多可用区部署的应用程序的存活时间（TTL）设置

故障转移机制自动更改数据库实例的域名系统 (DNS) 记录，使其指向备用数据库实例。因此，您需要重新建立与数据库实例之间的所有现有连接。确保任何 DNS 缓存存活时间（TTL）配置值都较低，并验证您的应用程序不会长时间缓存 DNS。TTL 值较高可能会阻止应用程序在失效转移后快速重新连接到数据库实例。

备份和还原 Amazon RDS Custom for SQL Server 数据库实例

与 Amazon RDS 一样，RDS Custom 会在启用备份保留时创建并保存 RDS Custom for SQL Server 数据库实例的自动备份。您还可以手动备份数据库实例。自动备份由快照备份和事务日志备份组成。在您指定的备份时段内，将对数据库实例的整个存储卷进行快照备份。按固定间隔为符合 PITR 条件的数据库进行事务日志备份。RDS Custom 根据您指定的备份保留期保存数据库实例的自动备份。您可以使用自动备份将数据库实例恢复到备份保留期内的某个时间点。

您也可以手动进行快照备份。您可以随时从这些快照备份创建新的数据库实例。有关手动创建数据库快照的更多信息，请参阅创建 RDS Custom for SQL Server 快照。

尽管快照备份在操作上用作完整备份，但您只需要为增量存储用量付费。RDS Custom 数据库实例的第一个快照包含完整数据库实例的数据。相同数据库的后续快照为增量快照，这意味着仅保存在最新快照后更改的数据。

主题

创建 RDS Custom for SQL Server 快照

RDS Custom for SQL Server 可创建数据库实例的存储卷快照，并备份整个数据库实例而不仅仅是单个数据库。创建快照时，请指定要备份哪个 RDS Custom for SQL Server 数据库实例。为您的快照命名，以便稍后通过它进行还原。

当您创建快照时，RDS Custom for SQL Server 会为卷 (`D:`) 创建 Amazon EBS 快照，该卷是附加到数据库实例的数据库卷。为了使快照轻松与特定数据库实例关联，请使用 `DBSnapshotIdentifier`、`DbiResourceId` 和 `VolumeType` 标记它们。

创建数据库快照会导致短暂的输入/输出暂停。此暂停可持续数秒到几分钟，具体取决于数据库实例的大小和类。快照创建时间因数据库的总计数和大小而异。要详细了解符合条件可执行时间点还原（PITR）操作的数据库数量，请参阅每种实例类类型符合 PITR 条件的数据库数量。

由于快照包含整个存储卷，因此，文件（如临时文件）的大小也会影响创建快照创建时间。要了解创建快照的更多信息，请参阅 为 Amazon RDS 的单可用区数据库实例创建数据库快照。

使用控制台或 Amazon CLI 创建 RDS Custom for SQL Server 快照。

控制台

创建 RDS Custom 快照

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases（数据库）。
3. 在 RDS Custom 数据库实例列表中，选择您想拍摄快照的实例。
4. 对于操作，选择拍摄快照。

   将显示 Take DB snapshot (拍摄数据库快照) 窗口。
5. 在 Snapshot name（快照名称）框中，请输入快照的名称。
6. 选择拍摄快照。

Amazon CLI

您可以通过使用 create-db-snapshot Amazon CLI 命令创建 RDS Custom 数据库实例的快照。

指定以下选项：

- `--db-instance-identifier` – 标识要备份的 RDS Custom 数据库实例
- `--db-snapshot-identifier` – 命名 RDS Custom 快照，以便以后通过它进行还原

在此示例中，您将给名为 *my-custom-instance* 的 RDS Custom 数据库实例创建名为 *my-custom-snapshot* 的数据库快照。

Example

对于 Linux、macOS 或 Unix：

```
aws rds create-db-snapshot \
    --db-instance-identifier my-custom-instance \
    --db-snapshot-identifier my-custom-snapshot
```

对于 Windows：

```
aws rds create-db-snapshot ^
```

```
    --db-instance-identifier my-custom-instance ^
    --db-snapshot-identifier my-custom-snapshot
```

从 RDS Custom for SQL Server 数据库快照还原

在还原 RDS Custom for SQL Server 数据库实例时，需要提供数据库快照的名称以及新实例的名称。您不能从快照还原到现有的 RDS Custom 数据库实例。还原时将新建一个 RDS Custom for SQL Server 数据库实例。

从快照还原会将存储卷还原到拍摄快照的时间点。这将包括 (`D:`) 卷上存在的所有数据库和任何其它文件。

控制台

要从数据库快照还原 RDS Custom 数据库实例

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择快照。
3. 选择要从其还原的数据库快照。
4. 对于操作，选择还原快照。
5. 请在 Restore DB instance（还原数据库实例）页面上，为 DB instance identifier（数据库实例标识符）输入还原的 RDS Custom 数据库实例的名称。
6. 选择还原数据库实例。

Amazon CLI

您可以通过使用 restore-db-instance-from-db-snapshot Amazon CLI 命令还原 RDS Custom 数据库快照。

如果要从中还原的快照适用于私有数据库实例，请务必同时指定正确的 `db-subnet-group-name` 和 `no-publicly-accessible`。否则，数据库实例原定设置为可公开访问。以下选项为必填：

- `db-snapshot-identifier` – 标识要从中进行还原的快照
- `db-instance-identifier` – 指定要从数据库快照创建的 RDS Custom 数据库实例的名称
- `custom-iam-instance-profile` – 指定与 RDS Custom 数据库实例的基础 Amazon EC2 实例关联的实例配置文件。

以下代码为 my-custom-instance 还原名为 my-custom-snapshot 的快照。

Example

对于 Linux、macOS 或 Unix：

```
aws rds restore-db-instance-from-db-snapshot \
  --db-snapshot-identifier my-custom-snapshot \
  --db-instance-identifier my-custom-instance \
  --custom-iam-instance-profile AWSRDSCustomInstanceProfileForRdsCustomInstance \
  --no-publicly-accessible
```

对于 Windows：

```
aws rds restore-db-instance-from-db-snapshot ^
  --db-snapshot-identifier my-custom-snapshot ^
  --db-instance-identifier my-custom-instance ^
  --custom-iam-instance-profile AWSRDSCustomInstanceProfileForRdsCustomInstance ^
  --no-publicly-accessible
```

将 RDS Custom for SQL Server 实例还原到某个时间点

您可以将数据库实例还原到某个特定时间点 (PITR)，以创建新数据库实例。要支持 PITR，您的数据库实例必须已启用备份保留期。

RDS Custom for SQL Server 数据库实例的最近可还原时间取决于多个因素，但通常为当前时间的 5 分钟内。要查看某个数据库实例的最近可还原时间，请使用 Amazon CLI describe-db-instances 命令，并查看该数据库实例的 LatestRestorableTime 字段中返回的值。要在 Amazon RDS 控制台中查看每个数据库实例的最近可还原时间，请选择自动备份。

您可以还原至备份保留期内的任何时间点。要查看每个数据库实例的最早可还原时间，请在 Amazon RDS 控制台中选择自动备份。

有关 PITR 的一般信息，请参阅将 Amazon RDS 的数据库实例还原到指定时间。

主题

- RDS Custom for SQL Server 的 PITR 注意事项
- 每种实例类类型符合 PITR 条件的数据库数量
- 使数据库不符合 PITR 资格

- Amazon S3 中的事务日志
- 使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 进行 PITR 还原。

RDS Custom for SQL Server 的 PITR 注意事项

在 RDS Custom for SQL Server 中，PITR 与 Amazon RDS 中的 PITR 有以下重要的区别：

- PITR 仅还原数据库实例中的数据库。它不会还原操作系统或 C: 驱动器上的文件。
- 对于 RDS Custom for SQL Server 数据库实例，数据库将自动备份，并且只有在以下条件下才有资格用于 PITR：
  - 数据库处于联机状态。
  - 它的还原模式设置为 `FULL`。
  - 它是可写的。
  - 它的物理文件放在 D: 驱动器上。
  - 它未在 `rds_pitr_blocked_databases` 表中列出。有关更多信息，请参阅 使数据库不符合 PITR 资格。
- 这些符合 PITR 条件的数据库由其数据库 ID 的顺序决定。RDS Custom for SQL Server 允许每个数据库实例最多 5000 个数据库。不过，针对 RDS Custom for SQL Server 数据库实例，PITR 操作还原的数据库的最大数量取决于实例类的类型。有关更多信息，请参阅 每种实例类类型符合 PITR 条件的数据库数量。

  其它不属于 PITR 的数据库可以从数据库快照中还原，包括用于 PITR 的自动快照备份。
- 添加新数据库、重命名数据库或还原符合 PITR 条件的数据库将启动数据库实例的快照。
- 当数据库实例进行扩展计算操作时，符合 PITR 条件的最大数据库数量会发生变化，具体取决于目标实例类的类型。如果实例已纵向扩展，同时允许该实例上的更多数据库符合 PITR 条件，则会拍摄新的快照。
- 还原的数据库的名称与源数据库实例中的名称相同。您不能指定不同的名称。
- `AWSRDSCustomSQLServerIamRolePolicy` 需要访问其它 Amazon 服务。有关更多信息，请参阅 将访问策略添加到 AWSRDSCustomSQLServerInstanceRole。
- RDS Custom for SQL Server 不支持时区更改。如果更改操作系统或数据库实例时区，PITR（和其他自动化）将无法正常工作。

每种实例类类型符合 PITR 条件的数据库数量

下表显示了根据实例类的类型符合 PITR 条件的最大数据库数量。

实例类类型	符合 PITR 条件的数据库的最大数量				
db.*.large	100				
db.*.xlarge 至 db.*.2xlarge	150				
db.*.4xlarge 至 db.*.8xlarge	300				
db.*.12xlarge 至 db.*.16xlarge	600				
db.*.24xlarge、db.*32xlarge	1000				

* 表示不同实例类类型。

数据库实例上符合 PITR 条件的最大数据库数量取决于实例类类型。该数字的范围从最小的 100 到 RDS Custom for SQL Server 支持的最大实例类类型上的 1000 不等。此限制中并不包括 Server 系统数据库（`master`, `model`, `msdb`, `tempdb`）。当数据库实例纵向扩展或缩减时，根据目标实例类类型，RDS Custom 将自动更新符合 PITR 条件的数据库数量。当数据库实例上符合 PITR 条件的最大数据库数量发生更改时，RDS Custom for SQL Server 将发送 `RDS-EVENT-0352`。有关更多信息，请参阅 自定义引擎版本事件。

> **Note**
>
> 对超过 100 个数据库的 PITR 支持仅在 2023 年 8 月 26 日之后创建的数据库实例上可用。对于 2023 年 8 月 26 日之前创建的实例，无论实例类如何，符合 PITR 条件的最大数据库数量均为 100。要在 2023 年 8 月 26 日之前创建的数据库实例上启用对超过 100 个数据库的 PITR 支持，您可以执行以下操作：

- 将数据库引擎版本升级到 15.00.4322.2.v1 或更高版本

在 PITR 操作期间，RDS Custom 将在还原时在源数据库实例上还原属于 PITR 的所有数据库。目标数据库实例完成还原操作后，如果启用了备份保留，数据库实例将根据目标数据库实例上符合 PITR 条件的最大数据库数量开始备份。

例如，如果您的数据库实例在具有 200 个数据库的 `db.*.xlarge` 上运行：

1. RDS Custom for SQL Server 将选择前 150 个数据库进行 PITR 备份，按其数据库 ID 排序。
2. 您可以修改实例以纵向扩展到 db.*.4xlarge。
3. 扩展计算操作完成后，RDS Custom for SQL Server 将选择前 300 个数据库进行 PITR 备份，按其数据库 ID 排序。现在，满足 PITR 要求条件的 200 个数据库中的每一个都符合 PITR 条件。
4. 现在，您可以修改实例以缩减回到 db.*.xlarge。
5. 扩展计算操作完成后，RDS Custom for SQL Server 将再次选择前 150 个数据库进行 PITR 备份，按其数据库 ID 排序。

使数据库不符合 PITR 资格

您可以选择从 PITR 中排除各个数据库。要做到这一点，请将其 `database_id` 值放入 `rds_pitr_blocked_databases` 表中。使用以下 SQL 脚本创建表。

要创建 rds_pitr_blocked_databases 表

- 请运行以下 SQL 脚本。

```
create table msdb..rds_pitr_blocked_databases
(
database_id INT NOT NULL,
database_name SYSNAME NOT NULL,
db_entry_updated_date datetime NOT NULL DEFAULT GETDATE(),
db_entry_updated_by SYSNAME NOT NULL DEFAULT CURRENT_USER,
PRIMARY KEY (database_id)
);
```

有关符合条件和不符合条件的数据库的列表，请参阅 Amazon S3 存储桶 `do-not-delete-rds-custom-`*`$ACCOUNT_ID`*`-`*`$REGION`*`-`*`unique_identifier`* 中的 `RDSCustomForSQLServer/`

Instances/*DB_instance_resource_ID*/TransactionLogMetadata 目录中的 RI.End 文件。有关 RI.End 文件的更多信息，请参阅Amazon S3 中的事务日志。

您还可以使用以下 SQL 脚本确定符合 PITR 条件的数据库列表。将 @limit 变量设置为符合该实例类的 PITR 条件的最大数据库数量。有关更多信息，请参阅 每种实例类类型符合 PITR 条件的数据库数量。

在数据库实例类上确定符合 PITR 条件的数据库列表

- 请运行以下 SQL 脚本。

```
DECLARE @Limit INT;
SET @Limit = (insert-database-instance-limit-here);

USE msdb;
IF (EXISTS (SELECT * FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_SCHEMA = 'dbo' AND
 TABLE_NAME = 'rds_pitr_blocked_databases'))
    WITH TABLE0 AS (
        SELECT hdrs.database_id as DatabaseId, sdb.name as DatabaseName,
 'ALWAYS_ON_NOT_WRITABLE_REPLICA' as Reason, NULL as DatabaseNameOnPitrTable
        FROM sys.dm_hadr_database_replica_states hdrs
        INNER JOIN sys.databases sdb ON sdb.database_id = hdrs.database_id
        WHERE (hdrs.is_local = 1 AND hdrs.is_primary_replica = 0)
        OR (sys.fn_hadr_is_primary_replica (sdb.name) = 1 AND DATABASEPROPERTYEX
 (sdb.name, 'Updateability') = 'READ_ONLY')
    ),
    TABLE1 as (
            SELECT dbs.database_id as DatabaseId, sysdbs.name as DatabaseName,
 'OPTOUT' as Reason,
            CASE WHEN dbs.database_name = sysdbs.name THEN NULL ELSE
 dbs.database_name END AS DatabaseNameOnPitrTable
            FROM msdb.dbo.rds_pitr_blocked_databases dbs
            INNER JOIN sys.databases sysdbs ON dbs.database_id = sysdbs.database_id
            WHERE sysdbs.database_id > 4
            ),
    TABLE2 as (
            SELECT
            db.name AS DatabaseName,
            db.create_date AS CreateDate,
            db.state_desc AS DatabaseState,
            db.database_id AS DatabaseId,
            rs.database_guid AS DatabaseGuid,
            rs.last_log_backup_lsn AS LastLogBackupLSN,
```

```
            rs.recovery_fork_guid RecoveryForkGuid,
            rs.first_recovery_fork_guid AS FirstRecoveryForkGuid,
            db.recovery_model_desc AS RecoveryModel,
            db.is_auto_close_on AS IsAutoClose,
            db.is_read_only as IsReadOnly,
            NEWID() as FileName,
            CASE WHEN(db.state_desc = 'ONLINE'
                    AND db.recovery_model_desc != 'SIMPLE'
                    AND((db.is_auto_close_on = 0 and db.collation_name IS NOT NULL)
 OR db.is_auto_close_on = 1))
                    AND db.is_read_only != 1
                    AND db.user_access = 0
                    AND db.source_database_id IS NULL
                    AND db.is_in_standby != 1
                    THEN 1 ELSE 0 END AS IsPartOfSnapshot,
            CASE WHEN db.source_database_id IS NULL THEN 0 ELSE 1 END AS
 IsDatabaseSnapshot
            FROM sys.databases db
            INNER JOIN sys.database_recovery_status rs
            ON db.database_id = rs.database_id
            WHERE DB_NAME(db.database_id) NOT IN('tempdb') AND
            db.database_id NOT IN (SELECT DISTINCT DatabaseId FROM TABLE1) AND
            db.database_id NOT IN (SELECT DISTINCT DatabaseId FROM TABLE0)
        ),
        TABLE3 as(
            Select @Limit+count(DatabaseName) as TotalNumberOfDatabases from TABLE2
 where TABLE2.IsPartOfSnapshot=1 and DatabaseName in ('master','model','msdb')
        )
        SELECT TOP(SELECT TotalNumberOfDatabases from TABLE3)
 DatabaseName,CreateDate,DatabaseState,DatabaseId from TABLE2 where
 TABLE2.IsPartOfSnapshot=1
        ORDER BY TABLE2.DatabaseID ASC
ELSE
    WITH TABLE0 AS (
        SELECT hdrs.database_id as DatabaseId, sdb.name as DatabaseName,
 'ALWAYS_ON_NOT_WRITABLE_REPLICA' as Reason, NULL as DatabaseNameOnPitrTable
        FROM sys.dm_hadr_database_replica_states hdrs
        INNER JOIN sys.databases sdb ON sdb.database_id = hdrs.database_id
        WHERE (hdrs.is_local = 1 AND hdrs.is_primary_replica = 0)
        OR (sys.fn_hadr_is_primary_replica (sdb.name) = 1 AND DATABASEPROPERTYEX
 (sdb.name, 'Updateability') = 'READ_ONLY')
    ),
    TABLE1 as (
            SELECT
```

```
            db.name AS DatabaseName,
            db.create_date AS CreateDate,
            db.state_desc AS DatabaseState,
            db.database_id AS DatabaseId,
            rs.database_guid AS DatabaseGuid,
            rs.last_log_backup_lsn AS LastLogBackupLSN,
            rs.recovery_fork_guid RecoveryForkGuid,
            rs.first_recovery_fork_guid AS FirstRecoveryForkGuid,
            db.recovery_model_desc AS RecoveryModel,
            db.is_auto_close_on AS IsAutoClose,
            db.is_read_only as IsReadOnly,
            NEWID() as FileName,
            CASE WHEN(db.state_desc = 'ONLINE'
                    AND db.recovery_model_desc != 'SIMPLE'
                    AND((db.is_auto_close_on = 0 and db.collation_name IS NOT NULL)
OR db.is_auto_close_on = 1))
                    AND db.is_read_only != 1
                    AND db.user_access = 0
                    AND db.source_database_id IS NULL
                    AND db.is_in_standby != 1
                    THEN 1 ELSE 0 END AS IsPartOfSnapshot,
            CASE WHEN db.source_database_id IS NULL THEN 0 ELSE 1 END AS
IsDatabaseSnapshot
            FROM sys.databases db
            INNER JOIN sys.database_recovery_status rs
            ON db.database_id = rs.database_id
            WHERE DB_NAME(db.database_id) NOT IN('tempdb') AND
            db.database_id NOT IN (SELECT DISTINCT DatabaseId FROM TABLE0)
        ),
        TABLE2 as(
            SELECT @Limit+count(DatabaseName) as TotalNumberOfDatabases from TABLE1
where TABLE1.IsPartOfSnapshot=1 and DatabaseName in ('master','model','msdb')
        )
        select top(select TotalNumberOfDatabases from TABLE2)
DatabaseName,CreateDate,DatabaseState,DatabaseId from TABLE1 where
TABLE1.IsPartOfSnapshot=1
        ORDER BY TABLE1.DatabaseID ASC
```

> **Note**
>
> 仅作为符号链接的数据库也被排除在符合 PITR 操作条件的数据库之外。上述查询不会根据此条件进行筛选。

Amazon S3 中的事务日志

备份保留期决定了 RDS Custom for SQL Server 数据库实例的事务日志是否会自动提取并上传到 Amazon S3。非零值表示创建了自动备份，且 RDS Custom 代理每 5 分钟将事务日志上传到 S3 一次。

S3 上的事务日志文件将使用您在创建数据库实例时提供的 Amazon KMS key 静态加密。有关更多信息，请参阅《Amazon Simple Storage Service 用户指南》中的使用服务器端加密保护数据。

每个数据库的事务日志将被上传到名为 `do-not-delete-rds-custom-$ACCOUNT_ID-$REGION-unique_identifier` 的 S3 存储桶。S3 存储桶中的 `RDSCustomForSQLServer/Instances/DB_instance_resource_ID` 目录包含两个子目录：

- `TransactionLogs` – 包含每个数据库的事务日志及其各自的元数据。

  事务日志文件名称遵循模式 *`yyyyMMddHHmm.database_id.timestamp`*，例如：

  ```
  202110202230.11.1634769287
  ```

  带有后缀 `_metadata` 的相同文件名称包含有关事务日志的信息，例如日志序列号、数据库名称和 `RdsChunkCount`。`RdsChunkCount` 确定有多少个物理文件代表单个事务日志文件。您可能会看到带有后缀 `_0001`、`_0002` 等的文件，它们表示事务日志文件的物理数据块。如果要使用分块的事务日志文件，请确保在下载后合并这些数据块。

  考虑您拥有以下文件的情况：

  - `202110202230.11.1634769287`
  - `202110202230.11.1634769287_0001`
  - `202110202230.11.1634769287_0002`
  - `202110202230.11.1634769287_metadata`

  `RdsChunkCount` 为 `3`。合并文件的顺序如下：`202110202230.11.1634769287`、`202110202230.11.1634769287_0001`、`202110202230.11.1634769287_0002`。

- `TransactionLogMetadata` – 包含有关事务日志提取的每次迭代的元数据信息。

  `RI.End` 文件包含提取了事务日志的所有数据库的信息，以及所有存在但没有提取其事务日志的数据库的信息。`RI.End` 文件名称遵循模式 *yyyyMMddHHmm*`.RI.End.`*timestamp*，例如：

```
202110202230.RI.End.1634769281
```

使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 进行 PITR 还原。

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 将 RDS Custom for SQL Server 数据库实例还原到某个时间点。

控制台

要将 RDS Custom 数据库实例还原至指定时间

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Automated backups（自动备份）。
3. 选择要还原的 RDS Custom 数据库实例。
4. 对于 Actions (操作)，选择 Restore to point in time (还原到时间点)。

   此时会显示还原到时间点窗口。
5. 选择最近可还原时间以还原到可能的最近时间，或选择自定义来选择时间。

   如果您选择 Custom（自定义），请输入要将实例集群还原到的日期和时间。

   时间以您的本地时区显示，表示为协调世界时 (UTC) 的偏移量。例如，UTC-5 是东部标准时间/中部夏令时。
6. 对于 DB instance identifier（数据库实例标识符），请输入目标还原后的 RDS Custom 数据库实例的名称。名称必须唯一。
7. 根据需要选择其他选项，例如数据库实例类。
8. 选择还原到时间点。

Amazon CLI

通过使用 restore-db-instance-to-point-in-time Amazon CLI 命令创建新的 RDS Custom 数据库实例，将数据库实例还原到指定时间。

使用以下选项之一指定要从中恢复的备份：

- `--source-db-instance-identifier` *mysourcedbinstance*
- `--source-dbi-resource-id` *dbinstanceresourceID*
- `--source-db-instance-automated-backups-arn` *backupARN*

`custom-iam-instance-profile` 选项是必需的。

截至指定的时间，以下示例会将 `my-custom-db-instance` 还原到名为 `my-restored-custom-db-instance` 的新数据库实例。

Example

对于 Linux、macOS 或 Unix：

```
aws rds restore-db-instance-to-point-in-time \
    --source-db-instance-identifier my-custom-db-instance\
    --target-db-instance-identifier my-restored-custom-db-instance \
    --custom-iam-instance-profile AWSRDSCustomInstanceProfileForRdsCustomInstance \
    --restore-time 2022-10-14T23:45:00.000Z
```

对于 Windows：

```
aws rds restore-db-instance-to-point-in-time ^
    --source-db-instance-identifier my-custom-db-instance ^
    --target-db-instance-identifier my-restored-custom-db-instance ^
    --custom-iam-instance-profile AWSRDSCustomInstanceProfileForRdsCustomInstance ^
    --restore-time 2022-10-14T23:45:00.000Z
```

删除 RDS Custom for SQL Server 快照

如果不再需要，您可以删除 RDS Custom for SQL Server 管理的数据库快照。对于 Amazon RDS 和 RDS Custom 数据库实例而言，删除过程是相同的。

二进制卷和根卷的 Amazon EBS 快照会在您的账户中保留更长时间，因为它们可能与您账户中运行的某些实例或其他 RDS Custom for SQL Server 快照关联。当这些 EBS 快照不再与任何现有 RDS Custom for SQL Server 资源（数据库实例或备份）相关后，将会被自动删除。

控制台

要删除 RDS Custom 数据库实例的快照

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择快照。
3. 选择要删除的数据库快照。
4. 对于 Actions（操作），选择 Delete snapshot（删除快照）。
5. 在确认页面上选择 Delete (删除)。

Amazon CLI

要删除 RDS Custom 快照，请使用 Amazon CLI 命令 delete-db-snapshot。

以下选项为必填项：

- `--db-snapshot-identifier` – 要删除的快照

以下示例将删除 `my-custom-snapshot` 数据库快照。

Example

对于 Linux、macOS 或 Unix：

```
aws rds delete-db-snapshot \
  --db-snapshot-identifier my-custom-snapshot
```

对于 Windows：

```
aws rds delete-db-snapshot ^
  --db-snapshot-identifier my-custom-snapshot
```

删除 RDS Custom for SQL Server 自动备份

当不再需要 RDS Custom for SQL Server 的保留自动备份时，您可以删除它们。此过程与删除 Amazon RDS 备份的过程相同。

控制台

删除保留的自动备份

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Automated backups（自动备份）。
3. 选择 Retained (保留)。
4. 选择要删除的保留自动备份。
5. 对于 Actions (操作)，选择 Delete (删除)。
6. 在确认页面上，输入 **delete me** 并选择 Delete (删除)。

Amazon CLI

可以通过使用 Amazon CLI 命令 delete-db-instance-automated-backup 删除保留的自动备份。

以下选项用于删除保留的自动备份：

- `--dbi-resource-id` – 源 RDS Custom 数据库实例的资源标识符。

  可以通过使用 Amazon CLI 命令 describe-db-instance-automated-backups 查找保留自动备份的源数据库实例的资源标识符。

以下示例删除具有源数据库实例资源标识符 `custom-db-123ABCEXAMPLE` 的保留自动备份。

Example

对于 Linux、macOS 或 Unix：

```
aws rds delete-db-instance-automated-backup \
    --dbi-resource-id custom-db-123ABCEXAMPLE
```

对于 Windows：

```
aws rds delete-db-instance-automated-backup ^
    --dbi-resource-id custom-db-123ABCEXAMPLE
```

将本地数据库迁移到 Amazon RDS Custom for SQL Server

您可以使用以下过程通过本机备份和还原将本地 Microsoft SQL Server 数据库迁移到 Amazon RDS Custom for SQL Server。

1. 在本地数据库实例上对数据库进行完整备份。
2. 将备份文件上载到 Amazon S3。
3. 将备份文件从 S3 下载到 RDS Custom for SQL Server 数据库实例。
4. 使用 RDS Custom for SQL Server 数据库实例上已下载的备份文件还原数据库。

此过程解释了使用本机完全备份和还原将数据库从本地迁移到 RDS Custom for SQL Server 的情况。为了缩短迁移过程中的割接时间，您也可以考虑使用差异备份或日志备份。

有关 RDS for SQL Server 的本机备份和还原的一般信息，请参阅使用本机备份和还原导入和导出 SQL Server 数据库。

主题

先决条件

在迁移数据库之前，请执行以下任务：

1. 为 RDS Custom for SQL Server 数据库实例配置远程桌面连接 (RDP)。有关更多信息，请参阅 使用 RDP 连接到 RDS Custom 数据库实例。
2. 配置对 Amazon S3 的访问权限，以便您可以上载和下载数据库备份文件。

备份本地数据库

您可以使用 SQL Server 本机备份在本地数据库实例上对数据库进行完整备份。

以下示例显示了称为 mydatabase 的数据库备份，以及指定用于减小备份文件大小的 COMPRESSION 选项。

要备份本地数据库

1. 使用 SQL Server Management Studio (SSMS)，连接到本地 SQL Server 实例。
2. 运行以下 T-SQL 命令。

```
backup database mydatabase to
disk ='C:\Program Files\Microsoft SQL Server\MSSQL13.MSSQLSERVER\MSSQL\Backup\mydb-full-compressed.bak'
with compression;
```

将备份文件上载到 Amazon S3

您可以使用 Amazon Web Services Management Console 将备份文件 mydb-full-compressed.bak 上载到 Amazon S3。

要将备份文件上载到 S3

1. 登录到 Amazon Web Services Management Console，然后通过以下网址打开 Amazon S3 控制台：https://console.aws.amazon.com/s3/。
2. 对于 Buckets（存储桶），选择要将备份文件上载到的存储桶的名称。
3. 请选择 Upload（上传）。
4. 在上传窗口中，执行下列操作之一：
   - 将 mydb-full-compressed.bak 拖放到 Upload（上载）窗口。
   - 选择 Add file（添加文件），选择 mydb-full-compressed.bak，然后选择 Open（打开）。

   Amazon S3 将备份文件上载为 S3 对象。上传完成后，您可以在上传：状态页面上看到成功消息。

从 Amazon S3 下载备份文件

您可以使用控制台将备份文件从 S3 下载到 RDS Custom for SQL Server 数据库实例。

要从 S3 下载备份文件

1. 使用 RDP 连接到 RDS Custom for SQL Server 数据库实例。
2. 登录到 Amazon Web Services Management Console，然后通过以下网址打开 Amazon S3 控制台：https://console.aws.amazon.com/s3/。
3. 在 Buckets（存储桶）列表中，选择包含备份文件的存储桶的名称。
4. 选择备份文件 `mydb-full-compressed.bak`。
5. 对于 Actions（操作），选择 Download as（下载为）。
6. 打开提供的链接的上下文（右键单击）菜单，然后选择 Save As（另存为）。
7. 将 `mydb-full-compressed.bak` 保存到 `D:\rdsdbdata\BACKUP` 目录。

将备份文件还原到 RDS Custom for SQL Server 数据库实例

您可以使用 SQL Server 本机还原将备份文件还原到 RDS Custom for SQL Server 数据库实例。

在此示例中，因为数据和日志文件目录与本地数据库实例不同，将指定 `MOVE` 选项。

要恢复备份文件

1. 使用 SSMS，连接到 RDS Custom for SQL Server 数据库实例。
2. 运行以下 T-SQL 命令。

```
restore database mydatabase from disk='D:\rdsdbdata\BACKUP\mydb-full-
compressed.bak'
with move 'mydatabase' to 'D:\rdsdbdata\DATA\mydatabase.mdf',
move 'mydatabase_log' to 'D:\rdsdbdata\DATA\mydatabase_log.ldf';
```

升级 Amazon RDS Custom for SQL Server 数据库实例

您可以通过修改 Amazon RDS Custom for SQL Server 数据库实例，以使其使用新的数据库引擎版本来升级该实例。有关升级数据库实例的一般信息，请参阅升级数据库实例引擎版本。

主题

RDS Custom for SQL Server 升级概述

Amazon RDS Custom for SQL Server 支持主要版本和次要版本升级。次要版本升级可能包括安全补丁、错误修复和引擎改进。Microsoft 以累积更新（CU）形式发布这些更新。主要版本升级在版本之间引入了新功能和引擎更改，例如从 SQL Server 2019 升级到 2022。您可以立即或在计划的维护时段内应用这两种升级。为防止出现潜在的向后兼容性问题，建议在升级之前，先在非生产环境中测试应用程序。

RDS Custom for SQL Server 可让您升级 RDS 提供的引擎版本（RPEV）或自定义引擎版本（CEV）。

- RDS 提供的引擎版本（RPEV）包含最新的操作系统（OS）补丁和 SQL Server 累积更新（CU）。
- 对于自定义引擎版本（CEV），您必须遵循由两个步骤组成的过程。首先，使用目标 SQL Server 版本创建新的 CEV，请参阅正在准备创建适用于 RDS Custom for SQL Server 的 CEV。此目标版本必须等于或高于您的当前版本。创建新的 CEV 后，修改数据库实例以使用此新版本。有关更多信息，请参阅 Performing a minor version upgrade for Amazon RDS Custom for SQL Server CEV with Multi-AZ。

如果您在多可用区部署中升级 RDS Custom for SQL Server 数据库实例，则 RDS Custom for SQL Server 会对实例执行滚动升级。这种方法通过一次升级一个实例来最大限度地减少停机时间。RDS 执行以下操作来执行滚动升级：

1. 升级备用数据库实例。
2. 失效转移到升级后的备用数据库实例，使其成为新的主数据库实例。
3. 升级新的备用数据库实例。

多可用区部署的数据库实例停机时间是失效转移所花费的时间。

升级 RDS Custom for SQL Server 数据库实例时，以下限制适用：

- 不支持自定义数据库选项和参数组。
- 升级后，将不会连接到您连接到 RDS Custom for SQL Server 数据库实例的任何其它存储卷。

升级主要和次要引擎版本

主要和次要引擎版本升级都是不可逆转的，必须始终升级到较新的版本。要确定可用的目标版本，请在修改数据库实例时使用 Amazon Web Services Management Console并从可用版本中进行选择。或者，使用 describe-db-engine-versions CLI 命令或 DescribeDBEngineVersions RDS API 命令。

对于 Linux、macOS 或 Unix：

```
aws rds describe-db-engine-versions \
    --engine custom-sqlserver-se \
    --engine-version 15.00.4322.2.v1 \
    --query "DBEngineVersions[*].ValidUpgradeTarget[*].{EngineVersion:EngineVersion}" \
    --output table
```

对于 Windows：

```
aws rds describe-db-engine-versions ^
    --engine custom-sqlserver-se ^
    --engine-version 15.00.4322.2.v1 ^
    --query "DBEngineVersions[*].ValidUpgradeTarget[*].{EngineVersion:EngineVersion}" ^
    --output table
```

输出显示可用的目标引擎版本：

```
--------------------------
|DescribeDBEngineVersions|
+------------------------+
|      EngineVersion     |
+------------------------+
|   15.00.4410.1.v1      |
|   15.00.4415.2.v1      |
|   15.00.4430.1.v1      |
```

```
|   16.00.4165.4.v1        |
|   16.00.4175.1.v1        |
|   16.00.4185.3.v1        |
+--------------------------+
```

确定目标版本后，使用 Amazon Web Services Management Console并按照修改 RDS Custom for SQL Server 数据库实例中的说明操作。或者，也可以使用 `modify-db-instance` CLI 命令或 ModifyDBInstance RDS API 命令。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier DB_INSTANCE_IDENTIFIER \
    --engine-version ENGINE_VERSION \
    --allow-major-version-upgrade \
    --region Region \
    --no-apply-immediately
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier DB_INSTANCE_IDENTIFIER ^
    --engine-version ENGINE_VERSION ^
    --allow-major-version-upgrade ^
    --region Region ^
    --no-apply-immediately
```

> **Note**
>
> 必须包含 `--allow-major-version-upgrade` 参数才能执行主要版本升级。

数据库兼容性级别

您可以使用 Microsoft SQL Server 数据库兼容性级别调整某些数据库行为，以模仿之前版本的 SQL Server。有关更多信息，请参阅 Microsoft 文档中的兼容性级别。

在升级数据库实例时，任何现有的数据库将保持其原始兼容性级别。例如，如果从 SQL Server 2019 升级到 SQL Server 2022，所有现有数据库的兼容性级别均为 150。在升级后创建的所有新数据库的兼容性级别均为 160。

您可以使用 ALTER DATABASE 命令更改数据库的兼容性级别。例如，要将名为 `customeracct` 的数据库更改为与 SQL Server 2022 兼容，请发出以下命令：

```
ALTER DATABASE customeracct SET COMPATIBILITY_LEVEL = 160
```

针对 Amazon RDS Custom for SQL Server 排查数据库的问题

RDS Custom 的责任共担模型提供了操作系统外壳级别的访问权限和数据库管理员访问权限。RDS Custom 会在您的账户中运行资源，与 Amazon RDS 不同，因为后者会在系统账户中运行资源。访问权限更多，责任也更大。在下面的各个部分，您可以了解如何排查 Amazon RDS Custom for SQL Server 数据库实例的问题。

> **Note**
>
> 本节介绍如何排查 RDS Custom for SQL Server 的问题。有关排查 RDS Custom for Oracle 的问题，请参阅针对 Amazon RDS Custom for Oracle 排查数据库问题。

主题

- 查看 RDS Custom 事件
- 订阅 RDS Custom 事件
- RDS Custom for SQL Server 的 CEV 错误故障排除
- 修复 RDS Custom for SQL Server 中不支持的配置
- 在 RDS Custom for SQL Server 中排查 Storage-Full 问题
- 对 RDS Custom for SQL Server 中启用 TDE 的数据库的 PENDING_RECOVERY 状态进行故障排除

查看 RDS Custom 事件

对于 RDS Custom 和 Amazon RDS 数据库实例而言，查看事件的流程是相同的。有关更多信息，请参阅查看 Amazon RDS 事件。

要使用 Amazon CLI 查看 RDS Custom 事件通知，请使用 `describe-events` 命令。RDS Custom 推出了几项新事件。事件类别与 Amazon RDS 的事件类别相同。有关事件列表，请参阅 Amazon RDS 事件类别和事件消息。

以下示例检索指定 RDS Custom 数据库实例发生的事件的详细信息。

```
aws rds describe-events \
    --source-identifier my-custom-instance \
    --source-type db-instance
```

订阅 RDS Custom 事件

对于 RDS Custom 和 Amazon RDS 数据库实例而言，订阅事件的流程是相同的。有关更多信息，请参阅订阅 Amazon RDS 事件通知。

要使用 CLI 订阅 RDS Custom 事件通知，请使用 `create-event-subscription` 命令。包括以下必需参数：

- `--subscription-name`
- `--sns-topic-arn`

以下示例为当前的 Amazon 账户中的 RDS Custom 数据库实例创建了备份和恢复事件的订阅。您也可以将通知发送到 `--sns-topic-arn` 指定的 Amazon Simple Notification Service (Amazon SNS) 主题。

```
aws rds create-event-subscription \
    --subscription-name my-instance-events \
    --source-type db-instance \
    --event-categories '["backup","recovery"]' \
    --sns-topic-arn arn:aws:sns:us-east-1:123456789012:interesting-events
```

RDS Custom for SQL Server 的 CEV 错误故障排除

当您尝试创建 CEV 时，可能会失败。在这种情况下，RDS Custom 会发出 `RDS-EVENT-0198` 事件消息。有关查看 RDS 事件的更多信息，请参阅 Amazon RDS 事件类别和事件消息。

以下信息可帮助您解决可能的原因。

消息	故障排除建议		
`Custom Engine Version creation expected a Sysprep'd AMI. Retry creation using a Sysprep'd AMI.`	在您从 AMI 创建的 EC2 实例上运行 Sysprep。有关使用 Sysprep 准备 AMI 的更多信息，请参阅使用 Sysprep 创建标准化的 Amazon 机器映像（AMI）。		
`EC2 Image permissions for image (AMI_ID) weren't found for customer`	验证您用于创建的账户和配置文件对所选 AMI 的 `create EC2`		

消息	故障排除建议
(Customer_ID). Verify customer (Customer_ID) has valid permissions on the EC2 Image.	Instance 和 Describe Images 具有所需的权限。
Failed to rebuild databases with server collation (collation name) due to missing setup.exe file for SQL Server.	确认 setup 文件在 C:\Program Files\Microsoft SQL Server\nnn\Setup Bootstrap \SQLnnnn\setup.exe 中可用。
Image (AMI_ID) doesn't exist in your account (ACCOUNT_ID). Verify (ACCOUNT_ID) is the owner of the EC2 image.	确保 AMI 存在于同一个客户账户中。
Image id (AMI_ID) isn't valid. Specify a valid image id, and try again.	AMI 的名称不正确。确保提供正确的 AMI ID。
Image (AMI_ID) operating system platform isn't supported. Specify a valid image, and try again.	选择支持的 AMI，该 AMI 具有 Windows Server 以及 SQL Server Enterprise、Standard 或 Web 版。从 EC2 Marketplace 选择具有以下使用情况操作代码之一的 AMI： • RunInstances:0102 - 含有 SQL Server Enterprise 的 Windows • RunInstances:0006 - 含有 SQL Server Standard 的 Windows • RunInstances:0202 - 有 SQL Server Web 的 Windows

消息	故障排除建议
`SQL Server Web Edition isn't supported for creating a Custom Engine Version using Bring Your Own Media. Specify a valid image, and try again.`	使用包含支持的 SQL Server 版本的 AMI。有关更多信息，请参阅 RDS Custom for SQL Server CEV 的版本支持。
`The custom engine version can't be the same as the OEV engine version. Specify a valid CEV, and try again.`	不支持经典 RDS Custom for SQL Server 引擎版本。例如，版本 15.00.4073.23.v1。使用支持的版本号。
`The custom engine version isn't in an active state. Specify a valid CEV, and try again.`	CEV 必须处于 `AVAILABLE` 状态才能完成操作。将 CEV 从 `INACTIVE` 修改为 `AVAILABLE` 。
`The custom engine version isn't valid for an upgrade. Specify a valid CEV with an engine version greater or equal to (X), and try again.`	目标 CEV 无效。检查对有效升级路径的要求。
`The custom engine version isn't valid. Names can include only lowercase letters (a-z), dashes (-), underscores (_), and periods (.). Specify a valid CEV, and try again.`	遵循所需的 CEV 命名约定。有关更多信息，请参阅 RDS Custom for SQL Server CEV 的要求。

消息	故障排除建议
The custom engine version isn't valid. Specify valid database engine version, and try again. Example: 15.00.4073.23-cev123.	提供了不支持的数据库引擎版本。使用支持的数据库引擎版本。
The expected architecture is (X) for image (AMI_ID), but architecture (Y) was found.	使用基于 x86_64 架构构建的 AMI。
The expected owner of image (AMI_ID) is customer account ID (ACCOUNT_ID), but owner (ACCOUNT_ID) was found.	从您拥有权限的 AMI 创建 EC2 实例。在 EC2 实例上运行 Sysprep 以创建和保存基本映像。
The expected platform is (X) for image (AMI_ID), but platform (Y) was found.	使用在 Windows 平台上构建的 AMI。
The expected root device type is (X) for image %s, but root device type (Y) was found.	使用 EBS 设备类型创建 AMI。

消息	故障排除建议
`The expected SQL Server edition is (X), but (Y) was found.`	选择支持的 AMI，该 AMI 具有 Windows Server 以及 SQL Server Enterprise、Standard 或 Web 版。从 EC2 Marketplace 选择具有以下使用情况操作代码之一的 AMI： • RunInstances:0102 - 含有 SQL Server Enterprise 的 Windows • RunInstances:0006 - 含有 SQL Server Standard 的 Windows • RunInstances:0202 - 含有 SQL Server Web 的 Windows
`The expected state is (X) for image (AMI_ID), but the following state was found: (Y).`	确保 AMI 的状态为 `AVAILABLE` 。
`The provided Windows OS name (X) isn't valid. Make sure the OS is one of the following: (Y).`	使用支持的 Windows 操作系统。
`Unable to find bootstrap log file in path.`	确认日志文件在 `C:\Program Files\Microsoft SQL Server\nnn\Setup Bootstrap\Log\Summary.txt` 中可用。
`RDS expected a Windows build version greater than or equal to (X), but found version (Y)..`	使用操作系统编译版本最低为 14393 的 AMI。

消息	故障排除建议
`RDS expected a Windows major version greater than or equal to (X), but found version (Y)..`	使用最低操作系统主版本为 10.0 或更高版本的 AMI。

修复 RDS Custom for SQL Server 中不支持的配置

由于责任共担模式，您应负责修复导致 RDS Custom for SQL Server 数据库实例进入 `unsupported-configuration` 状态的配置问题。如果问题与 Amazon 基础设施有关，则您可以使用控制台或 Amazon CLI 修复该问题。如果问题与操作系统或数据库配置有关，则您可以登录到主机进行修复。

> **Note**
>
> 本节介绍如何修复 RDS Custom for SQL Server 中不支持的配置。有关 RDS Custom for Oracle 的信息，请参阅修复 RDS Custom for Oracle 中不支持的配置。

在下表中，您可以找到支持外围所发送的通知和事件以及如何修复这些问题的说明。这些通知和支持外围可能会更改。有关支持外围的背景，请参阅RDS Custom 支持外围。有关事件描述，请参阅Amazon RDS 事件类别和事件消息。

事件代码	配置区域	RDS 事件消息	验证过程
`SP-S0000`	手动不支持的配置	RDS Custom 数据库实例状态设置为 [不受支持的配置]，原因是：X。	要解决此问题，请创建支持案例。

Amazon 资源（基础设施）

事件代码	配置区域	RDS 事件消息	验证过程
`SP-S1001`	EC2 实例状态	RDS Custom 数据库实例状态设置为 [不	要检查数据库实例的状态，请使用控制台或运行以下 Amazon CLI 命令：

事件代码	配置区域	RDS 事件消息	验证过程
		受支持的配置]，原因是：底层 EC2 实例 %s 已停止，但未停止 RDS 实例。您可以通过启动底层 EC2 实例并确保已附加二进制和数据卷来解决此问题。如果您打算停止 RDS 实例，请先确保底层 EC2 实例处于可用状态，然后使用 RDS 控制台或 CLI 停止 RDS 实例。	`aws rds describe-db-instances \ --db-instance-identifier db-instance-name |grep DBInstanceStatus`

`SP-S1002`	EC2 实例状态	RDS Custom 数据库实例状态设置为 [不受支持的配置]，原因是：RDS 数据库实例状态设置为 `STOPPED`，但底层 EC2 实例 %s 已启动。可以通过停止底层 EC2 实例来解决此问题。如果您打算启动 RDS 实例，请使用控制台或 CLI。	使用 Amazon CLI 命令检查数据库实例的状态：
			`aws rds describe-db-instances \ --db-instance-identifier db-instance-name | grep DBInstanceStatus`
			您还可以使用 EC2 控制台检查 EC2 实例的状态。
			要启动数据库实例，请使用控制台或运行以下 Amazon CLI 命令：
			`aws rds start-db-instance \ --db-instance-identifier db-instance-name`

事件代码	配置区域	RDS 事件消息	验证过程
SP-S1003	EC2 实例类	RDS Custom 数据库实例状态设置为 [不受支持的配置]，原因是：EC2 主机的预期和已配置数据库实例类别不匹配。可以通过将数据库实例类修改为其原始类的类型来解决此问题。	使用以下 CLI 命令可检查预期的数据库实例类： `aws rds describe-db-instances \ --db-instance-identifier db-instance-name | grep DBInstanceClass`
SP-S1004	无法访问 EBS 存储卷	RDS Custom 数据库实例状态设置为 [不受支持的配置]，原因是：与 EC2 实例关联的原始 EBS 存储卷 %s 目前无法访问。	
SP-S1005	EBS 存储卷已分离	RDS Custom 数据库实例状态设置为 [不受支持的配置]，原因是：未连接原始 EBS 存储卷“volume-id”。您可以通过附加与 EC2 实例关联的 EBS 卷来解决此问题。	重新附加 EBS 卷后，使用以下 CLI 命令检查 EBS 卷“volume-id”是否已正确附加到 RDS 实例： `aws ec2 describe-volumes \ --volume-ids volume-id | grep InstanceId`

事件代码	配置区域	RDS 事件消息	验证过程

SP-S1006	EBS 存储卷大小	RDS Custom 数据库实例状态设置为 [不受支持的配置]，原因是：EBS 存储卷“volume-id”的预期设置和已配置设置不匹配。卷大小已在 EC2 级别从其原始值 [%s] 手动更改。要解决此问题，请创建支持案例。	使用以下 CLI 命令比较 EBS 卷“volume-id”详细信息和 RDS 实例详细信息的卷大小：
			`aws rds describe-db-instances \ --db-instance-identifier db-instance-name | grep AllocatedStorage`
			使用以下 CLI 命令查看实际分配的卷大小：
			`aws ec2 describe-volumes \ --volume-ids |grep Size`

<table>
<tr><th>事件代码</th><th>配置区域</th><th>RDS 事件消息</th><th>验证过程</th></tr>
<tr><td>SP-S1007</td><td>EBS 存储卷配置</td><td>RDS Custom 数据库实例状态设置为 [不受支持的配置]，原因是：EBS 存储卷“volume-id”的预期设置和已配置设置不匹配。您可以通过在 EC2 级别将 EBS 存储卷配置 [IOPS、吞吐量、卷类型] 修改为其原始值 [IOPS：%s，吞吐量：%s，卷类型：%s] 来解决此问题。如果将来需要修改存储，请使用 RDS 控制台或 CLI。卷大小也已在 EC2 级别从其原始值 [%s] 手动更改。要解决此问题，请创建支持案例。</td><td>使用以下 CLI 命令比较 EBS 卷“volume-id”详细信息和 RDS 实例详细信息的卷类型。确保 EBS 级别的值与 RDS 级别的值相匹配：<pre>aws rds describe-db-instances \
    --db-instance-identifier db-instance-name |
grep StorageType</pre>要获取 RDS 级别的存储吞吐量的预期值，请执行以下操作：<pre>aws rds describe-db-instances \
    --db-instance-identifier db-instance-name |
grep StorageThroughput</pre>要获取 RDS 级别的卷 IOPS 的预期值，请执行以下操作：<pre>aws rds describe-db-instances \
    --db-instance-identifier db-instance-name |
grep Iops</pre>要获取 EC2 级别的当前存储类型，请执行以下操作：<pre>aws ec2 describe-volumes \
    --volume-ids |grep
 VolumeType</pre></td></tr>
</table>

事件代码	配置区域	RDS 事件消息	验证过程

			要获取 EC2 级别的存储吞吐量的当前值，请执行以下操作：
			`aws ec2 describe-volumes \ --volume-ids |grep Throughput`
			要获取 EC2 级别的卷 IOPS 的当前值，请执行以下操作：
			`aws ec2 describe-volumes \ --volume-ids |grep Iops`

事件代码	配置区域	RDS 事件消息	验证过程

SP-S1008	EBS 存储卷大小和配置	RDS Custom 数据库实例状态设置为 [不受支持的配置]，原因是：EBS 存储卷“volume-id”的预期设置和已配置设置不匹配。您可以通过在 EC2 级别将 EBS 存储卷配置 [IOPS、吞吐量、卷类型] 修改为其原始值 [IOPS：%s，吞吐量：%s，卷类型：%s] 来解决此问题。如果将来需要修改存储，请使用 RDS 控制台或 CLI。卷大小也已在 EC2 级别从其原始值 [%s] 手动更改。要解决此问题，请创建支持案例。	使用以下 CLI 命令比较 EBS 卷“volume-id”详细信息和 RDS 实例详细信息的卷类型。确保 EBS 级别的值与 RDS 级别的值相匹配：
			`aws rds describe-db-instances \`
			`    --db-instance-identifier db-instance-name | grep StorageType`
			要获取 RDS 级别的存储吞吐量的预期值，请执行以下操作：
			`aws rds describe-db-instances \`
			`    --db-instance-identifier db-instance-name | grep StorageThroughput`
			要获取 RDS 级别的卷 IOPS 的预期值，请执行以下操作：
			`aws rds describe-db-instances \`
			`    --db-instance-identifier db-instance-name | grep Iops`
			要获取 EC2 级别的当前存储类型，请执行以下操作：
			`aws ec2 describe-volumes \`
			`    --volume-ids |grep VolumeType`

事件代码	配置区域	RDS 事件消息	验证过程

			要获取 EC2 级别的存储吞吐量的当前值，请执行以下操作：
			`aws ec2 describe-volumes \ --volume-ids |grep Throughput`
			要获取 EC2 级别的卷 IOPS 的当前值，请执行以下操作：
			`aws ec2 describe-volumes \ --volume-ids |grep Iops`
			要获得预期的分配卷大小，请执行以下操作：
			`aws rds describe-db-instances \ --db-instance-identifier db-instance-name | grep AllocatedStorage`
			要获取实际的分配卷大小，请执行以下操作：
			`aws ec2 describe-volumes \ --volume-ids |grep Size`

事件代码	配置区域	RDS 事件消息	验证过程
SP-S1009	SQS 权限	RDS Custom 数据库实例状态设置为 [不受支持的配置]，原因是：IAM 实例配置文件缺少 Amazon Simple Queue Service（SQS）权限。您可以通过确保与主机关联的 IAM 配置文件具有以下权限来解决此问题：["SQS:SendMessage"、"SQS:ReceiveMessage"、"SQS:DeleteMessage"、"SQS:GetQueueUrl"]。	
SP-S1010	SQS VPC 端点	RDS Custom 数据库实例状态设置为 [不受支持的配置]，原因是：VPC 端点策略阻止了 Amazon Simple Queue Service（SQS）操作。您可以通过修改 VPC 端点策略以允许所需的 SQS 操作来解决此问题。	

事件代码	配置区域	RDS 事件消息	验证过程
SP-S1011	事件总线策略	RDS Custom 数据库实例状态设置为 [不受支持的配置]，原因是：事件总线 arn:aws:events:region-1:123456789012:event-bus/default 的基于资源的策略拒绝了 Amazon CloudWatch events:PutEvents 操作。要解决这个问题，请修改基于资源的策略，以支持对 EventBus %s 执行 events:PutEvents 操作。	
SP-S1012	CloudWatch VPC 权限	RDS Custom 数据库实例状态设置为 [不受支持的配置]，原因是：VPC 端点策略缺少访问 Amazon CloudWatch 事件的权限。要解决这个问题，请修改 VPC 端点策略，以便在 EventBus arn:aws:events:region-1:123456789012:event-bus/default 上支持 events:PutEvents。	

事件代码	配置区域	RDS 事件消息	验证过程
SP-S1013	服务控制策略	RDS Custom 数据库实例状态设置为 [不受支持的配置]，原因是：Amazon Organizations 中的服务控制策略缺少访问 Amazon CloudWatch 事件的权限。要解决这个问题，请修改服务控制策略，以便在 EventBus arn:aws:events:region-1:123456789012:event-bus/default 上支持 events:PutEvents。	
SP-S1014	IAM 实例配置文件	RDS Custom 数据库实例状态设置为 [不受支持的配置]，原因是：IAM 实例配置文件 %s 权限拒绝 Amazon CloudWatch 事件。要解决这个问题，请在与实例关联的 IAM 配置文件中，将 ["events:PutEvents"] 设置为“Allow”，然后在 EventBus arn:aws:events:region-1:123456789012:event-bus/default 上支持 events:PutEvents。	

事件代码	配置区域	RDS 事件消息	验证过程
SP-S1015	IAM 实例配置文件	RDS Custom 数据库实例状态设置为 [不受支持的配置]，原因是：IAM 实例配置文件 %s 缺少 Amazon CloudWatch 事件权限。要解决这个问题，请在与实例关联的 IAM 配置文件中包括 ["events:PutEvents"] 权限，然后在 EventBus arn:aws:events:region-1:123456789012:event-bus/default 上支持 events:PutEvents。	
SP-S1016	IAM 权限边界	RDS Custom 数据库实例状态设置为 [不受支持的配置]，原因是：IAM 实例配置文件 %s 具有拒绝 Amazon CloudWatch 事件的权限边界。要解决这个问题，请在 IAM 实例配置文件权限边界中，对于 EventBus arn:aws:events:region-1:123456789012:event-bus/default 将 ["events:PutEvents"] 设置为“Allow”。	

操作系统

<table>
<tr><th>事件代码</th><th>配置区域</th><th>RDS 事件消息</th><th>验证过程</th></tr>
<tr><td>SP-S2001</td><td>SQL 服务状态</td><td>RDS Custom 数据库实例状态设置为 [不受支持的配置]，原因是：SQL Server 服务未启动。您可以通过在主机上重启 SQL Server 服务来解决此问题。如果此数据库实例为多可用区数据库实例且重启失败，请停止并启动主机以发起失效转移。</td><td></td></tr>
<tr><td>SP-S2002</td><td>RDS Custom 代理状态</td><td>RDS Custom 数据库实例状态设置为 [不受支持的配置]，原因是：RDS Custom 代理服务未安装或无法启动。要解决此问题，可以查看 Windows 事件日志以确定服务无法启动的原因，然后采取适当的步骤来修复问题。如需其他帮助，请创建支持案例。</td><td>登录到主机并确保 RDS Custom 代理正在运行。

您可以使用以下命令来查看代理状态。

<pre><code>$name = "RDSCustomAgent"
$service = Get-Service $name
Write-Host $service.Status</code></pre>
如果状态不是 <code>Running</code>，则可使用以下命令启动该服务：

<pre><code>Start-Service $name</code></pre>
如果代理无法启动，请检查 Windows 事件以了解其无法启动的原因。代理需要 Windows 用户才能启动该服务。确保 Windows 用户存在并且具有运行该服务的权限。</td></tr>
</table>

事件代码	配置区域	RDS 事件消息	验证过程
SP-S2003	SSM 代理状态	RDS Custom 数据库实例状态设置为 [不受支持的配置]，原因是：无法访问 Amazon SSM 代理服务。可以通过使用 `Get-Service AmazonSSMAgent` PowerShell 命令检查服务状态或使用 `Start-Service AmazonSSMAgent` 启动服务来解决此问题。确保允许发往 ssm、ssmmessages 和 ec2messages 区域端点的 HTTPS（端口 443）出站流量。	有关更多信息，请参阅 SSM 代理故障排除。 要排查 SSM 端点故障，请参阅无法连接到 SSM 端点和使用 ssm-cli 排查托管式节点可用性问题。
SP-S2004	RDS Custom 代理登录	RDS Custom 数据库实例状态设置为 [不受支持的配置]，原因是：SQL 登录名 "`$HOSTNAME/RDSAgent`" 出现意外问题。要解决此问题，请创建支持案例。	

事件代码	配置区域	RDS 事件消息	验证过程
SP-S2005	时区	RDS Custom 数据库实例状态设置为 [不受支持的配置]，原因是：Amazon EC2 实例 [%s] 上的时区已更改。可以通过将时区修改回在实例创建期间指定的设置来解决此问题。如果您想创建具有特定时区的实例，请参阅 RDS Custom 文档。	运行 `Get-Timezone` PowerShell 命令以确认时区。 有关更多信息，请参阅 [RDS Custom for SQL Server 数据库实例的本地时区](#)。
SP-S2006	高可用性软件解决方案版本	RDS Custom 数据库实例状态设置为 [不受支持的配置]，原因是：当前实例的高可用性软件解决方案与预期版本不同。要解决此问题，请创建支持案例。	
SP-S2007	高可用性软件解决方案配置	RDS Custom 数据库实例状态设置为 [不受支持的配置]，原因是：实例 %s 上的高可用性软件解决方案的配置设置已修改为意外的值。要解决此问题，请重启 EC2 实例。当您重启 EC2 实例时，它会自动将设置更新为高可用性软件解决方案所需的配置。	

<table>
<tr><th>事件代码</th><th>配置区域</th><th>RDS 事件消息</th><th>验证过程</th></tr>
<tr><td>SP-S2008</td><td>SQL Server 服务</td><td>RDS Custom 数据库实例设置为 [不受支持的配置]：主机上不存在 SQLServer（MSSQLServer）服务。要解决此问题，请创建支持案例。</td><td>您可以使用以下命令来查看代理状态。<pre>$name = "MSSQLServer"
$service = Get-Service $name
Write-Host $service.Status</pre></td></tr>
<tr><td>SP-2009</td><td>SSL 证书</td><td>RDS Custom 数据库实例设置为 [不支持的配置]，原因是：非自签名 SSL 证书导致 RDS 中断。要解决此问题，请从受信任的根证书存储中移除非自签名证书。</td><td>运行以下 PowerShell 命令来查看非自签名证书。<pre>Get-ChildItem cert:\LocalMachine\root -Recurse | Where-Object {$_.Issuer -ne $_.Subject -and $_.Issuer -notlike "*RDSCustomAgentCA*"}</pre>有关更多信息，请参阅 HTTP Error 403.16 when you try to access a website that's hosted on IIS。</td></tr>
<tr><td>SP-2010</td><td>根卷存储状态</td><td>RDS Custom 数据库实例设置为 [不受支持的配置]，原因是：根卷存储已满。要解决此问题，请在根 EBS 卷“volume-id”中释放至少 500 MiB 的存储空间，或者在 EC2 实例“instance-id”上增加卷大小并调整 C 盘的大小。替换 EC2 实例时，根卷大小更改不会持续。</td><td>使用以下命令查看根（C:）卷上的可用存储空间。<pre>(Get-PSDrive -Name C).Free / 1MB</pre>有关修改 EBS 根卷的更多信息，请参阅 How</td></tr>
</table>

数据库

事件代码	配置区域	RDS 事件消息	验证过程
SP-S3001	SQL Server 共享内存协议	RDS Custom 数据库实例状态设置为 [不受支持的配置]，原因是：SQL Server 共享内存协议被禁用。可以通过在 SQL Server 配置管理器中启用共享内存协议来解决此问题。	可以通过检查 SQL Server 配置管理器 > SQL Server 网络配置 > MSSQLSERVER 的协议> 共享内存为“已启用”来验证这一点。启用协议后，重启 SQL Server 进程。
SP-S3002	服务主密钥	RDS Custom 数据库实例状态设置为 [不受支持的配置]，原因是：RDS Automation 无法将服务主密钥（SMK）的备份作为新 SMK 生成的一部分。要解决此问题，请创建支持案例。	
SP-S3003	服务主密钥	RDS Custom 数据库实例状态设置为 [不受支持的配置]，原因是：与服务主密钥（SMK）相关的元数据缺失或不完整。要解决此问题，请创建支持案例。	
SP-S3004	数据库引擎版本	RDS Custom 数据库实例状态设置为 [不受支持的配置]，原因是：预期的 SQL Server 版本和已安	运行以下查询以获取 SQL 版本： `select @@version`

<table>
<tr><th>事件代码</th><th>配置区域</th><th>RDS 事件消息</th><th>验证过程</th></tr>
<tr><td></td><td></td><td>装的 SQL Server 版本不匹配：不支持在 RDS Custom for SQL Server 上修改 SQL Server 版本。此外，不支持在 RDS Custom EC2 实例上手动更改 SQL Server 版本。要解决此问题，请创建支持案例。</td><td>运行以下 Amazon CLI 命令以获取 RDS SQL 引擎版本：
<pre>aws rds describe-db-instances \
--db-instance-identifier db-instance-name |grep EngineVersion
aws rds describe-db-instances \
--db-instance-identifier db-instance-name |grep Engine</pre>有关更多信息，请参阅修改 RDS Custom for SQL Server 数据库实例和升级数据库实例引擎版本。</td></tr>
<tr><td>SP-S3005</td><td>数据库引擎版本</td><td>RDS Custom 数据库实例状态设置为 [不受支持的配置]，原因是：当前 SQL Server 版本与预期的 SQL Server 版本 [%s] 不匹配：不支持在 RDS Custom for SQL Server 上修改 SQL Server 版本。要解决此问题，请创建支持案例。</td><td>运行以下查询可获取 SQL 版本：

Example
<pre>select @@version</pre>运行以下 Amazon CLI 命令以获取 RDS SQL 引擎版本：
<pre>aws rds describe-db-instances \
--db-instance-identifier db-instance-name |grep Engine</pre></td></tr>
</table>

事件代码	配置区域	RDS 事件消息	验证过程

SP-S3006	数据库引擎版本	RDS Custom 数据库实例状态设置为 [不受支持的配置]，原因是：当前 SQL Server 版本与预期的 SQL Server 版本 [%s] 不匹配：不能在 RDS Custom EC2 实例上手动更改 SQL Server 版本。要解决此问题，请创建支持案例。要在日后对 SQL Server 版本进行任何修改，可以从 Amazon RDS 控制台或通过 modify-db-instance CLI 命令修改实例。	运行以下查询以获取 SQL 版本：
			
			Example
			
			`select @@version`
			
			运行以下 Amazon CLI 命令以获取 RDS SQL 引擎版本：
			
			`aws rds describe-db-instances \ --db-instance-identifier db-instance-name |grep EngineVersion`
			
			有关更多信息，请参阅修改 RDS Custom for SQL Server 数据库实例和升级数据库实例引擎版本。

SP-S3007	数据库文件位置	RDS Custom 数据库实例状态设置为 [不受支持的配置]，原因是：数据库文件是在 D:\ 驱动器外部配置的。可以通过确保所有数据库文件（包括 ROW、LOG、FILESTREAM 等...）都存储在 D:\ 驱动器上来解决此问题。	运行以下查询以列出不在默认路径中的数据库文件的位置： `USE master; SELECT physical_name as files_not_in_default_path FROM sys.master_files WHERE SUBSTRING(physical_name,1,3)!='D:\';`

事件代码	配置区域	RDS 事件消息	验证过程
SP-S3008	已超出数据库计数限制	RDS Custom 数据库实例状态设置为 [不受支持的配置]，原因是：数据库实例上的数据库总数超过了 5000 的最大限制。要解决这个问题，请将数据库数量减少到支持的最大限制以下。	使用以下命令查看数据库总计数： `SELECT COUNT(name) as databaseCount FROM sys.databases WHERE name not in ('tempdb','master','model','msdb','DWDiagnostics','DWConfiguration','DWQueue');`

在 RDS Custom for SQL Server 中排查 **Storage-Full** 问题

RDS Custom 还会监控根（C:）卷。当根卷的可用磁盘空间小于 500 MiB 时，RDS Custom for SQL Server 数据库实例将移到 unsupported-configuration 状态。请参阅 修复 RDS Custom for SQL Server 中不支持的配置 中的 Event SP-S2010。

对 RDS Custom for SQL Server 中启用 TDE 的数据库的 PENDING_RECOVERY 状态进行故障排除

如果自动解密遇到问题，启用了透明数据加密（TDE）的 SQL Server 数据库可能会保持 PENDING_RECOVERY 状态。如果源数据库实例服务主密钥（SMK）备份文件（存储在您账户的 RDS Custom 托管式 S3 存储桶中）在还原完成之前已被删除，则通常会在数据库实例还原后发生这种情况。

要启用自动解密并使启用 TDE 的数据库联机，需要使用密码打开数据库主密钥（DMK），然后使用 SMK 对 DMK 进行加密。

使用以下 SQL Server 命令作为参考：

```
-- Identify PENDING_RECOVERY TDE databases
USE MASTER;
GO
SELECT name, is_encrypted, state_desc FROM sys.databases;
GO
```

```
-- Open DMK using password
OPEN MASTER KEY DECRYPTION BY PASSWORD = '<password>';
GO

-- Encrypt DMK using SMK
ALTER MASTER KEY ADD ENCRYPTION BY SERVICE MASTER KEY;
GO

-- Close SMK
CLOSE MASTER KEY;
GO

-- Bring the TDE databases online
ALTER DATABASE <database_name> SET ONLINE;
GO

-- Verify TDE databases are now in ONLINE state
SELECT name, is_encrypted, state_desc FROM sys.databases;
GO
```

Amazon RDS on Amazon Outposts

Amazon RDS on Amazon Outposts 将 RDS for SQL Server、RDS for MySQL 和 RDS for PostgreSQL 数据库扩展到 Amazon Outposts Rack Gen1 和 Gen2 环境中。Amazon Outposts 使用与公共 Amazon Web Services 区域中相同的硬件将 Amazon 服务、基础设施和操作模型引入本地。通过使用 RDS on Outposts，您可以在必须本地运行的业务应用程序附近预置托管数据库实例。有关 Amazon Outposts 的更多信息，请参阅 Amazon Outposts 文档和 Amazon Outposts 产品页。

与在Amazon Web Services 云中运行的 RDS 数据库实例一样，您可使用相同的Amazon Web Services Management Console、Amazon CLI 和 RDS API 来预置和管理本地部署 RDS on Outposts 数据库实例。RDS on Outposts 自动执行 Amazon S3 中的数据库预置、操作系统和数据库修补、备份和长期存档等任务。

RDS on Outposts 支持数据库实例的自动备份。要备份和还原数据库实例，需要 Outpost 和 Amazon Web Services 区域之间的网络连接。Outpost 中的所有数据库快照和事务日志都存储在您的 Amazon Web Services 区域中。从您的 Amazon 区域中，您可以将数据库快照中的数据库实例还原到其他 Outpost。有关更多信息，请参阅“备份简介”。

RDS on Outposts 支持数据库实例的自动维护和升级。有关更多信息，请参阅 维护数据库实例。

RDS on Outposts 使用 Amazon KMS key 密钥对数据库实例和数据库快照进行静态加密。有关静态加密的更多信息，请参阅 加密 Amazon RDS 资源。

默认情况下，Outposts 子网中的 EC2 实例可以使用 Amazon Route 53 DNS 服务将域名解析为 IP 地址。使用 Route 53，您的 DNS 解析时间可能会更长，这取决于 Outpost 和 Amazon Web Services 区域之间的路径延迟。在这种情况下，您可以使用在本地环境中以本地方式安装的 DNS 服务器。有关更多信息，请参阅 Amazon Outposts 用户指南中的 DNS。

与 Amazon Web Services 区域的网络连接不可用时，您的数据库实例将继续在本地运行。您可以将本地 DNS 服务器配置为辅助服务器，使用 DNS 名称解析继续访问数据库实例。但是，您不能创建新的数据库实例或修改现有的数据库实例。没有连接时，不会进行自动备份。如果某个数据库实例发生故障，则在恢复连接之前不会自动替换该数据库实例。我们建议尽快恢复网络连接。

主题

- 使用 Amazon RDS on Amazon Outposts 的多可用区部署
- 创建 Amazon RDS on Amazon Outposts 数据库实例
- 为 Amazon RDS on Amazon Outposts 创建只读副本
- 在 Amazon RDS on Amazon Outposts 上还原数据库实例的注意事项

Amazon RDS on Amazon Outposts 的先决条件

以下是使用 Amazon RDS on Amazon Outposts 的先决条件：

- 在您的本地数据中心安装 Amazon Outposts。有关更多信息，请参阅《Amazon Outposts 服务器安装指南》中的 Installing an Amazon Outposts server。
- 确保您至少有一个子网可用于 RDS on Outposts。您可以将同一子网用于其他工作负载。
- 确保 Outpost 和 Amazon 区域之间具有可靠的网络连接。

Amazon RDS on Amazon Outposts 对 Amazon RDS 功能的支持

下表介绍 Amazon Outposts 上 Amazon RDS 支持的 Amazon RDS 功能。

功能	支持	备注	更多信息

数据库实例配置	是	您只能为 RDS for SQL Server、RDS for MySQL 和 RDS for PostgreSQL 数据库引擎创建数据库实例。支持以下版本：
		• Microsoft SQL Server:
		  • 15.00.4043.16.v1 及更高的 2019 版本
		  • 14.00.3294.2.v1 及更高的 2017 版本
		  • 13.00.5820.21.v1 及更高的 2016 版本
		• MySQL 8.0 和 8.4 版本
		• 所有 PostgreSQL 16、15、14 和 13 版本，以及 PostgreSQL 版本 12.5 及更高的 12 版本
			创建 Amazon RDS on Amazon Outposts 数据库实例

使用 Microsoft SQL Server Management Studio 连接到 Microsoft SQL Server 数据库实例	是	某些 TLS 版本和加密密码可能不安全。要关闭它们，请按照 配置 SQL Server 安全协议和密码 中的说明进行操作。	连接到 Microsoft SQL Server 数据库实例
修改主用户密码	是	无	修改 Amazon RDS 数据库实例

功能	支持	备注	更多信息
重命名数据库实例	是	无	修改 Amazon RDS 数据库实例
重启数据库实例	是	无	重启中的数据库实例
停止数据库实例	是	无	暂时停止 Amazon RDS 数据库实例
启动数据库实例	是	无	启动之前停止的 Amazon RDS 数据库实例
多可用区部署	是	MySQL 和 PostgreSQL 数据库实例支持多可用区部署。 多可用区部署不支持直接 VPC 路由（DVR）。	创建 Amazon RDS on Amazon Outposts 数据库实例 配置和管理 Amazon RDS 的多可用区部署
数据库参数组	是	无	Amazon RDS 的参数组
只读副本	是	MySQL 和 PostgreSQL 数据库实例支持只读副本。 只读副本不支持直接 VPC 路由（DVR）。	为 Amazon RDS on Amazon Outposts 创建只读副本
静态加密	是	RDS on Outposts 不支持未加密的数据库实例。	加密 Amazon RDS 资源
Amazon Identity and Access Management (IAM) 数据库身份验证	否	无	适用于 MariaDB、MySQL 和 PostgreSQL 的 IAM 数据库身份验证

功能	支持	备注	更多信息
将 IAM 角色与数据库实例关联	否	无	add-role-to-db-instance Amazon CLI 命令 AddRoleToDBInstance RDS API 操作
Kerberos 身份验证	否	无	Kerberos 身份验证
为 Amazon RDS 资源添加标签	是	无	为 Amazon RDS 资源添加标签
选项组	是	无	使用选项组
修改维护时段	是	无	维护数据库实例
自动次要版本升级	是	无	自动升级次要引擎版本
修改备份时段	是	无	备份简介 修改 Amazon RDS 数据库实例
更改数据库实例类	是	无	修改 Amazon RDS 数据库实例
更改分配的存储空间	是	无	修改 Amazon RDS 数据库实例
Storage autoscaling (存储自动扩展)	是	无	使用 Amazon RDS 存储自动扩展功能自动管理容量

功能	支持	备注	更多信息
手动和自动数据库实例快照	是	您可以将自动备份和手动快照存储在您的 Amazon Web Services 区域 中。也可以将它们本地存储在 Outpost 上。 MySQL 和 PostgreSQL 数据库实例支持本地备份。 要在 Outposts 站上存储备份，请确保配置了 Amazon S3 on Outposts。 多可用区实例部署不支持本地备份。	创建 Amazon RDS on Amazon Outposts 数据库实例 Amazon S3 on Outposts 为 Amazon RDS 的单可用区数据库实例创建数据库快照
从数据库快照还原	是	您可以将已恢复的数据库实例的自动备份和手动快照存储在父级 Amazon Web Services 区域中或您本地的 Outpost 中。	在 Amazon RDS on Amazon Outposts 上还原数据库实例的注意事项 还原到数据库实例
从 Amazon S3 还原数据库实例	否	无	将备份还原到 Amazon RDS for MySQL 数据库实例
将快照数据导出到 Amazon S3	否	无	将 Amazon RDS 的数据库快照数据导出到 Amazon S3
时间点故障恢复	是	您可以将已恢复的数据库实例的自动备份和手动快照存储在父级 Amazon Web Services 区域中或您本地的 Outpost 中，但有一个例外情况。	在 Amazon RDS on Amazon Outposts 上还原数据库实例的注意事项 将 Amazon RDS 的数据库实例还原到指定时间

功能	支持	备注	更多信息
增强监控	否	无	使用增强监控来监控操作系统指标
Amazon CloudWatch 监控	是	您可以查看可用于 Amazon Web Services 区域中的数据库的同一组指标。	使用 Amazon CloudWatch 监控 Amazon RDS 指标
将数据库引擎日志发布到 CloudWatch Logs	是	无	将数据库日志发布到 Amazon CloudWatch Logs
事件通知	是	无	使用 Amazon RDS 事件通知
Amazon RDS 性能详情	否	无	在 Amazon RDS 上使用性能详情监控数据库负载
查看或下载数据库日志	否	RDS on Outposts 不支持使用控制台查看数据库日志或使用 Amazon CLI 或 RDS API 描述数据库日志。 RDS on Outposts 不支持使用控制台下载数据库日志或使用 Amazon CLI 或 RDS API 下载数据库日志。	监控 Amazon RDS 日志文件
Amazon RDS 代理	否	无	的 Amazon RDS 代理
Amazon RDS for MySQL 存储过程	是	无	RDS for MySQL 存储过程参考
使用 RDS for MySQL 的外部数据库进行复制	否	无	配置与外部源实例之间的二进制日志文件位置复制

功能	支持	备注	更多信息
Amazon RDS for Microsoft SQL Server 本机备份和还原	是	无	使用本机备份和还原导入和导出 SQL Server 数据库

Amazon RDS on Amazon Outposts 支持的数据库实例类

Amazon RDS on Amazon Outposts 支持以下数据库实例类：

- 通用数据库实例类
  - db.m7i.48xlarge
  - db.m7i.24xlarge
  - db.m7i.16xlarge
  - db.m7i.12xlarge
  - db.m7i.8xlarge
  - db.m7i.4xlarge
  - db.m7i.2xlarge
  - db.m7i.xlarge
  - db.m7i.large
  - db.m5.24xlarge
  - db.m5.16xlarge
  - db.m5.12xlarge
  - db.m5.8xlarge
  - db.m5.4xlarge
  - db.m5.2xlarge
  - db.m5.xlarge
  - db.m5.large
- 内存优化型数据库实例类
  - db.r7i.48xlarge

  - db.r7i.24xlarge

- db.r7i.16xlarge
- db.r7i.12xlarge
- db.r7i.8xlarge
- db.r7i.4xlarge
- db.r7i.2xlarge
- db.r7i.xlarge
- db.r7i.large
- db.r5.24xlarge
- db.r5.16xlarge
- db.r5.12xlarge
- db.r5.8xlarge
- db.r5.4xlarge
- db.r5.2xlarge
- db.r5.xlarge
- db.r5.large

根据您配置 Outpost 的方式，可能并非所有这些类都可用。例如，如果您尚未为 Outpost 购买 db.r5 类，则不能将它们与 RDS on Outposts 一起使用。

RDS on Outposts 数据库实例仅支持通用 SSD 存储。有关数据库实例类的更多信息，请参阅数据库实例类。

Amazon RDS 负责管理数据库实例的维护和恢复，并需要 Outpost 上的活动容量来执行此操作。我们建议您为生产环境中的每个数据库实例类配置 N+1 个 EC2 实例。RDS on Outposts 可以将这些 EC2 实例的额外容量用于维护和修复操作。例如，如果您的生产环境有 3 个 db.m5.large 和 5 个 db.r5.xlarge 数据库实例类，那么我们建议它们至少配置 4 个 m5.large EC2 实例和 6 个 r5.xlarge EC2 实例。有关更多信息，请参阅 Amazon Outposts 用户指南中的 Amazon Outposts 中的弹性。

客户拥有的适用于 Amazon RDS on Amazon Outposts 的 IP 地址

Amazon RDS on Amazon Outposts 会使用您提供的有关本地部署网络的信息创建一个地址池。该池称为客户拥有的 IP 地址池（CoIP 池）。客户拥有的 IP 地址 (COIP) 通过您的本地网络提供连接到 Outpost 子网中资源的本地或外部连接。有关 CoIP 的更多信息，请参阅 Amazon Outposts 用户指南中的客户拥有的 IP 地址。

每个 RDS on Outposts 数据库实例都有一个用于其 Virtual Private Cloud (VPC) 内流量的私有 IP 地址。此私有 IP 地址不可公开访问。您可以使用公有选项设置数据库实例除了私有 IP 地址之外是否还具有公有 IP 地址。使用公有 IP 地址进行连接是通过互联网路由它们，在某些情况下可能会导致较高的延迟。

RDS on Outposts 支持通过子网将 CoIP 用于数据库实例，而不是使用这些私有和公有 IP 地址。在将 CoIP 用于 RDS on Outposts 数据库实例时，您将使用数据库实例端点连接到数据库实例。然后，RDS on Outposts 可以自动将 CoIP 用于来自 VPC 内部和外部的所有连接。

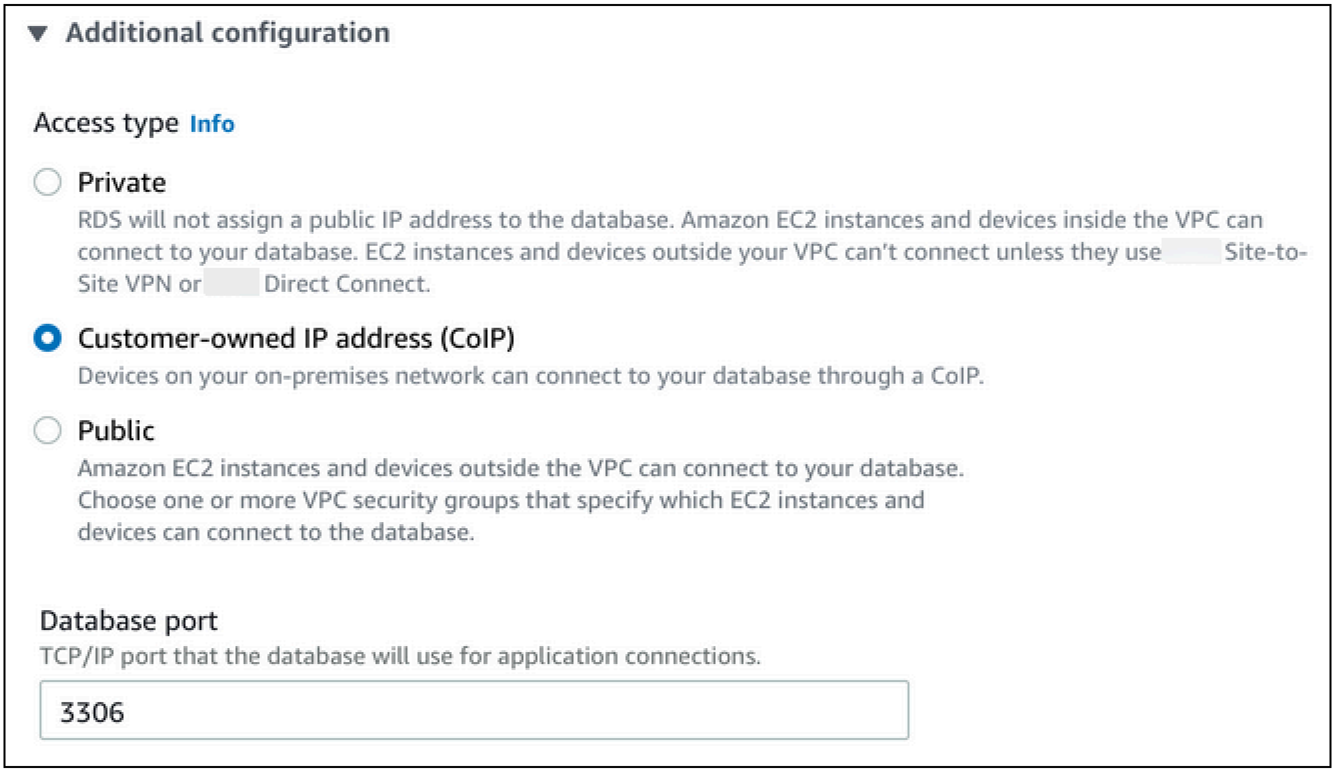
COIP 可以为 RDS on Outposts 数据库实例带来以下益处：

- 降低连接延迟
- 增强安全性

使用 CoIP

您可以使用Amazon Web Services Management Console、Amazon CLI 或 RDS API 为 RDS on Outposts 数据库实例启用或禁用 CoIP：

- 通过Amazon Web Services Management Console启用时，请选择 Access type（访问类型）中的 Customer-owned IP address (CoIP)（客户拥有的 IP 地址 (CoIP)）来使用 CoIP。选择其他设置之一可将其禁用。



- 通过 Amazon CLI 来启用/禁用时，请使用 `--enable-customer-owned-ip | --no-enable-customer-owned-ip` 选项。
- 通过 RDS API 来启用/禁用时，请使用 `EnableCustomerOwnedIp` 参数。

您可以在执行以下任一操作时启用或禁用 CoIP：

- 创建数据库实例

  有关更多信息，请参阅“创建 Amazon RDS on Amazon Outposts 数据库实例”。
- 修改数据库实例

  有关更多信息，请参阅 修改 Amazon RDS 数据库实例。
- 创建只读副本

  有关更多信息，请参阅 为 Amazon RDS on Amazon Outposts 创建只读副本。
- 从数据库快照还原数据库实例

  有关更多信息，请参阅“还原到数据库实例”。
- 将数据库实例还原到指定时间

有关更多信息，请参阅 将 Amazon RDS 的数据库实例还原到指定时间。

> Note
>
> 在某些情况下，您可以为数据库实例启用 CoIP，但 Amazon RDS 无法为数据库实例分配 CoIP。在此类情况下，数据库实例状态更改为 incompatible-network（不兼容网络）。有关数据库实例状态的更多信息，请参阅 查看Amazon RDS 数据库实例状态。

限制

以下限制适用于 RDS on Outposts 数据库实例的 CoIP 支持：

- 将 CoIP 用于数据库实例时，请确保对数据库实例禁用了公开可访问性。
- 确保您的 VPC 安全组的入站规则包含 CoIP 地址范围（CIDR 块）。有关设置安全组的更多信息，请参阅通过创建安全组提供对 VPC 中的数据库实例的访问。
- 您无法将 CoIP 池中的 CoIP 分配给数据库实例。在将 CoIP 用于数据库实例时，Amazon RDS 会自动将 CoIP 池中的 CoIP 分配给数据库实例。
- 您必须使用拥有 Outpost 资源的 Amazon Web Services 账户（拥有者），或与同一组织中的其他 Amazon Web Services 账户（使用者）共享以下资源：
  - Outpost
  - 数据库实例 VPC 的本地网关 (LGW) 路由表
  - LGW 路由表的 CoIP 池

  有关更多信息，请参阅 Amazon Outposts 用户指南中的使用共享的 Amazon Outposts 资源。

使用 Amazon RDS on Amazon Outposts 的多可用区部署

对于多可用区部署，Amazon RDS 会在一个 Amazon Outpost 上创建主数据库实例。RDS 会将数据同步复制到另一个 Outpost 上的备用数据库实例。

Amazon Outposts 上多可用区部署的操作原理与 Amazon Web Services 区域中的多可用区部署相似，但有以下区别：

- 它们需要两个或更多 Outposts 之间的本地连接。
- 它们需要客户拥有的 IP (CoIP) 池。有关更多信息，请参阅 客户拥有的适用于 Amazon RDS on Amazon Outposts 的 IP 地址。
- 复制在您的本地网络上运行。

Amazon Outposts 上的多可用区适用于 RDS on Outposts 上所有受支持的 MySQL 和 PostgreSQL 版本。多可用区部署不支持本地备份。有关更多信息，请参阅 创建 Amazon RDS on Amazon Outposts 数据库实例。

使用责任共担模式

虽然 Amazon 通过商业上合理的努力来提供配置为高可用性的数据库实例，但可用性使用责任共担模式。RDS on Outposts 的故障切换和修复数据库实例的能力要求您的每个 Outposts 都连接到其 Amazon Web Services 区域。

RDS on Outposts 还要求托管主数据库实例的 Outpost 和托管备用数据库实例的 Outpost 之间具有连接性，以进行同步复制。对此连接的任何影响都可能会阻止 RDS on Outposts 执行故障转移。

您可能会看到，由于执行同步数据复制，标准数据库实例部署的延迟有所增加。托管主数据库实例的 Outpost 和托管备用数据库实例的 Outpost 之间连接的带宽和延迟直接影响部署的延迟。有关更多信息，请参阅 先决条件。

提高可用性

我们建议通过执行以下操作来提高可用性：

- 为任务关键型应用程序分配足够的额外容量，以便在出现潜在主机问题时进行恢复和故障转移。这适用于数据库子网组中包含子网的所有 Outposts。有关更多信息，请参阅 Amazon Outposts 中的弹性。
- 为您的 Outposts 提供冗余网络连接。

- 使用两个以上的 Outposts。拥有两个以上的 Outpost 时，Amazon RDS 可恢复数据库实例。如果当前 Outpost 出现故障，RDS 可以通过将数据库实例移到另一个 Outpost 来执行此恢复。
- 为您的 Outpost 提供双电源和冗余网络连接。

我们建议您的本地网络满足以下要求：

- 托管主数据库实例的 Outpost 和托管备用数据库实例的 Outpost 之间的往返时间 (RTT) 延迟直接影响写入延迟。将 Amazon Outposts 之间的 RTT 延迟保持在较低的一位数毫秒内。我们建议不超过 5 毫秒，但您的要求可能会有所不同。

  您可以在 `WriteLatency` 的 Amazon CloudWatch 指标中找到对网络延迟的净影响。有关更多信息，请参阅 [Amazon RDS 的 Amazon CloudWatch 指标](#)。
- Outposts 之间连接的可用性会影响数据库实例的整体可用性。Outposts 之间有冗余的网络连接。

先决条件

RDS on Outposts 上的多可用区部署具有以下先决条件：

- 至少有两个 Outposts，通过本地连接进行连接并连接到 Amazon Web Services 区域中的不同可用区。
- 确保数据库子网组包含以下内容：
  - 在给定的 Amazon Web Services 区域中，在至少两个可用区中至少有两个子网。
  - 子网只在 Outposts 中。
  - 在同一 Virtual Private Cloud (VPC) 中，在至少两个 Outposts 中有至少两个子网。
- 将数据库实例的 VPC 与所有本地网关路由表相关联。必须进行此关联，因为复制使用 Outposts 的本地网关在本地网络上运行。

  例如，假设您的 VPC 在 Outpost A 中包含子网 A，在 Outpost B 中包含子网 B。Outpost A 使用 LocalGateway-A (LGW-A)，Ootpost B 使用 LocalGateway-B (LGW-B)。LGW-A 有 RouteTable-A，LGW-B 有 RouteTable-B。您希望同时将 RouteTable-A 和 RouteTable-B 用于复制流量。为此，请将您的 VPC 同时与 RouteTable-A 和 RouteTable-B 相关联。

  有关如何创建关联的更多信息，请参阅 Amazon EC2 [create-local-gateway-route-table-vpc-association](#) Amazon CLI 命令。
- 确保您的 Outposts 使用客户拥有的 IP (CoIP) 路由。每个路由表还必须至少有一个地址池。Amazon RDS 会为主数据库实例和备用数据库实例分别分配一个额外的 IP 地址以进行数据同步。

- 确保拥有 RDS 数据库实例的 Amazon Web Services 账户 拥有本地网关路由表和 CoIP 池。或者确保它是 Resource Access Manager 共享的一部分，可访问本地网关路由表和 CoIP 池。
- 确保可以将 CoIP 池中的 IP 地址从一个 Outpost 本地网关路由到其他网关。
- 确保 VPC 的 CIDR 块（例如 10.0.0.0/4）和您的 CoIP 池 CIDR 块不包含来自 E 类 (240.0.0.0/4) 的 IP 地址。RDS 在内部使用这些 IP 地址。
- 确保正确设置出站和相关的入站流量。

  RDS on Outposts 会在主数据库实例和备用数据库实例之间建立虚拟专用网络 (VPN) 连接。要使此连接正常工作，您的本地网络必须允许 Internet 安全关联和密钥管理协议 (ISAKMP) 的出站和相关入站流量。它使用用户数据报协议 (UDP) 端口 500 和使用 UDP 端口 4500 的 IP 安全 (IPsec) 网络地址转换遍历 (NAT-T) 来完成此操作。

有关 CoIP 的更多信息，请参阅本指南中的 客户拥有的适用于 Amazon RDS on Amazon Outposts 的 IP 地址，以及《Amazon Outposts 用户指南》中的客户拥有的 IP 地址。

使用 Amazon EC2 权限 API 操作

无论您是否将 CoIP 用于 Amazon Outposts 上的数据库实例，RDS 都需要访问您的 CoIP 池资源。RDS 可以代表您为 CoIP 调用以下 EC2 权限 API 操作来进行多可用区部署：

- `CreateCoipPoolPermission` – 当您在 RDS on Outposts 上创建多可用区数据库实例时
- `DeleteCoipPoolPermission` – 当您从 RDS on Outposts 上删除多可用区数据库实例时

这些 API 操作向内部 RDS 账户授予或从中删除从权限指定的 CoIP 池分配弹性 IP 地址的权限。您可以使用 `DescribeCoipPoolUsage` API 操作查看这些 IP 地址。有关 CoIP 的更多信息，请参阅客户拥有的适用于 Amazon RDS on Amazon Outposts 的 IP 地址和《Amazon Outposts 用户指南》中的客户拥有的 IP 地址。

RDS 还可以代表您为本地网关路由表调用以下 EC2 权限 API 操作来进行多可用区部署：

- `CreateLocalGatewayRouteTablePermission` – 当您在 RDS on Outposts 上创建多可用区数据库实例时
- `DeleteLocalGatewayRouteTablePermission` – 当您从 RDS on Outposts 上删除多可用区数据库实例时

这些 API 操作授予或删除内部 RDS 账户将内部 RDS VPC 与本地网关路由表关联的权限。您可以使用 `DescribeLocalGatewayRouteTableVpcAssociations` API 操作查看这些路由表–VPC 关联。

创建 Amazon RDS on Amazon Outposts 数据库实例

创建 Amazon RDS on Amazon Outposts 数据库实例类似于在Amazon云中创建 Amazon RDS 数据库实例。但是，请确保指定与 Outpost 关联的数据库子网组。

基于 Amazon VPC 服务的 Virtual Private Cloud (VPC) 可以跨越一个 Amazon Web Services 区域中的所有可用区。您可以通过添加 Outpost 子网将 Amazon Web Services 区域中的任何 VPC 扩展到您的 Outpost。要将 Outpost 子网添加到 VPC，请在创建子网时指定 Outpost 的 Amazon Resource Name (ARN)。

创建 Amazon RDS on Outposts 数据库实例之前，您可以创建一个数据库子网组，该组包括一个与您的 Outpost 相关联的子网。在创建 Outposts 上的 RDS 数据库实例时，请指定此数据库子网组。您也可以在创建数据库实例时选择创建新的数据库子网组。

有关配置 Amazon Outposts 的信息，请参阅 [Amazon Outposts 用户指南]。

控制台

创建数据库子网组

创建包括一个与您的 Outpost 相关联的子网的数据库子网组。

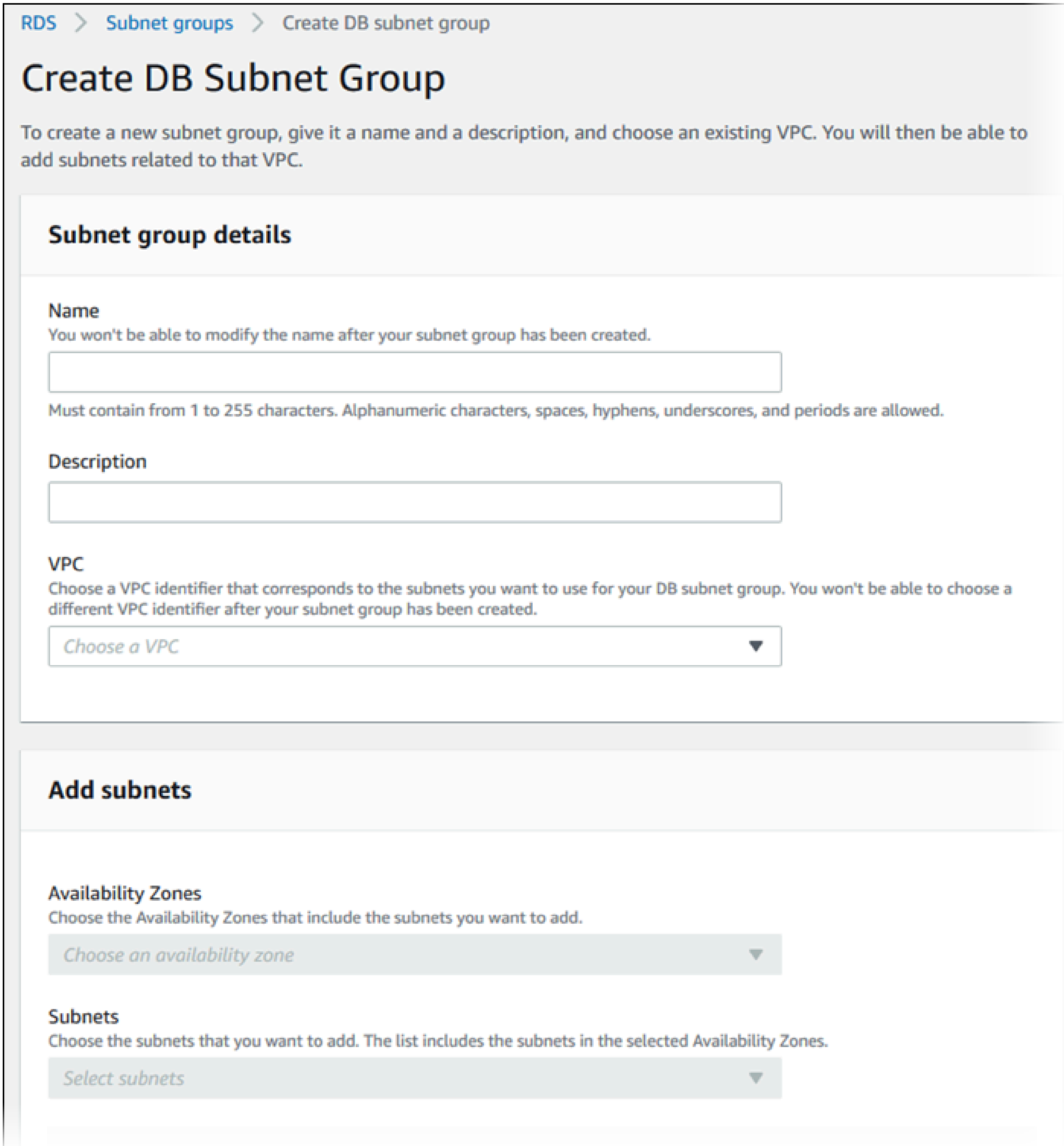
您也可以在创建数据库实例时选择为 Outpost 创建新的数据库子网组。如果您想要这样做，则跳过此过程。

> **Note**
>
> 要为Amazon Web Services 云创建数据库子网组，请至少指定两个子网。

要为 Outpost 创建数据库子网组

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在 Amazon RDS 控制台的右上角，选择要在其中创建数据库子网组的 Amazon Web Services 区域。
3. 选择子网组，然后选择创建数据库子网组。

   将显示创建数据库子网组页面。



4. 对于 Name（名称），选择数据库子网组的名称。
5. 对于 Description（描述），选择数据库子网组的描述。
6. 对于 VPC，选择您为其创建数据库子网组的 VPC。
7. 对于可用区，为您的 Outpost 选择可用区。

8. 对于 Subnets (子网)，选择 RDS on Outposts 使用的子网。
9. 选择 Create (创建) 以创建数据库子网组。

创建 RDS on Outposts 数据库实例

创建数据库实例，并为您的数据库实例选择 Outpost。

使用控制台创建 Outposts 上的 RDS 数据库实例

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在 Amazon RDS 控制台的右上角，选择要在其上创建数据库实例的 Outpost 所连接的 Amazon Web Services 区域。
3. 在导航窗格中，选择 Databases（数据库）。
4. 选择创建数据库。

   Amazon Web Services Management Console 将检测您已配置的可用 Outpost，并在数据库位置部分中显示本地选项。

   > **Note**
   >
   > 如果您尚未配置任何 Outpost，则数据库位置部分不会显示，或 Outposts 上的 RDS 选项在选择本地创建方法部分中不可用。

5. 对于 Database location（数据库位置），选择 On-premises（本地）。
6. 对于 On-premises creation method（本地创建方法），选择 RDS on Outposts。
7. 为 Outposts Connectivity（Outposts 连接性）指定您的设置。这些设置适用于使用具有数据库实例的数据库子网组的 VPC 的 Outpost。您的 VPC 必须基于 Amazon VPC 服务。

   a. 对于 Virtual Private Cloud (VPC)，选择包含您的数据库实例的数据库子网组的 VPC。
   b. 对于 VPC security group（VPC 安全组），为您的数据库实例选择 Amazon VPC 安全组。
   c. 对于 DB subnet group（数据库子网组），选择数据库实例的数据库子网组。

      例如，如果您执行了 创建数据库子网组 中的过程，可以选择与 Outpost 关联的现有数据库子网组。

      您也可以为 Outpost 创建新的数据库子网组。

8. 对于多可用区部署，选择 Create a standby instance (recommended for production usage)（创建备用实例（推荐用于生产用途）），以在另一个 Outpost 中创建备用数据库实例。

    > **Note**
    >
    > 此选项不可用于 Microsoft SQL Server。
    >
    > 如果您选择创建多可用区部署，将无法在 Outpost 上存储备份。

9. 在 Backup（备份）下，执行以下操作：

    a. 对于 Backup target（备份目标），请选择以下选项之一：

        - 在父级 Amazon Web Services 区域中存储自动备份和手动快照的 Amazon Web Services 云。
        - 创建本地备份的 Outposts（本地）。

        > **Note**
        >
        > 要在 Outpost 上存储备份，您的 Outpost 必须具有 Amazon S3 功能。有关更多信息，请参阅 Amazon S3 on Outposts。
        >
        > 多可用区部署或只读副本不支持本地备份。

    b. 选择 Enable automated backups（启用自动备份）创建数据库实例的时间点快照。

        如果您启用自动备份，可以选择 Backup retention period（备份保留期）和 Backup window（备份时段）的值，或保留原定设置值。

10. 根据需要，指定其他数据库实例设置。

    有关创建数据库实例时每个设置的信息，请参阅数据库实例的设置。

11. 选择创建数据库。

    随后会显示 Databases（数据库）页面。广告条告诉您正在创建数据库实例，并显示 View credential details（查看凭证详细信息）按钮。

查看数据库实例详细信息

创建数据库实例后，您可以查看凭证以及它的其他详细信息。

查看数据库实例详细信息

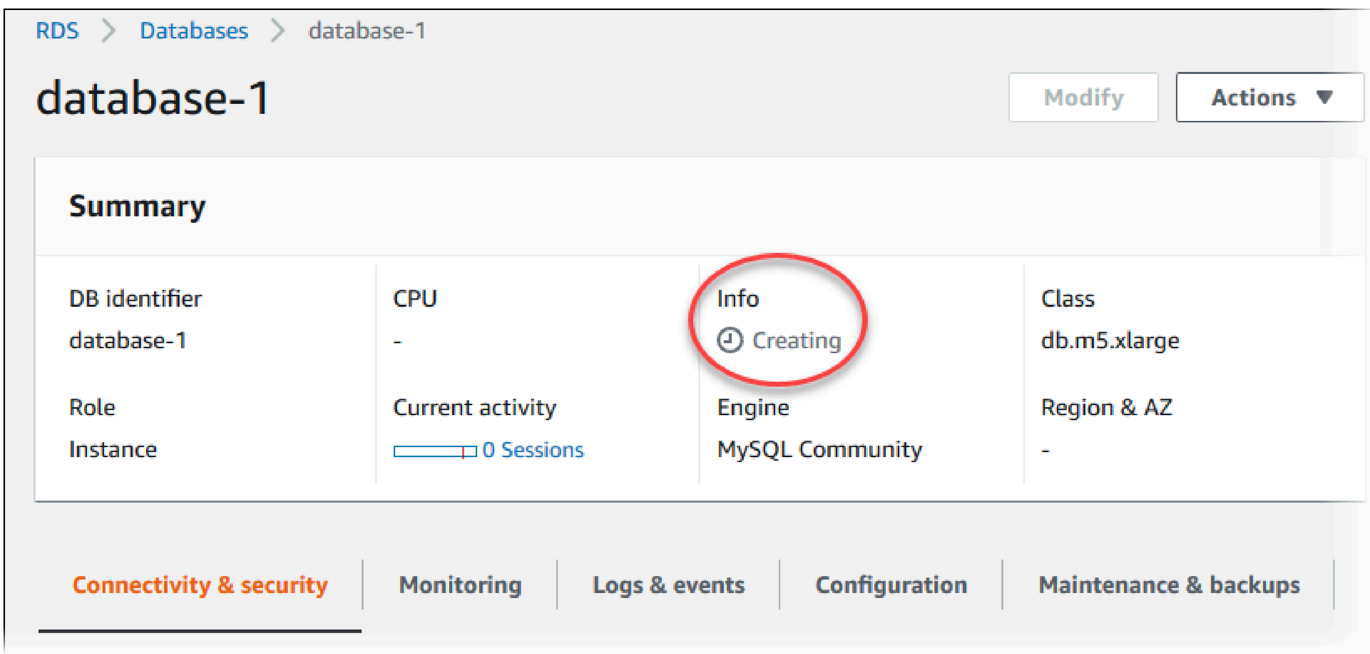
1. 要查看数据库实例的主用户名和密码，请选择 Databases（数据库）页面上的 View credential details（查看凭证详细信息）。

   您可以使用这些凭证以主用户身份连接到数据库实例。

   > **Important**
   >
   > 您无法再次查看主用户密码。如果您不记录它，您可能需要更改它。要在数据库实例可用后更改主用户密码，请修改数据库实例。有关修改数据库实例的更多信息，请参阅 修改 Amazon RDS 数据库实例。

2. 在 Databases（数据库）页面上选择新数据库实例的名称。

   在 RDS 控制台上，将显示新数据库实例的详细信息。数据库实例具有 Creating (正在创建) 状态，直到该数据库实例完成创建并可供使用。当状态变为 Available (可用) 时，您可以连接到该数据库实例。根据所分配的数据库实例类和存储的不同，新数据库实例可能需要数分钟时间才能变得可用。



数据库实例可用后，可以像在 Amazon Web Services 云 中管理 RDS 数据库实例一样进行管理。

Amazon CLI

在 Outpost 中使用 Amazon CLI 创建新的数据库实例之前，请先创建数据库子网组以供 RDS on Outposts 使用。

要为 Outpost 创建数据库子网组

- 使用 create-db-subnet-group 命令。对于 `--subnet-ids`，请在 Outpost 中指定 RDS on Outposts 使用的子网组。

  对于 Linux、macOS 或 Unix：

  ```
  aws rds create-db-subnet-group \
      --db-subnet-group-name myoutpostdbsubnetgr \
      --db-subnet-group-description "DB subnet group for RDS on Outposts" \
      --subnet-ids subnet-abc123
  ```

  对于 Windows：

  ```
  aws rds create-db-subnet-group ^
      --db-subnet-group-name myoutpostdbsubnetgr ^
      --db-subnet-group-description "DB subnet group for RDS on Outposts" ^
      --subnet-ids subnet-abc123
  ```

使用 Amazon CLI 创建 RDS on Outposts 数据库实例

- 使用 create-db-instance 命令。指定 Outpost 的可用区、与 Outpost 关联的 Amazon VPC 安全组以及您为 Outpost 创建的数据库子网组。您可以包括以下选项：
  - `--db-instance-identifier`
  - `--db-instance-class`
  - `--engine` – 数据库引擎。使用下列值之一：
    - MySQL – 指定 `mysql`。
    - PostgreSQL – 指定 `postgres`。
    - Microsoft SQL Server – 指定 `sqlserver-ee`、`sqlserver-se` 或 `sqlserver-web`。
  - `--availability-zone`
  - `--vpc-security-group-ids`

- `--db-subnet-group-name`
- `--allocated-storage`
- `--max-allocated-storage`
- `--master-username`
- `--master-user-password`
- `--multi-az | --no-multi-az` –（可选）是否在不同的可用区中创建备用数据库实例。默认为 `--no-multi-az`。

  `--multi-az` 选项不可用于 SQL Server。
- `--backup-retention-period`
- `--backup-target` –（可选）存储自动备份和手动快照的位置。使用下列值之一：
  - `outposts` – 将它们在本地存储在 Outpost 上。
  - `region` – 将它们存储在父级 Amazon Web Services 区域中。这是默认值。

  如果使用 `--multi-az` 选项，无法将 `outposts` 用于 `--backup-target`。此外，如果您将 `outposts` 用于 `--backup-target`，则数据库实例不能有只读副本。
- `--storage-encrypted`
- `--kms-key-id`

Example

以下示例使用 Outpost 中存储的备份创建名为 `myoutpostdbinstance` 的 MySQL 数据库实例。

对于 Linux、macOS 或 Unix：

```
aws rds create-db-instance \
    --db-instance-identifier myoutpostdbinstance \
    --engine-version 8.0.17 \
    --db-instance-class db.m5.large \
    --engine mysql \
    --availability-zone us-east-1d \
    --vpc-security-group-ids outpost-sg \
    --db-subnet-group-name myoutpostdbsubnetgr \
    --allocated-storage 100 \
    --max-allocated-storage 1000 \
    --master-username masterawsuser \
    --manage-master-user-password \
```

```
    --backup-retention-period 3 \
    --backup-target outposts \
    --storage-encrypted \
    --kms-key-id mykey
```

对于 Windows：

```
aws rds create-db-instance ^
    --db-instance-identifier myoutpostdbinstance ^
    --engine-version 8.0.17 ^
    --db-instance-class db.m5.large ^
    --engine mysql ^
    --availability-zone us-east-1d ^
    --vpc-security-group-ids outpost-sg ^
    --db-subnet-group-name myoutpostdbsubnetgr ^
    --allocated-storage 100 ^
    --max-allocated-storage 1000 ^
    --master-username masterawsuser ^
    --manage-master-user-password ^
    --backup-retention-period 3 ^
    --backup-target outposts ^
    --storage-encrypted ^
    --kms-key-id mykey
```

有关创建数据库实例时每个设置的信息，请参阅数据库实例的设置。

RDS API

要使用 RDS API 在 Outpost 中创建新的数据库实例，请首先通过调用 CreateDBSubnetGroup 操作来创建 RDS on Outposts 使用的数据库子网组。对于 `SubnetIds`，请在 Outpost 中指定 RDS on Outposts 使用的子网组。

接下来，使用以下参数调用 CreateDBInstance 操作。指定 Outpost 的可用区、与 Outpost 关联的 Amazon VPC 安全组以及您为 Outpost 创建的数据库子网组。

- `AllocatedStorage`
- `AvailabilityZone`
- `BackupRetentionPeriod`
- `BackupTarget`

如果您正在创建多可用区数据库实例部署，则无法将 outposts 用于 BackupTarget。此外，如果您将 outposts 用于 BackupTarget，则数据库实例不能有只读副本。

- DBInstanceClass
- DBInstanceIdentifier
- VpcSecurityGroupIds
- DBSubnetGroupName
- Engine
- EngineVersion
- MasterUsername
- MasterUserPassword
- MaxAllocatedStorage（可选）
- MultiAZ（可选）
- StorageEncrypted
- KmsKeyID

有关创建数据库实例时每个设置的信息，请参阅数据库实例的设置。

为 Amazon RDS on Amazon Outposts 创建只读副本

Amazon RDS on Amazon Outposts 使用 MySQL 和 PostgreSQL 数据库引擎的内置复制功能，从源数据库实例创建只读副本。源数据库实例将成为主数据库实例。对主数据库实例进行的更新将异步复制到只读副本。您可以将从应用程序发出的读取查询路由到只读副本，以减轻主数据库实例上的负载。利用只读副本，可以实现弹性扩展并超越单个数据库实例的容量限制，以处理高读取量的数据库工作负载。

当您从 RDS on Outposts 数据库实例创建只读副本时，该只读副本使用客户拥有的 IP 地址（CoIP）。有关更多信息，请参阅 客户拥有的适用于 Amazon RDS on Amazon Outposts 的 IP 地址。

RDS on Outposts 上的只读副本具有以下限制：

- 您无法为 RDS on Outposts 上的 RDS for SQL Server 数据库实例创建只读副本。
- RDS on Outposts 不支持跨区域只读副本。
- RDS on Outposts 不支持级联只读副本。
- 源 RDS on Outposts 数据库实例不能具有本地备份。源数据库实例的备份目标必须是您的 Amazon Web Services 区域。
- 只读副本需要客户拥有的 IP（CoIP）池。有关更多信息，请参阅 客户拥有的适用于 Amazon RDS on Amazon Outposts 的 IP 地址。
- 只能在与源数据库实例相同的虚拟私有云（VPC）中，在 RDS on Outposts 上创建只读副本。
- RDS on Outposts 上的只读副本可以位于与源数据库实例相同的 Outpost 上或同一 VPC 中的另一个 Outpost 上。
- 您无法为使用 Amazon KMS External Key Store (XKS) 加密的数据库实例创建只读副本。

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 从 RDS on Outposts 数据库实例创建只读副本。有关只读副本的更多信息，请参阅 使用数据库实例只读副本。

控制台

从源数据库实例创建只读副本

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。
3. 选择要作为只读副本源的数据库实例。

4. 对于 Actions (操作)，选择 Create read replica (创建只读副本)。
5. 对于数据库实例标识符，输入该只读副本的名称。
6. 为 Outposts Connectivity（Outposts 连接性）指定您的设置。这些设置适用于使用具有数据库实例的数据库子网组的虚拟私有云（VPC）的 Outpost。您的 VPC 必须基于 Amazon VPC 服务。
7. 选择您的 DB instance class（数据库实例类）。我们建议您为只读副本使用与源数据库实例相同或更大的数据库实例类和存储类型。
8. 对于 Multi-AZ deployment（多可用区部署），选择 Create a standby instance (recommended for production usage) [创建备用实例（推荐用于生产用途）]，以在另一个多可用区中创建备用数据库实例。

   创建您的只读副本作为多可用区数据库实例与源数据库是否为多可用区数据库实例无关。
9. （可选）在 Connectivity（连接）下，设置 Subnet Group（子网组）和 Availability Zone（可用区）的值。

   如果您为 Subnet Group（子网组）和 Availability Zone（可用区）指定值，则只读副本将在与数据库子网组中的可用区关联的 Outpost 上创建。

   如果您为 Subnet Group（子网组）指定一个值，并为 Availability Zone（可用区）指定 No preference（无首选项），则将在数据库子网组的随机 Outpost 上创建只读副本。
10. 对于 Amazon KMS key，选择 KMS 密钥的 Amazon KMS key 标识符。

    必须加密只读副本。
11. 根据需要选择其他选项。
12. 选择 Create read replica (创建只读副本)。

创建只读副本后，您可以在 RDS 控制台的数据库页面上看到它。它在角色列中显示副本。

Amazon CLI

要从源 MySQL 或 PostgreSQL 数据库实例创建只读副本，请使用 Amazon CLI 命令 [create-db-instance-read-replica]。

您可以通过指定 `--db-subnet-group-name` 和 `--availability-zone` 选项来控制创建只读副本的位置：

- 如果您同时指定 `--db-subnet-group-name` 和 `--availability-zone` 选项，则只读副本将在与数据库子网组中的可用区关联的 Outpost 上创建。

- 如果您指定 `--db-subnet-group-name` 选项但未指定 `--availability-zone` 选项，则只读副本将在数据库子网组中的随机 Outpost 上创建。
- 如果您未指定任一选项，则只读副本将在与源 RDS on Outposts 数据库实例相同的 Outpost 上创建。

以下示例创建了一个副本并通过包含 `--db-subnet-group-name` 和 `--availability-zone` 选项来指定只读副本的位置。

Example

对于 Linux、macOS 或 Unix：

```
aws rds create-db-instance-read-replica \
    --db-instance-identifier myreadreplica \
    --source-db-instance-identifier mydbinstance \
    --availability-zone us-west-2a
```

对于 Windows：

```
aws rds create-db-instance-read-replica ^
    --db-instance-identifier myreadreplica ^
    --source-db-instance-identifier mydbinstance ^
    --availability-zone us-west-2a
```

RDS API

要从源 MySQL 或 PostgreSQL 数据库实例创建只读副本，请使用以下必要的参数调用 Amazon RDS API CreateDBInstanceReadReplica 操作：

- `DBInstanceIdentifier`
- `SourceDBInstanceIdentifier`

您可以通过指定 `DBSubnetGroupName` 和 `AvailabilityZone` 参数来控制创建只读副本的位置：

- 如果您同时指定 `DBSubnetGroupName` 和 `AvailabilityZone` 参数，则只读副本将在与数据库子网组中的可用区关联的 Outpost 上创建。
- 如果您指定 `DBSubnetGroupName` 参数但未指定 `AvailabilityZone` 参数，则只读副本将在数据库子网组中的随机 Outpost 上创建。

- 如果您未指定任一参数，则只读副本将在与源 RDS on Outposts 数据库实例相同的 Outpost 上创建。

在 Amazon RDS on Amazon Outposts 上还原数据库实例的注意事项

在 Amazon RDS on Amazon Outposts 中还原数据库实例时，您通常可以为已还原数据库实例的自动备份和手动快照选择存储位置。

- 从手动数据库快照还原时，您可以将备份存储在父级 Amazon Web Services 区域中或在本地存储在您的 Outpost 中。
- 从自动备份（时间点恢复）还原时，您的选择更少：
  - 如果从父级 Amazon Web Services 区域还原，您可以将备份存储在 Amazon Web Services 区域或您的 Outpost 中。
  - 如果从 Outpost 中恢复，则只能在 Outpost 上存储备份。

的 Amazon RDS 代理

通过使用 Amazon RDS 代理，您可以允许您的应用程序池化和共享数据库连接，以提高其扩展能力。RDS 代理通过在保留应用程序连接的同时自动连接到备用数据库实例，使应用程序能够更好地抵御数据库故障。使用 RDS 代理还使您能够为数据库强制执行 Amazon Identity and Access Management (IAM) 身份验证，并将凭证安全地存储在 Amazon Secrets Manager。

使用 RDS 代理，您可以处理不可预测的数据库流量突增。否则，这些突增情况可能会由于超额订阅连接或快速创建新连接而导致问题。RDS 代理建立数据库连接池，并重用该池中的连接。这种方法可避免每次打开新数据库连接所产生的内存和 CPU 开销。要防止数据库超额订阅，您可以控制创建的数据库连接数。

RDS 代理对无法立即从连接池提供服务的应用程序连接进行排队或限制。尽管延迟可能会增加，但您的应用程序可以继续扩展，而不会导致数据库突然出现故障或不堪重负。如果连接请求超出您指定的限制，则 RDS 代理会拒绝应用程序连接（即，减轻负载）。与此同时，它为 RDS 可通过可用容量提供服务的负载维护可预测的性能。

您可以减少处理凭证的开销，并为每个新连接建立安全连接。RDS 代理可以代表数据库处理其中的一些工作。

RDS 代理与它支持的引擎版本完全兼容。您可以在不更改代码的情况下为大多数应用程序启用 RDS 代理。

主题

区域和版本可用性

功能可用性和支持因每个数据库引擎的特定版本以及 Amazon Web Services 区域而异。有关使用 RDS 代理的 Amazon RDS 的版本和区域可用性的更多信息，请参阅 支持 Amazon RDS 代理的区域和数据库引擎。

RDS 代理的配额和限制

以下配额和限制适用于 RDS 代理：

- 每个 Amazon Web Services 账户 ID 限制为 20 个代理。如果您的应用程序需要更多代理，请通过 Amazon Web Services Management Console中的服务配额页面申请增加代理。在服务配额页面中，选择 Amazon Relational Database Service（Amazon RDS），然后找到代理以申请增加配额。Amazon 可以自动增加您的配额，或者等待 Amazon Web Services 支持 审核您的申请。
- 每个代理最多可拥有 200 个关联的 Secrets Manager 密钥。因此，每个代理可以在任何给定时间连接多达 200 个不同的用户账户。
- 每个代理都具有一个默认端点。您还可以为每个代理添加最多 20 个代理端点。您可以创建、查看、修改和删除这些端点。
- 对于复制配置中的 RDS 数据库实例，您只能将代理与写入器数据库实例关联，而不能与只读副本关联。
- 您的 RDS 代理必须与数据库位于同一 Virtual Private Cloud (VPC) 中。代理无法公开访问，但数据库可以。例如，如果在本地主机上进行数据库原型设计，则无法连接到代理，除非设置必要的网络要求来允许连接到代理。这是因为您的本地主机不在代理的 VPC 范围内。
- 您不能将 RDS 代理用于其租户设置为 `dedicated` 的 VPC。
- 如果您将 RDS 代理与启用了 IAM 身份验证的 RDS 数据库实例结合使用，请检查用户身份验证。通过代理连接的用户必须通过登录凭证进行身份验证。有关 RDS 代理中 Secrets Manager 和 IAM 支持的详细信息，请参阅在 Amazon Secrets Manager 中设置 RDS 代理的数据库凭证和为 RDS 代理配置 IAM 身份验证。
- 使用 SSL 主机名验证时，不能将 RDS 代理与自定义 DNS 一起使用。
- 每个代理都可以与单个目标数据库实例相关联。不过，您可以将多个代理与同一个数据库实例关联。
- 文本大小大于 16 KB 的任何语句都会导致代理将会话固定到当前连接。
- 某些区域在创建代理时需要考虑可用区（AZ）限制。美国东部（弗吉尼亚州北部）区域在 `use1-az3` 可用区中不支持 RDS 代理。美国西部（加利福尼亚北部）区域在 `usw1-az2` 可用区中不支持 RDS 代理。如果在创建代理时选择子网，请确保不要在上述可用区中选择子网。
- 目前，RDS 代理不支持任何全局条件上下文键。

有关全局条件上下文键的更多信息，请参阅《IAM 用户指南》中的 Amazon 全局条件上下文键。

- 不能将 RDS 代理与 RDS Custom for SQL Server 一起使用。
- 要将任何数据库参数组修改反映到您的代理，需要重启实例，即使您选择立即应用更改也不例外。对于集群级参数，需要在集群范围内重启。
- 当注册代理目标时，您的代理会自动创建 `rdsproxyadmin` 数据库用户。这是受保护的用户，对于代理功能至关重要。您应该避免以任何身份篡改 `rdsproxyadmin` 用户。删除或修改 `rdsproxyadmin` 用户或其权限可能导致应用程序完全无法使用代理。

有关每个数据库引擎的更多信息，请参阅下面几节：

- RDS for MariaDB S 的额外限制
- RDS for Microsoft SQL Server 的额外限制
- RDS for MySQL 的额外限制
- RDS for PostgreSQL 的额外限制

RDS for MariaDB S 的额外限制

以下附加限制应用于适用于 RDS for MariaDB 数据库的 RDS 代理：

- 目前，所有代理均在端口 3306 上侦听 MariaDB。代理仍使用您在数据库设置中指定的端口连接到您的数据库。
- 不能将 RDS 代理与 Amazon EC2 实例中自行管理的 MariaDB 数据库结合使用。
- 您不能将 RDS 代理与已将数据库参数组中的 `read_only` 参数设置为 1 的 RDS for MariaDB 数据库实例一起使用。
- RDS 代理不支持 MariaDB 压缩模式。例如，它不支持 `mysql` 命令的 `--compress` 或 `-C` 选项使用的压缩。
- 某些 SQL 语句和函数可以更改连接状态而不会引起固定。有关最新的固定行为，请参阅 避免固定 RDS 代理。
- RDS 代理不支持 MariaDB `auth_ed25519` 插件。
- RDS 代理对于 MariaDB 数据库不支持传输层安全性协议（TLS）版本 1.3。
- 当 RDS 代理重用相同的数据库连接运行其他查询时，处理 `GET DIAGNOSTIC` 命令的数据库连接可能会返回不准确的信息。当 RDS 代理多路复用数据库连接时，可能会发生这种情况。有关更多信息，请参阅 RDS 代理概念概述。

- 对于 MariaDB，RDS 代理目前不支持 ClientPasswordAuthType 的 caching_sha2_password 选项。

> **Important**
>
> 对于与 MariaDB 数据库关联的代理，请勿在初始化查询中将配置参数 sql_auto_is_null 设置为 true 或非零值。否则，可能会导致不正确的应用程序行为。

RDS for Microsoft SQL Server 的额外限制

以下附加限制应用于适用于 RDS for Microsoft SQL Server 数据库的 RDS 代理：

- 您需要为代理创建的 Secrets Manager 密钥数量取决于您的数据库实例使用的排序规则。例如，假设您的数据库实例使用区分大小写的排序规则。如果您的应用程序同时接受“Admin”和“admin”，则代理需要两个单独的密钥。有关 SQL Server 中的排序规则的更多信息，请参阅 Microsoft SQL Server 文档。
- RDS 代理不支持使用 Active Directory 的连接。
- 您不能对不支持令牌属性的客户端使用 IAM 身份验证。有关更多信息，请参阅使用 Microsoft SQL Server 连接到代理的注意事项。
- @@IDENTITY、@@ROWCOUNT 和 SCOPE_IDENTITY 的结果并非始终准确。作为一种解决方法，可在同一会话语句中检索它们的值，以确保它们返回正确的信息。
- 如果连接使用多个活动结果集（MARS），则 RDS 代理不会运行初始化查询。有关 MARS 的更多信息，请参阅 Microsoft SQL Server 文档。
- 目前，RDS 代理不支持在主要版本 SQL Server 2022 上运行的 RDS for SQL Server 数据库实例。
- RDS 代理不支持在 SQL Server 2014 主版本上运行的 RDS for SQL Server 数据库实例。
- RDS 代理不支持无法在一个 TLS 记录中处理多条响应消息的客户端应用程序。

RDS for MySQL 的额外限制

以下附加限制应用于适用于 RDS for MySQL 数据库的 RDS 代理：

- RDS 代理对 caching_sha2_password 身份验证的支持需要安全（TLS）连接。
- 对 caching_sha2_password 的 RDS 代理支持已知与某些 go-sql 驱动程序版本存在兼容性问题。

- 使用 MySQL 8.4 C 驱动程序时，如果预编译语句中的参数计数超过占位符计数，mysql_stmt_bind_named_param API 可能会形成格式错误的数据包。这会导致错误的响应。有关更多信息，请参阅 MySQL 错误报告。
- 目前，所有代理均在端口 3306 上侦听 MySQL。代理仍使用您在数据库设置中指定的端口连接到您的数据库。
- 不能将 RDS 代理与 EC2 实例中自管理的 MySQL 数据库一起使用。
- 您不能将 RDS 代理与数据库参数组中的 read_only 参数设置为 1 的 RDS for MySQL 数据库实例一起使用。
- RDS 代理不支持 MySQL 压缩模式。例如，它不支持 mysql 命令的 --compress 或 -C 选项使用的压缩。
- 当 RDS 代理重用相同的数据库连接运行其他查询时，处理 GET DIAGNOSTIC 命令的数据库连接可能会返回不准确的信息。当 RDS 代理多路复用数据库连接时，可能会发生这种情况。
- 某些 SQL 语句和函数（如 SET LOCAL）可以更改连接状态而不会引起固定。有关最新的固定行为，请参阅 避免固定 RDS 代理。
- 不支持在多语句查询中使用 ROW_COUNT() 函数。
- RDS 代理不支持无法在一个 TLS 记录中处理多条响应消息的客户端应用程序。

> **Important**
>
> 对于与 MySQL 数据库关联的代理，请勿在初始化查询中将配置参数 sql_auto_is_null 设置为 true 或非零值。否则，可能会导致不正确的应用程序行为。

RDS for PostgreSQL 的额外限制

以下附加限制应用于适用于 RDS for PostgreSQL 数据库的 RDS 代理：

- RDS 代理不支持 PostgreSQL 的会话固定筛选条件。
- 目前，所有代理均在端口 5432 上侦听 PostgreSQL。
- 对于 PostgreSQL，RDS 代理目前不支持通过发出 CancelRequest 来取消来自客户端的查询。例如，使用 Ctrl+C 取消交互式 psql 会话中长时间运行的查询时，就会出现这种情况。
- PostgreSQL 函数 lastval 的结果并不总是准确的。作为解决方法，请将 INSERT 语句与 RETURNING 子句一起使用。

- RDS 代理目前不支持流式复制模式。
- 在 RDS for PostgreSQL 16 中，对 `scram_iterations` 值的修改只会影响代理和数据库之间的身份验证过程。具体而言，如果您将 `ClientPasswordAuthType` 配置为 `scram-sha-256`，则对 `scram_iterations` 值所做的任何自定义都不会影响客户端到代理密码身份验证。相反，客户端到代理密码身份验证的迭代值固定为 4096。
- `default` 数据库必须存在。
- 如果您使用 `ALTER ROLE` 通过 `SET ROLE` 更改用户角色，但后续以该用户身份连接到代理时若遇到固定问题，则这些后续连接可能不会使用此角色设置。为防止出现这种情况，在使用代理时，请在代理的初始化查询中使用 `SET ROLE`。有关更多信息，请参阅 为 Amazon RDS 创建代理中的初始化查询。

> **Important**
>
> 对于适用于 PostgreSQL 数据库的现有代理，如果您将数据库身份验证修改为仅使用 `SCRAM`，代理将在长达 60 秒的时间内不可用。要避免此问题，请执行以下操作之一：
>
> - 确保数据库同时允许 `SCRAM` 和 `MD5` 身份验证。
> - 要仅使用 `SCRAM` 身份验证，请创建一个新代理，将应用程序流量迁移到新代理，然后删除先前与数据库关联的代理。

规划在哪里使用 RDS 代理

您可以确定哪些数据库实例、集群和应用程序可能从使用 RDS Proxy 中受益匪浅。为此，请考虑以下因素：

- 任何出现“连接过多”错误的数据库实例都很适合与代理相关联。这通常以 `ConnectionAttempts` CloudWatch 指标的值较高为特征。代理允许应用程序打开许多客户端连接，而代理管理与数据库实例的少量长期连接。
- 对于使用较小 Amazon 实例类（如 T2 或 T3）的数据库实例，使用代理可以帮助避免内存不足情况。它还可以帮助减少建立连接时的 CPU 开销。处理大量连接时可能会发生这些情况。
- 您可以监控某些 Amazon CloudWatch 指标，以确定数据库实例是否接近某些类型的限制。这些限制针对连接数以及与连接管理关联的内存。您还可以监控某些 CloudWatch 指标，以确定数据库实例是否正在处理许多短期连接。打开和关闭此类连接可能会给数据库带来性能开销。有关要监控的指标的信息，请参阅 使用 Amazon CloudWatch 监控 RDS Proxy 指标。

- Amazon Lambda 函数也可以很好地使用代理。这些函数进行频繁的短数据库连接，可受益于 RDS Proxy 提供的连接池。您可以利用已为 Lambda 函数提供的任何 IAM 身份验证，而不是在 Lambda 应用程序代码中管理数据库凭证。
- 代理服务器非常适合那些通常需要打开和关闭大量数据库连接，并且没有内置的连接池机制的应用程序。
- 长时间保持大量连接打开的应用程序通常可以很好地使用代理。诸如软件即服务 (SaaS) 或电子商务等行业中的应用程序通常会使连接保持打开状态，从而最大限度地减少数据库请求的延迟。通过 RDS 代理，与直接连接到数据库实例相比，应用程序可以将更多连接保持为打开状态。
- 由于为所有数据库实例设置 IAM 身份验证和 Secrets Manager 较为复杂，因此，您可能尚未采用此类身份验证。如果是这样，您可以保留现有的身份验证方法，并将身份验证委派给代理。代理可以为特定应用程序的客户端连接强制执行身份验证策略。您可以利用已为 Lambda 函数提供的任何 IAM 身份验证，而不是在 Lambda 应用程序代码中管理数据库凭证。
- RDS 代理可以帮助使应用程序更具弹性，对数据库故障更透明。RDS 代理绕过域名系统（DNS）缓存，以将 Amazon RDS 多可用区数据库实例的故障转移时间缩短多达 66%。RDS 代理还会自动将流量路由到新的数据库实例，同时保留应用程序连接。这使得失效转移对应用程序更加透明。

RDS 代理概念和术语

您可以通过使用 RDS 代理来简化 Amazon RDS 数据库实例的连接管理。

RDS 代理处理客户端应用程序和数据库之间的网络流量。为实现这一点，它首先了解数据库协议。然后，它会根据应用程序中的 SQL 操作和数据库中的结果集来调整其行为。

RDS 代理减少了数据库上的连接管理的内存和 CPU 开销。当应用程序打开多个同时连接时，数据库所需的内存和 CPU 资源较少。而且也不需要应用程序中的逻辑来关闭和重新打开长时间处于空闲状态的连接。同样，在出现数据库问题时，需要较少的应用程序逻辑来重新建立连接。

RDS 代理的基础设施高度可用，部署在多个可用区 (AZ) 上。RDS 代理的计算、内存和存储独立于您的 RDS 数据库实例。这种分离有助于减少数据库服务器的开销，使它们能够将资源用于处理数据库工作负载。RDS 代理计算资源是无服务器的，根据您的数据库工作负载自动进行扩展。

主题

- 事务

RDS 代理概念概述

RDS 代理处理基础设施以执行连接池和以下各节中所述的其他功能。您可以在代理页面上看到 RDS 控制台中表示的代理。

每个代理处理与单个 RDS 数据库实例的连接。对于 RDS 多可用区实例或集群，代理自动确定当前的写入器实例。

代理保持打开状态且可供数据库应用程序用于形成连接池的连接。

默认情况下，RDS 代理可在会话中的每个事务之后重用连接。这种事务级别的重用称为多路复用。当 RDS 代理临时从连接池中删除某个连接以重用该连接时，该操作称为借用连接。如果这样做是安全的，RDS 代理会将该连接返回到连接池。

在某些情况下，RDS 代理不能确定在当前会话之外重用数据库连接是否安全。在这些情况下，它将会话保持在同一个连接上，直到会话结束。此回退行为称为固定。

代理具有默认端点。使用 Amazon RDS 数据库实例时，您将连接到该端点。您可以这样做，而不是连接到与实例直接连接的读/写端点。对于 RDS 数据库集群，您还可以创建其它读/写和只读端点。有关更多信息，请参阅 代理终端节点概述。

例如，您仍可连接到集群端点以实现读/写连接，而无需连接池。您仍可连接到读取器端点以实现负载均衡的只读连接。您仍可连接到实例端点，以便对集群中的特定数据库实例进行诊断和故障排除。如果您使用其他 Amazon 服务（例如 Amazon Lambda）连接到 RDS 数据库，请将其连接设置更改为使用代理端点。例如，指定代理端点，以允许 Lambda 函数在利用 RDS 代理功能的同时访问数据库。

每个代理都包含一个目标组。该目标组包含代理可连接到的 RDS 数据库实例。与代理关联的 RDS 数据库实例称为该代理的目标。为方便起见，当您通过控制台创建代理时，RDS 代理也会创建相应的目标组并自动注册关联的目标。

引擎系列是使用相同数据库协议的相关数据库引擎集。您可以为创建的每个代理选择引擎系列。

连接池

每个代理都为其关联的 RDS 数据库 的写入器和读取器实例建立连接池。连接池 是一项优化操作，可减少与打开和关闭连接相关的开销，并可保持许多连接同时打开。该开销包括处理每个新连接所需的内存。关闭每个连接并打开一个新连接还涉及 CPU 开销。示例包括传输层安全性协议/安全套接字层

（TLS/SSL）握手、身份验证、协商功能等。连接池可简化应用程序逻辑。无需编写应用程序代码，即可最大限度地减少同时打开的连接数量。

每个代理还执行连接多路复用，也称为连接重用。对于多路复用，RDS 代理使用一个底层数据库连接对事务执行所有操作。然后，RDS 可以对下一个事务使用不同的连接。您可以同时打开许多到代理的连接，而代理可确保打开的到数据库实例或集群的连接数量较少。这样做可进一步减少数据库服务器上连接的内存开销。这项技术还降低了出现"连接过多"错误的可能性。

RDS 代理安全性

RDS 代理使用现有的 RDS 安全机制，如 TLS/SSL 和 Amazon Identity and Access Management (IAM)。有关这些安全功能的一般信息，请参阅 Amazon RDS 中的安全性。此外，请务必熟悉 RDS 如何处理身份验证、授权和其它安全领域。

RDS 代理可作为客户端应用程序和底层数据库之间的附加安全层。例如，即使底层数据库实例支持旧版 TLS，您也可以使用 TLS 1.3 连接到代理。您可以使用 IAM 角色连接到代理。即使代理使用本机用户和密码身份验证方法连接到数据库，也是如此。通过采用这项技术，您可以对数据库应用程序强制执行强身份验证要求，而无需对数据库实例本身进行成本高昂的迁移。

您将 RDS 代理使用的数据库凭证存储在 Amazon Secrets Manager 中。代理访问的 RDS 数据库实例的每个数据库用户都必须在 Secrets Manager 中拥有相应的密钥。您还可为 RDS 代理的用户设置 IAM 身份验证。这样一来，即使数据库使用本机密码身份验证，您也可以对数据库访问强制执行 IAM 身份验证。我们建议使用这些安全功能，而不是在应用程序代码中嵌入数据库凭证。

将 TLS/SSL 与 RDS 代理结合使用

您可以使用 TLS/SSL 协议连接到 RDS Proxy。

> **Note**
>
> RDS 代理使用来自 Amazon Certificate Manager (ACM) 的证书。如果您正在使用 RDS Proxy，则无需下载 Amazon RDS 证书或更新使用 RDS 代理连接的应用程序。

要对代理与数据库之间的所有连接强制执行 TLS，您可以在 Amazon Web Services Management Console中创建或修改代理时，指定需要传输层安全性设置。

RDS 代理还可以确保您的会话在客户端与 RDS 代理端点之间使用 TLS/SSL。要让 RDS 代理这样做，请在客户端上指定此要求。没有为使用 RDS 代理的数据库 SSL 连接设置 SSL 会话变量。

- 对于 RDS for MySQL，在运行 `mysql` 命令时使用 `--ssl-mode` 参数在客户端上指定此要求。

- 对于 Amazon RDS PostgreSQL，在运行 `psql` 命令时将 `sslmode=require` 指定为 `conninfo` 字符串的一部分。

RDS 代理支持 TLS 协议版本 1.0、1.1、1.2 和 1.3。您可以使用比底层数据库中使用的更高版本的 TLS 连接到代理。

默认情况下，客户端程序会与 RDS 代理建立加密连接，并通过 `--ssl-mode` 选项进行进一步控制。从服务器端，RDS 代理支持所有 SSL 模式。

对于客户端，SSL 模式如下：

PREFERRED

SSL 是首选项，但不是必需项。

DISABLED

不允许使用 SSL。

REQUIRED

强制 SSL。

VERIFY_CA

强制 SSL 并验证证书颁发机构 (CA)。

VERIFY_IDENTITY

强制 SSL 并验证 CA 和 CA 主机名。

将客户端与 `--ssl-modeVERIFY_CA` 或 `VERIFY_IDENTITY` 结合使用时，以 `--ssl-ca` 格式指定指向 CA 的 `.pem` 选项。对于要使用的 `.pem` 文件，请从 Amazon Trust Services 下载所有根 CA PEM，然后把它们放入一个 `.pem` 文件。

RDS 代理使用通配符证书，这些证书应用到域及其子域。如果您使用 `mysql` 客户端以 SSL 模式 `VERIFY_IDENTITY` 进行连接，则您当前必须使用与 MySQL 8.0 兼容的 `mysql` 命令。

失效转移

失效转移是一项高可用性功能，可在原始实例变得不可用时将数据库实例替换为另一个数据库实例。可能会因为数据库实例出现问题而发生失效转移。失效转移也可能是正常维护程序的一部分，例如在数据库升级期间。失效转移适用于多可用区配置中的 RDS 数据库实例。

通过代理进行连接可使您的应用程序对数据库失效转移更具弹性。当原始数据库实例变得不可用时，RDS 代理将连接到备用数据库，而不删除空闲应用程序连接。这有助于加快和简化失效转移过程。这与典型的重启或数据库问题相比，对应用程序造成的干扰更小。

如果没有 RDS Proxy，失效转移会导致短暂中断。在中断期间，您无法在失效转移中对该数据库执行写入操作。任何现有数据库连接都会中断，您的应用程序必须重新打开这些连接。当一个只读数据库实例被提升以替代不可用的数据库实例时，数据库将可用于新连接和写入操作。

在数据库失效转移期间，RDS Proxy 将继续接受相同 IP 地址的连接，并自动将连接定向到新的主数据库实例。通过 RDS 代理连接的客户端不会受到以下情况的影响：

- 失效转移时的域名系统 (DNS) 传播延迟。
- 本地 DNS 缓存。
- 连接超时。
- 不确定哪个数据库实例是当前的写入器。
- 等待来自以前写入器的查询响应，该写入器在未关闭连接的情况下变得不可用。

对于维护自身连接池的应用程序，完成 RDS Proxy 意味着大多数连接在失效转移或其他中断期间保持活动状态。只有处于事务或 SQL 语句中间的连接才会被取消。RDS 代理会立即接受新连接。当数据库写入器不可用时，RDS 代理将对传入的请求进行排队。

对于不维护自身连接池的应用程序，RDS 代理提供更快的连接速度和更多的打开连接。它减轻了频繁从数据库重新连接而产生的昂贵开销。这是通过重用 RDS 代理连接池中维护的数据库连接来实现。此方法对于 TLS 连接尤其重要，因为这些连接的设置成本很高。

事务

单个事务中的所有语句始终使用相同的底层数据库连接。当事务结束时，连接可供另一会话使用。使用事务作为粒度单位会产生以下后果：

- 开启 RDS for MySQL `autocommit` 设置后，每个单独的语句之后都会发生连接重用。
- 相反，关闭 `autocommit` 设置后，您在会话中发出的第一个语句会开始新的事务。例如，假设您输入 `SELECT`、`INSERT`、`UPDATE` 序列和其他数据操作语言（DML）语句。在这种情况下，在您发出 `COMMIT`、`ROLLBACK` 或以其他方式结束事务之前，不会发生连接重用。
- 输入数据定义语言 (DDL) 语句会导致事务在该语句完成后结束。

RDS 代理通过数据库客户端应用程序使用的网络协议检测事务何时结束。事务检测不依赖于关键字，例如 SQL 语句文本中显示的 `COMMIT` 或 `ROLLBACK`。

在某些情况下，RDS 代理可能会检测到使得无法将会话移动到其他连接的数据库请求。在这些情况下，它会在会话的剩余时间关闭该连接的多路复用。当 RDS 代理不能确定多路复用是否适用于会话时，这一规则也适用。该操作称为固定。有关检测和最小化固定的方法，请参阅 避免固定 RDS 代理。

开始使用 RDS 代理

使用以下页面中的信息来设置、管理 的 Amazon RDS 代理以及设置相关的安全选项。这些安全选项控制哪些人可以访问每个代理，以及每个代理如何连接到数据库实例。

如果您不熟悉 RDS 代理，我们建议您按照我们展示页面的顺序进行操作。

主题

- 设置 RDS 代理的网络先决条件
- 在 Amazon Secrets Manager 中设置 RDS 代理的数据库凭证
- 为 RDS 代理配置 IAM 身份验证
- 为 Amazon RDS 创建代理
- 查看代理
- 通过 RDS Proxy 连接到数据库

设置 RDS 代理的网络先决条件

使用 RDS 代理需要您在 RDS 数据库实例和 RDS 代理之间拥有通用的虚拟私有云（VPC）。此 VPC 应至少有两个位于不同可用区中的子网。您的账户可以拥有这些子网，或与其他账户共享它们。有关 VPC 共享的信息，请参阅使用共享 VPC。

您的客户端应用程序资源（例如 Amazon EC2、Lambda 或 Amazon ECS）可以与代理位于同一 VPC 中。它们也可以位于与代理不同的 VPC 中。如果已成功连接到任何 RDS 数据库实例，则您已拥有所需的网络资源。

主题

- 获取有关您的子网的信息
- 计划 IP 地址容量

获取有关您的子网的信息

要创建代理，必须提供代理在其中运行的子网和 VPC。以下 Linux 示例展示了检查 Amazon Web Services 账户 拥有的 VPC 和子网的 Amazon CLI 命令。尤其是，在使用 CLI 创建代理时，您可以将子网 ID 作为参数传递。

```
aws ec2 describe-vpcs
aws ec2 describe-internet-gateways
aws ec2 describe-subnets --query '*[].[VpcId,SubnetId]' --output text | sort
```

以下 Linux 示例展示了用于确定与特定 RDS 数据库实例对应的子网 ID 的 Amazon CLI 命令。找到数据库实例的 VPC ID。检查 VPC 以查找其子网。以下 Linux 示例展示了操作步骤。

```
$ #From the DB instance, trace through the DBSubnetGroup and Subnets to find the subnet
 IDs.
$ aws rds describe-db-instances --db-instance-identifier my_instance_id --query '*[].
[DBSubnetGroup]|[0]|[0]|[Subnets]|[0]|[*].SubnetIdentifier' --output text
```

```
subnet_id_1
subnet_id_2
subnet_id_3
...
```

```
$ #From the DB instance, find the VPC.
$ aws rds describe-db-instances --db-instance-identifier my_instance_id --query '*[].
[DBSubnetGroup]|[0]|[0].VpcId' --output text
```

```
my_vpc_id
```

```
$ aws ec2 describe-subnets --filters Name=vpc-id,Values=my_vpc_id --query '*[].
[SubnetId]' --output text
```

```
subnet_id_1
subnet_id_2
subnet_id_3
subnet_id_4
subnet_id_5
subnet_id_6
```

计划 IP 地址容量

RDS 代理会根据向它注册的数据库实例的大小和数量，视需要自动调整其容量。某些操作可能还需要更多代理容量，例如增加注册数据库的大小或内部 RDS 代理维护操作。在这些操作期间，您的代理可能需要更多 IP 地址来预调配额外的容量。这些额外的地址使您的代理可以在不影响工作负载的情况下进行扩展。您的子网中缺少可用的 IP 地址会阻止代理纵向扩展。这可能导致查询延迟更长或客户端连接故障。当您的子网中没有足够的可用 IP 地址时，RDS 会通过事件 `RDS-EVENT-0243` 通知您。有关此事件的信息，请参阅 使用 RDS 代理事件。

根据数据库实例类大小，您应在子网中为代理保留以下最少可用的 IP 地址数。

数据库实例类	最少可用 IP 地址数
db.*.xlarge 或更小	10
db.*.2xlarge	15
db.*.4xlarge	25
db.*.8xlarge	45
db.*.12xlarge	60
db.*.16xlarge	75
db.*.24xlarge	110

这些建议的 IP 地址数是针对仅具有默认端点的代理的估计数。具有更多端点或只读副本的代理可能需要更多可用的 IP 地址。对于每个其他端点，建议您另外预留三个 IP 地址。对于每个只读副本，建议您根据该只读副本的大小，保留表中指定的额外 IP 地址。

> **Note**
>
> RDS 代理在一个 VPC 中支持的 IP 地址数不超过 215 个。

在 Amazon Secrets Manager 中设置 RDS 代理的数据库凭证

Amazon RDS 中的 RDS 代理使用 Amazon Secrets Manager 安全地存储和管理数据库凭证。您无需在应用程序中嵌入凭证，而是将代理与包含必要身份验证详细信息的 Secrets Manager 密钥相关联。您可以在 RDS 数据库实例上，为代理连接到的每个数据库用户账户分别创建 Secrets Manager 密钥。

主题

创建与 RDS 代理结合使用的密钥

在创建代理之前，必须先创建至少一个用于存储数据库凭证的密钥。

控制台

创建密钥

1. 通过 https://console.aws.amazon.com/secretsmanager/ 打开 Secrets Manager 控制台。
2. 选择存储新密钥。
3. 选择 Amazon RDS 数据库的凭证。
4. 输入用户名和密码。您输入的凭证必须与关联的 RDS 数据库中存在的数据库用户的凭证相匹配。RDS 代理使用这些凭证代表应用程序对数据库进行身份验证和建立连接。

   如果不匹配，您可以更新密钥来与数据库密码匹配。在更新密钥之前，尝试使用该密钥通过代理进行连接会失败，但使用其它有效密钥的连接仍然有效。

   > **Note**
   >
   > 对于 RDS For SQL Server，无论数据库实例排序规则设置如何，RDS 代理都需要在 Secrets Manager 中设置一个区分大小写的密钥。如果应用程序支持用户名使用不同的大小写，例如“Admin”和“admin”，则必须为每个用户名创建单独的密钥。RDS 代理不支持在客户端和代理之间进行不区分大小写的用户名身份验证。
   > 有关 SQL Server 中的排序规则的更多信息，请参阅 Microsoft SQL Server 文档。

5. 对于数据库，选择密钥将访问的 Amazon RDS 数据库。
6. 填写密钥的其它设置，然后选择存储。有关全面的说明，请参阅《Amazon Secrets Manager User Guide》中的 Creating an Amazon Secrets Manager secret。

Amazon CLI

通过 Amazon CLI 创建代理时，请指定相应密钥的 Amazon 资源名称（ARN）。请为代理可以访问的所有数据库用户账户执行此操作。在 Amazon Web Services Management Console 中，根据描述性名称选择密钥。

- 要创建与 RDS 代理结合使用的 Secrets Manager 密钥，请使用 create-secret 命令：

```
aws secretsmanager create-secret \
  --name "secret_name" \
  --description "secret_description" \
  --region region_name \
  --secret-string '{"username":"db_user","password":"db_user_password"}'
```

- 您也可以创建自定义密钥来加密 Secrets Manager 密钥。以下命令创建一个密钥示例。

```
aws kms create-key --description "test-key" --policy '{
  "Id":"kms-policy",
  "Version":"2012-10-17",
  "Statement":
    [
      {
        "Sid":"Enable IAM User Permissions",
        "Effect":"Allow",
        "Principal":{"AWS":"arn:aws:iam::account_id:root"},
        "Action":"kms:*","Resource":"*"
      },
      {
        "Sid":"Allow access for Key Administrators",
        "Effect":"Allow",
        "Principal":
          {
            "AWS":
              ["$USER_ARN","arn:aws:iam:account_id::role/Admin"]
          },
        "Action":
          [
            "kms:Create*",
            "kms:Describe*",
            "kms:Enable*",
            "kms:List*",
            "kms:Put*",
            "kms:Update*",
```

```
                "kms:Revoke*",
                "kms:Disable*",
                "kms:Get*",
                "kms:Delete*",
                "kms:TagResource",
                "kms:UntagResource",
                "kms:ScheduleKeyDeletion",
                "kms:CancelKeyDeletion"
              ],
            "Resource":"*"
          },
          {
            "Sid":"Allow use of the key",
            "Effect":"Allow",
            "Principal":{"AWS":"$ROLE_ARN"},
            "Action":["kms:Decrypt","kms:DescribeKey"],
            "Resource":"*"
          }
        ]
}'
```

例如，以下命令为两个数据库用户创建 Secrets Manager 密钥：

```
aws secretsmanager create-secret \
  --name secret_name_1 --description "db admin user" \
  --secret-string '{"username":"admin","password":"choose_your_own_password"}'

aws secretsmanager create-secret \
  --name secret_name_2 --description "application user" \
  --secret-string '{"username":"app-user","password":"choose_your_own_password"}'
```

要创建使用自定义 Amazon KMS 密钥加密的这些密钥，请使用以下命令：

```
aws secretsmanager create-secret \
  --name secret_name_1 --description "db admin user" \
  --secret-string '{"username":"admin","password":"choose_your_own_password"}' \
  --kms-key-id arn:aws:kms:us-east-2:account_id:key/key_id

aws secretsmanager create-secret \
  --name secret_name_2 --description "application user" \
  --secret-string '{"username":"app-user","password":"choose_your_own_password"}' \
```

```
  --kms-key-id arn:aws:kms:us-east-2:account_id:key/key_id
```

要查看您的 Amazon 账户拥有的密钥，请使用 list-secrets 命令：

```
aws secretsmanager list-secrets
```

使用 CLI 创建代理时，将一个或多个密钥的 Amazon Resource Name (ARN) 传递给了 `--auth` 参数。以下示例展示了如何准备报告，其中仅包含 Amazon 账户所拥有的每个密钥的名称和 ARN。此示例使用了 `--output table` 版本 2 中提供的 Amazon CLI 参数。如果您使用的是 Amazon CLI 版本 1，请改用 `--output text`。

```
aws secretsmanager list-secrets --query '*[].[Name,ARN]' --output table
```

要确认密钥包含格式正确的凭证，请使用 get-secret-value 命令。将 *`your_secret_name`* 替换为密钥的短名称或 ARN。

```
aws secretsmanager get-secret-value --secret-id your_secret_name
```

输出包含一行 JSON 编码值，类似于以下内容：

```
...
"SecretString": "{\"username\":\"your_username\",\"password\":\"your_password\"}",
...
```

为 RDS 代理配置 IAM 身份验证

要在 Amazon RDS 中为 RDS 代理设置 Amazon Identity and Access Management（IAM）身份验证，请创建并配置用于授予必要权限的 IAM 策略。RDS 代理使用 Amazon Secrets Manager 安全地管理数据库凭证，这使应用程序无需直接处理凭证即可通过代理进行身份验证。

本主题提供了为 RDS 代理配置 IAM 身份验证的步骤，包括创建所需的 IAM 策略并将其附加到 IAM 角色。

> **Tip**
>
> 仅当您想创建自己的 IAM 角色时，才需要执行此过程。否则，RDS 可以在您设置代理时自动创建所需的角色，因此您可以跳过这些步骤。

先决条件

在为 RDS 代理设置 IAM 身份验证之前，请确保您具有以下各项：

- Amazon Secrets Manager：至少一个包含数据库凭证的存储密钥。有关创建密钥的说明，请参阅the section called “设置数据库凭证”。
- IAM 权限：具有在 Amazon Secrets Manager 中创建和管理 IAM 策略、角色和密钥的权限的 IAM 角色或用户。

创建用于访问 Secrets Manager 的 IAM 策略

要支持 RDS 代理从 Secrets Manager 检索数据库凭证，请创建一个 IAM 角色，并使用授予必要权限的策略。

控制台

创建用于访问您的密钥以便与代理一起使用的角色

1. 登录 Amazon Web Services Management Console，然后打开 IAM 控制台：https://console.aws.amazon.com/iam/。
2. 为该角色创建权限策略。有关一般步骤，请参阅创建 IAM 策略（控制台）。

   将此策略粘贴到 JSON 编辑器中，并进行以下更改：

   - 替换您自己的账户 ID。
   - 将 `us-east-2` 替换为代理所在的区域。
   - 将密钥名称替换为您创建的名称。有关更多信息，请参阅 Specifying KMS keys in IAM policy statements。
   - 将 KMS 密钥 ID 替换为您用于加密 Secrets Manager 密钥（默认密钥或您自己的密钥）的密钥 ID。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "secretsmanager:GetSecretValue",
            "Resource": [
```

```
                "arn:aws:secretsmanager:us-east-2:account_id:secret:secret_name_1",
                "arn:aws:secretsmanager:us-east-2:account_id:secret:secret_name_2"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "kms:Decrypt",
            "Resource": "arn:aws:kms:us-east-2:account_id:key/key_id",
            "Condition": {
                "StringEquals": {
                    "kms:ViaService": "secretsmanager.us-east-2.amazonaws.com"
                }
            }
        }
    ]
}
```

3. 创建角色并向其附加权限策略。有关一般步骤，请参阅创建向 Amazon 服务委派权限的角色。

   对于可信实体类型，选择 Amazon 服务。在用例下，选择 RDS，然后为用例选择 RDS - 向数据库添加角色。

4. 对于权限策略，选择您创建的策略。

5. 对于选择可信实体，为该角色输入以下信任策略：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "rds.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Amazon CLI

要使用 Amazon CLI 创建角色，请发送以下请求：

```
aws iam create-role \
  --role-name my_role_name \
  --assume-role-policy-document '{"Version":"2012-10-17","Statement":
[{"Effect":"Allow","Principal":{"Service":
["rds.amazonaws.com"]},"Action":"sts:AssumeRole"}]}'
```

然后，将策略附加到该角色：

```
aws iam put-role-policy \
  --role-name my_role_name \
  --policy-name secret_reader_policy \
  --policy-document '{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": "secretsmanager:GetSecretValue",
            "Resource": [
                "arn:aws:secretsmanager:us-east-2:account_id:secret:secret_name_1",
                "arn:aws:secretsmanager:us-east-2:account_id:secret:secret_name_2"
            ]
        },
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": "kms:Decrypt",
            "Resource": "arn:aws:kms:us-east-2:account_id:key/key_id",
            "Condition": {
                "StringEquals": {
                    "kms:ViaService": "secretsmanager.us-east-2.amazonaws.com"
                }
            }
        }
    ]
}'
```

配置了 IAM 角色和权限后，您现在可以创建代理并将其与该角色关联。这可让代理安全地从 Amazon Secrets Manager 中检索数据库凭证，并为您的应用程序启用 IAM 身份验证。有关说明，请参阅 the section called “创建代理”。

为 Amazon RDS 创建代理

您可以将代理与 RDS for MariaDB、RDS for Microsoft SQL Server、RDS for MySQL 或 RDS for PostgreSQL 数据库实例相关联。

控制台

创建代理

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Proxies (代理)。
3. 选择 Create proxy (创建代理)。
4. 为代理配置以下设置。

设置	描述
引擎系列	代理在解释进出数据库的网络流量时识别的数据库网络协议。 **Note** 要使用 RDS for PostgreSQL，请确保将 `postgres` 数据库保留在实例中。请参阅对已删除的 `postgres` 数据库进行故障排除。
代理标识符	在您的 Amazon 账户 ID 和当前 Amazon 区域内具有唯一性的名称。
空闲客户端连接超时	如果客户端连接在设定的时间段内保持空闲，则代理会关闭客户端连接。默认情况下，这是 1800 秒（30 分钟）。当应用程序在完成上一个请求后的指定时间内未提交新请求时，连接处于空闲状态。代理将底层数据库连接保持为打开状态，并将其返回到连接池，使其可用于新的客户端连接。 要主动移除过时的连接，请减少空闲客户端连接超时。要最大限度地降低工作负载高峰期间的连接成本，请增加此超时。

设置	描述
数据库	要通过此代理访问的 RDS 数据库实例。该列表仅包含具有兼容数据库引擎、引擎版本和其他设置的数据库实例和集群。如果列表为空，请创建与 RDS Proxy 兼容的新数据库实例或集群。为此，请按照创建 Amazon RDS 数据库实例中的过程操作 然后，尝试再次创建代理。
连接池最大连接数	一个介于 1 到 100 之间的值，用于定义 RDS 代理可以使用的 `max_connections` 限制的百分比。如果您只打算对此数据库实例或集群使用一个代理，请将此值设置为 100。有关 RDS 代理如何使用此设置的更多信息，请参阅 the section called “MaxConnectionsPercent”。
会话绑定筛选条件	防止 RDS 代理绑定某些检测到的会话状态来绕过多路复用连接的默认安全措施。目前，PostgreSQL 不支持此设置，唯一可用的选项是 `EXCLUDE_VARIABLE_SETS` 。启用它可能会导致来自一个连接的会话变量影响其它连接，如果查询依赖于在当前事务之外设置的会话变量，则会导致错误或正确性问题。只有在确认您的应用程序可以安全地共享数据库连接后，才能使用此选项。 以下模式被认为是安全的： • `SET` 语句，其中没有更改有效会话变量值。换句话说，会话变量没有变化。 • 您可以更改会话变量值并在同一个事务中执行一条语句。 有关更多信息，请参阅 避免固定 RDS 代理。
连接借用超时	如果您希望代理使用所有可用的数据库连接，请设置在代理返回超时错误之前的等待时间。最多可以指定五分钟。仅当代理已达到最大连接数并且所有连接都在使用中时，此设置才适用。

设置	描述
初始化查询	（可选）添加初始化查询或修改当前查询。您可以为代理指定一个或多个 SQL 语句，以便在打开每个新数据库连接时运行。此设置通常与 SET 语句一起使用，以确保每个连接具有完全相同的设置。确保您添加的查询有效。要在单个 SET 语句中包含多个变量，请使用逗号分隔符。例如： `SET variable1 =value1, variable2 =value2` 对于多个语句，请使用分号作为分隔符。 **Important** 由于您可以将初始化查询作为目标组配置的一部分进行访问，因此初始化查询不受身份验证或加密方法的保护。有权查看或管理代理目标组配置的任何人都可以查看初始化查询。您不应将敏感数据（例如密码或长期保存的加密密钥）添加至此选项。
Amazon Identity and Access Management（IAM）角色	一个有权访问 Secrets Manager 密钥的 IAM 角色，这些密钥代表代理可以使用的数据库用户账户的凭证。或者，您可以从 Amazon Web Services Management Console创建新的 IAM 角色。
Secrets Manager 密钥	至少选择一个 Secrets Manager 密钥，该密钥包含允许代理访问 RDS 数据库实例的数据库用户凭证。
客户端身份验证类型	代理对来自客户端的连接使用的身份验证类型。您的选择适用于与此代理关联的所有 Secrets Manager 密钥。如果您需要为每个密钥指定不同的客户端身份验证类型，请使用 Amazon CLI 或改用 API 创建代理。

设置	描述
IAM 身份验证	是要求、支持、还是禁止对代理连接进行 IAM 身份验证。允许选项仅对 RDS for SQL Server 的代理有效。您的选择适用于与此代理关联的所有 Secrets Manager 密钥。如果您需要为每个密钥指定不同的 IAM 身份验证，请使用 Amazon CLI 或改用 API 创建代理。
需要传输层安全性	对所有客户端连接强制执行 TLS/SSL。代理对其与底层数据库的连接使用相同的加密设置，无论客户端连接是已加密还是未加密。
子网	此字段预填充与您的 VPC 关联的所有子网。您可以移除代理不需要的任何子网，但必须至少保留两个子网。

设置	描述
VPC 安全组	选择现有 VPC 安全组或者从 Amazon Web Services Management Console创建一个新的安全组。配置入站规则以支持应用程序访问代理，并配置出站规则以支持来自数据库目标的流量。 Note 此安全组必须支持从代理到数据库的连接。它既用于从应用程序到代理的入口，也用于从代理到数据库的出口。例如，如果您对数据库和代理使用相同的安全组，则确保该安全组中的资源可以相互通信。 使用共享 VPC 时，避免使用 VPC 的默认安全组或与另一个账户关联的安全组。而是选择属于您的账户的安全组。如果不存在安全组，则创建一个安全组。有关更多信息，请参阅使用共享 VPC。 RDS 跨多个可用区部署代理以确保高可用性。要启用跨可用区通信，代理子网的网络访问控制列表（ACL）必须支持引擎端口上的出口和所有端口上的入口。有关网络 ACL 的更多信息，请参阅使用网络 ACL 控制指向子网的流量。如果代理和目标的网络 ACL 相同，则必须添加 TCP 协议入口规则，其中源设置为 VPC CIDR。您还必须添加特定于引擎端口的 TCP 协议出口规则，其中目标设置为 VPC CIDR。
激活增强的日志记录	启用该设置可解决代理兼容性或性能问题。启用后，RDS 代理会记录详细的性能信息，有助于您调试 SQL 行为或代理连接性能和可扩展性。 仅启用此设置来进行调试，并确保已采取适当的安全措施来保护日志中的敏感信息。为了最大限度地减少开销，RDS 代理会在激活后 24 小时自动禁用此设置。可以临时使用它来解决特定问题。

5. 选择 Create proxy (创建代理)。

Amazon CLI

要使用 Amazon CLI 创建代理，请使用以下必需的参数调用 create-db-proxy 命令：

- `--db-proxy-name`
- `--engine-family`
- `--role-arn`
- `--auth`
- `--vpc-subnet-ids`

`--engine-family` 值区分大小写。

Example

对于 Linux、macOS 或 Unix：

```
aws rds create-db-proxy \
    --db-proxy-name proxy_name \
    --engine-family { MYSQL | POSTGRESQL | SQLSERVER } \
    --auth ProxyAuthenticationConfig_JSON_string \
    --role-arn iam_role \
    --vpc-subnet-ids space_separated_list \
    [--vpc-security-group-ids space_separated_list] \
    [--require-tls | --no-require-tls] \
    [--idle-client-timeout value] \
    [--debug-logging | --no-debug-logging] \
    [--tags comma_separated_list]
```

对于 Windows：

```
aws rds create-db-proxy ^
    --db-proxy-name proxy_name ^
    --engine-family { MYSQL | POSTGRESQL | SQLSERVER } ^
    --auth ProxyAuthenticationConfig_JSON_string ^
    --role-arn iam_role ^
    --vpc-subnet-ids space_separated_list ^
    [--vpc-security-group-ids space_separated_list] ^
    [--require-tls | --no-require-tls] ^
    [--idle-client-timeout value] ^
    [--debug-logging | --no-debug-logging] ^
```

```
    [--tags comma_separated_list]
```

以下是 --auth 选项的 JSON 值的示例。此示例对每个密钥应用不同的客户端身份验证类型。

```
[
  {
    "Description": "proxy description 1",
    "AuthScheme": "SECRETS",
    "SecretArn": "arn:aws:secretsmanager:us-
west-2:123456789123:secret/1234abcd-12ab-34cd-56ef-1234567890ab",
    "IAMAuth": "DISABLED",
    "ClientPasswordAuthType": "POSTGRES_SCRAM_SHA_256"
  },

  {
    "Description": "proxy description 2",
    "AuthScheme": "SECRETS",
    "SecretArn": "arn:aws:secretsmanager:us-
west-2:111122223333:secret/1234abcd-12ab-34cd-56ef-1234567890cd",
    "IAMAuth": "DISABLED",
    "ClientPasswordAuthType": "POSTGRES_MD5"

  },

  {
    "Description": "proxy description 3",
    "AuthScheme": "SECRETS",
    "SecretArn": "arn:aws:secretsmanager:us-
west-2:111122221111:secret/1234abcd-12ab-34cd-56ef-1234567890ef",
    "IAMAuth": "REQUIRED"
  }

]
```

> **Tip**
>
> 如果您尚不知道要用于 --vpc-subnet-ids 参数的子网 ID，请参阅 设置 RDS 代理的网络先决条件 获取有关如何查找它们的示例。

> **Note**
>
> 安全组必须允许访问代理连接到的数据库。同一安全组用于从应用程序到代理的入口以及从代理到数据库的出口。例如，假设您对数据库和代理使用同一安全组。在这种情况下，请确保您指定该安全组中的资源可以与同一安全组中的其他资源进行通信。
>
> 使用共享 VPC 时，您不能使用 VPC 的默认安全组，也不能使用属于其他账户的安全组。选择属于您账户的安全组。如果不存在，请创建一个。有关此限制的详细信息，请参阅使用共享 VPC。

要为代理创建正确的关联，您还可以使用 register-db-proxy-targets 命令。指定目标组名称 `default`。创建每个代理时，RDS Proxy 会自动创建一个此名称的目标组。

```
aws rds register-db-proxy-targets
    --db-proxy-name value
    [--target-group-name target_group_name]
    [--db-instance-identifiers space_separated_list]  # rds db instances, or
    [--db-cluster-identifiers cluster_id]        # rds db cluster (all instances)
```

RDS API

要创建 RDS 代理，请调用 Amazon RDS API 操作 CreateDBProxy。传递具有 AuthConfig 数据结构的参数。

创建每个代理时，RDS Proxy 会自动创建一个名为 `default` 的目标组。通过调用函数 RegisterDBProxyTargets，将 RDS 数据库实例与目标组关联。

查看代理

创建一个或多个 RDS 代理后，您可以在 Amazon Web Services Management Console、Amazon CLI 或 RDS API 中查看和管理它们。您可以查看其配置详细信息，监控性能，并根据需要确定要修改或删除哪些代理。

要使数据库应用程序能够通过代理路由流量，必须在连接字符串中指定代理端点。

控制台

在控制台中查看代理

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。

2. 在导航窗格中，选择 Proxies (代理)。
3. 选择代理名称以查看其详细信息。
4. 在详细信息页面上，目标组部分显示了代理与特定 RDS 数据库实例关联的方式。您可以导航到默认目标组页面，以便更深入地查看此关联，包括在创建代理期间定义的配置设置。这些设置包括最大连接百分比、连接借用超时、引擎系列和会话绑定筛选条件。

CLI

要使用 CLI 查看您的代理，请使用 describe-db-proxies 命令。默认情况下，该请求会返回由您的 Amazon 账户拥有的所有代理。要查看单个代理的详细信息，请通过 `--db-proxy-name` 参数指定其名称。

```
aws rds describe-db-proxies [--db-proxy-name proxy_name]
```

要查看与代理关联的其它信息，请使用以下命令。

```
aws rds describe-db-proxy-target-groups  --db-proxy-name proxy_name

aws rds describe-db-proxy-targets --db-proxy-name proxy_name
```

使用以下命令序列查看有关与代理关联的内容的更多详细信息：

1. 要获取代理列表，请运行 describe-db-proxies。
2. 要显示连接参数（如代理可以使用的最大连接百分比），请运行 describe-db-proxy-target-groups `--db-proxy-name`。使用代理的名称作为参数值。
3. 要查看与返回的目标组关联的 RDS 数据库实例的详细信息，请运行 describe-db-proxy-targets。

RDS API

要使用 RDS API 查看您的代理，请使用 DescribeDBProxies 操作。该操作将返回 DBProxy 数据类型的值。

要查看代理连接设置的详细信息，请将此返回值中的代理标识符与 DescribeDBProxyTargetGroups 操作一起使用。该操作将返回 DBProxyTargetGroup 数据类型的值。

要查看与代理关联的 RDS 实例或 Aurora 数据库集群，请使用 DescribeDBProxyTargets 操作。该操作将返回 DBProxyTarget 数据类型的值。

通过 RDS Proxy 连接到数据库

通过代理或通过连接到数据库来连接到 RDS 数据库实例的方式通常是相同的。有关更多信息，请参阅代理终端节点概述。

主题

使用本机身份验证连接到代理

执行以下步骤，使用本机身份验证连接到代理：

1. 查找代理端点。在 Amazon Web Services Management Console 中，您可以在相应代理的详细信息页面上找到端点。通过 Amazon CLI，您可以使用 describe-db-proxies 命令。下面的示例演示如何操作。

```
# Add --output text to get output as a simple tab-separated list.
$ aws rds describe-db-proxies --query '*[*].
{DBProxyName:DBProxyName,Endpoint:Endpoint}'
[
    [
        {
            "Endpoint": "the-proxy.proxy-demo.us-east-1.rds.amazonaws.com",
            "DBProxyName": "the-proxy"
        },
        {
            "Endpoint": "the-proxy-other-secret.proxy-demo.us-
east-1.rds.amazonaws.com",
            "DBProxyName": "the-proxy-other-secret"
        },
        {
            "Endpoint": "the-proxy-rds-secret.proxy-demo.us-
east-1.rds.amazonaws.com",
            "DBProxyName": "the-proxy-rds-secret"
        },
        {
            "Endpoint": "the-proxy-t3.proxy-demo.us-east-1.rds.amazonaws.com",
```

```
                "DBProxyName": "the-proxy-t3"
            }
        ]
    ]
```

2. 在客户端应用程序的连接字符串中指定该端点作为主机参数。例如，指定代理端点作为 `mysql -h` 选项或 `psql -h` 选项的值。
3. 提供您通常使用的相同数据库用户名和密码。

使用 IAM 身份验证连接到代理

在将 IAM 身份验证与RDS Proxy结合使用时，请将数据库用户设置为使用常规用户名和密码进行身份验证。IAM 身份验证适用于从 Secrets Manager 中检索用户名和密码凭证的RDS Proxy。从RDS 代理至底层数据库的连接不经由 IAM。

要使用 IAM 身份验证连接到 RDS 代理，请使用与对 RDS 数据库实例使用 IAM 身份验证相同的常规连接过程。有关使用 IAM 的一般信息，请参阅Amazon RDS 中的安全性。

RDS Proxy 在 IAM 用法方面的主要区别包括：

- 您不会使用授权插件配置每个单独的数据库用户。数据库用户在数据库中仍有常规的用户名和密码。您可以设置包含这些用户名和密码的 Secrets Manager 密钥，并授权 RDS Proxy 从 Secrets Manager 中检索凭证。

  IAM 身份验证应用于您的客户端程序与代理之间的连接。然后，代理使用从 Secrets Manager 中检索的用户名和密码凭证对数据库进行身份验证。
- 指定代理端点，而不是实例、集群或读取器端点。有关代理端点的详细信息，请参阅 使用 IAM 身份验证连接到数据库实例。
- 在直接数据库 IAM 身份验证情况下，您可以有选择地选取数据库用户并将其配置为使用特殊身份验证插件进行标识。然后，您便可以使用 IAM 身份验证连接到这些用户。

  在代理使用案例中，您需要为代理提供包含某些用户的用户名和密码（本机身份验证）的密钥。然后，您便可以使用 IAM 身份验证连接到代理。在这里，您可以通过使用代理端点而非数据库端点生成身份验证令牌来实现此目的。您还可以使用与您所提供密钥的用户名之一匹配的用户名。
- 当使用 IAM 身份验证连接到代理时，确保您使用传输层安全性 (TLS)/安全套接字层 (SSL)。

您可以通过修改 IAM 策略授予特定用户对代理的访问权限。下面是一个示例。

```
"Resource": "arn:aws:rds-db:us-east-2:1234567890:dbuser:prx-ABCDEFGHIJKL01234/db_user"
```

使用 Microsoft SQL Server 连接到代理的注意事项

要使用 IAM 身份验证连接到代理，请勿使用密码字段。相反，您可以在令牌字段中为每种类型的数据库驱动程序提供相应的令牌属性。例如，对 JDBC 使用 `accessToken` 属性，或对 ODBC 使用 `sql_copt_ss_access_token` 属性。或者对于 .NET SqlClient 驱动程序使用 `AccessToken` 属性。您不能对不支持令牌属性的客户端使用 IAM 身份验证。

在某些情况下，代理无法共享数据库连接，而是将从客户端应用程序到代理的连接固定到专用的数据库连接。有关这些条件的更多信息，请参阅 避免固定 RDS 代理。

使用 PostgreSQL 连接到代理的注意事项

如果您创建新的 PostgreSQL 数据库用户来连接到 RDS 代理，请确保授予该用户对数据库的 `CONNECT` 权限。否则，用户将无法建立连接。有关更多信息，请参阅 the section called "使用 RDS 代理时向 PostgreSQL 数据库添加新数据库用户"。

当客户端启动到 PostgreSQL 数据库的连接时，它会发送一条启动消息。此消息包含参数名称/值字符串对。有关详细信息，请参阅 PostgreSQL 文档中的 PostgreSQL 消息格式中的 `StartupMessage`。

当您通过 RDS 代理进行连接时，启动消息可以包含以下当前识别的参数：

- `user`
- `database`

启动消息还可以包含以下其他运行时参数：

- `application_name`
- `client_encoding`
- `DateStyle`
- `TimeZone`
- `extra_float_digits`
- `search_path`

有关 PostgreSQL 消息收发的更多信息，请参阅 PostgreSQL 文档中的前端/后端协议。

对于 PostgreSQL，如果您使用 JDBC，我们建议您执行以下操作以避免固定：

- 将 JDBC 连接参数 `assumeMinServerVersion` 至少设置为 `9.0` 以避免固定。这可阻止 JDBC 驱动程序在运行 `SET extra_float_digits = 3` 时，在连接启动期间执行额外的往返行程。
- 将 JDBC 连接参数 `ApplicationName` 设置为 *`any/your-application-name`* 以避免固定。这样做会阻止 JDBC 驱动程序在运行 `SET application_name = "PostgreSQL JDBC Driver"` 时，在连接启动期间执行额外的往返行程。请注意，JDBC 参数为 `ApplicationName`，但 PostgreSQL `StartupMessage` 参数为 `application_name`。

有关更多信息，请参阅 避免固定 RDS 代理。有关使用 JDBC 进行连接的更多信息，请参阅 PostgreSQL 文档中的连接到数据库。

管理 RDS 代理

本节提供有关如何管理 RDS 代理操作和配置的信息。这些过程可帮助您的应用程序最有效地利用数据库连接，并实现最大程度的连接重用。您越多地利用连接重用，就可以节省越多的 CPU 和内存开销。这进而减少了应用程序的延迟，使数据库能够将更多资源用于处理应用程序请求。

主题

- 修改 RDS 代理
- 使用 RDS 代理时添加新数据库用户
- RDS 代理连接注意事项
- 避免固定 RDS 代理
- 删除 RDS 代理

修改 RDS 代理

您可以在创建代理后更改与代理关联的特定设置。可通过修改代理本身和/或其关联的目标组来执行此操作。每个代理都有一个关联的目标组。

Amazon Web Services Management Console

> **Important**
>
> Client authentication type（客户端身份验证类型）和 IAM authentication（IAM 身份验证）字段中的值适用于与此代理关联的所有 Secrets Manager 密钥。要为每个密钥指定不同的值，请使用 Amazon CLI 或改用 API 来修改代理。

修改代理的设置

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Proxies (代理)。
3. 在代理列表中，选择要修改其设置的代理或转到其详细信息页面。
4. 对于 Actions (操作)，选择 Modify (修改)。
5. 输入或选择要修改的属性。您可以指定如下内容：
   - 代理标识符 – 通过输入新标识符来重命名代理。
   - 空闲客户端连接超时 - 输入空闲客户端连接超时的时间段。
   - IAM 角色 – 更改用于从 Secrets Manager 中检索密钥的 IAM 角色。
   - Secrets Manager 密钥 - 添加或删除 SSecrets Manager 密钥。这些密钥对应于数据库用户名和密码。
   - Client authentication type（客户端身份验证类型）-（仅限 PostgreSQL）更改客户端与代理的连接的身份验证类型。
   - IAM authentication（IAM 身份验证）– 要求或禁止对与代理的连接进行 IAM 身份验证。
   - 需要传输层安全性 - 打开或关闭传输层安全性 (TLS) 的要求。
   - VPC 安全组 - 添加或删除供代理使用的 VPC 安全组。
   - 启用增强型日志记录 - 启用或禁用增强型日志记录。
6. 选择修改。

如果您找不到要更改的列出设置，请使用以下过程更新代理的目标组。与代理关联的目标组控制与物理数据库连接相关的设置。每个代理都有一个名为 `default` 的关联目标组，该目标组与代理一起自动创建。您无法重命名默认目标组。

您只能从代理详细信息页面修改目标组，而不能从 Proxies (代理) 页面上的列表中进行修改。

修改代理目标组的设置

1. 在 Proxies (代理) 页面上，转到代理的详细信息页面。
2. 对于 Target groups (目标组)，选择 `default` 链接。目前，所有代理都有一个名为 `default` 的目标组。
3. 在默认目标组的详细信息页面上，选择 Modify (修改)。
4. 为您可以修改的属性选择新设置：

- 数据库 – 选择不同的 RDS 数据库实例或集群。
- 连接池最大连接数 – 调整代理可使用的最大可用连接的百分比。
- 会话固定筛选条件 -（可选）选择会话固定筛选条件。这绕过了跨客户端多路复用数据库连接的默认安全措施。目前，PostgreSQL 不支持该设置。唯一的选择是 `EXCLUDE_VARIABLE_SETS`。

  启用此设置可能会导致一个连接的会话变量影响其它连接。如果您的查询依赖于在当前事务之外设置的会话变量值，则可能会导致错误或正确性问题。在确认您的应用程序可以安全地跨客户端连接共享数据库连接之后，请考虑使用此选项。

  可以认为以下模式是安全的：

  - SET 语句，其中有效会话变量值没有更改，即没有更改会话变量。
  - 您可以更改会话变量值并在同一个事务中执行一条语句。

  有关更多信息，请参阅 [避免固定 RDS 代理]。
- 连接借用超时 - 调整连接借用超时间隔。当最大连接数已用于代理时，此设置适用。该设置确定在返回超时错误之前代理等待连接可用的时间。
- 初始化查询。（可选）添加初始化查询或修改当前查询。您可以为代理指定一个或多个 SQL 语句，以便在打开每个新数据库连接时运行。此设置通常与 SET 语句一起使用，以确保每个连接具有完全相同的设置。确保您添加的查询有效。要在单个 SET 语句中包含多个变量，请使用逗号分隔符。例如：

  ```
  SET variable1=value1, variable2=value2
  ```

  对于多个语句，请使用分号作为分隔符。

  不能更改某些属性，例如，目标组标识符和数据库引擎。

5. 选择 Modify target group (修改目标组)。

Amazon CLI

要使用 Amazon CLI 修改代理，请使用 modify-db-proxy、modify-db-proxy-target-group、deregister-db-proxy-targets 和 register-db-proxy-targets 命令。

使用 `modify-db-proxy` 命令，您可以更改以下属性：

- 代理使用的一组 Secrets Manager 密钥。
- 是否需要 TLS。
- 空闲客户端超时。
- 是否记录 SQL 语句中的其他信息以进行调试。
- 用于检索 Secrets Manager 密钥的 IAM 角色。
- 代理使用的安全组。

以下示例演示了如何重命名现有代理。

```
aws rds modify-db-proxy --db-proxy-name the-proxy --new-db-proxy-name the_new_name
```

要修改与连接相关的设置或重命名目标组，请使用 `modify-db-proxy-target-group` 命令。目前，所有代理都有一个名为 `default` 的目标组。在您使用此目标组时，请指定代理的名称，并指定 `default` 作为目标组的名称。您无法重命名默认目标组。

以下示例演示了如何首先检查代理的 `MaxIdleConnectionsPercent` 设置，然后使用目标组对其进行更改。

```
aws rds describe-db-proxy-target-groups --db-proxy-name the-proxy

{
    "TargetGroups": [
        {
            "Status": "available",
            "UpdatedDate": "2019-11-30T16:49:30.342Z",
            "ConnectionPoolConfig": {
                "MaxIdleConnectionsPercent": 50,
                "ConnectionBorrowTimeout": 120,
                "MaxConnectionsPercent": 100,
                "SessionPinningFilters": []
            },
            "TargetGroupName": "default",
            "CreatedDate": "2019-11-30T16:49:27.940Z",
            "DBProxyName": "the-proxy",
            "IsDefault": true
        }
    ]
}
```

```
aws rds modify-db-proxy-target-group --db-proxy-name the-proxy --target-group-name
 default --connection-pool-config '
{ "MaxIdleConnectionsPercent": 75 }'

{
    "DBProxyTargetGroup": {
        "Status": "available",
        "UpdatedDate": "2019-12-02T04:09:50.420Z",
        "ConnectionPoolConfig": {
            "MaxIdleConnectionsPercent": 75,
            "ConnectionBorrowTimeout": 120,
            "MaxConnectionsPercent": 100,
            "SessionPinningFilters": []
        },
        "TargetGroupName": "default",
        "CreatedDate": "2019-11-30T16:49:27.940Z",
        "DBProxyName": "the-proxy",
        "IsDefault": true
    }
}
```

使用 `deregister-db-proxy-targets` 和 `register-db-proxy-targets` 命令，您可以通过相应的目标组来更改代理关联的 RDS 数据库实例。目前，每个代理都可以连接到一个 RDS 数据库实例。目标组可跟踪多可用区配置中所有 RDS 数据库实例的连接详细信息。

以下示例首先介绍了与名为 `cluster-56-2020-02-25-1399` 的 Aurora MySQL 集群关联的代理。该示例演示了如何更改代理，以便连接到名为 `provisioned-cluster` 的不同集群。

使用 RDS 数据库实例时，请指定 `--db-instance-identifier` 选项。

以下示例修改 Aurora MySQL 代理。Aurora PostgreSQL 代理具有端口 5432。

```
aws rds describe-db-proxy-targets --db-proxy-name the-proxy

{
    "Targets": [
        {
            "Endpoint": "instance-9814.demo.us-east-1.rds.amazonaws.com",
            "Type": "RDS_INSTANCE",
            "Port": 3306,
            "RdsResourceId": "instance-9814"
        },
        {
```

```
            "Endpoint": "instance-8898.demo.us-east-1.rds.amazonaws.com",
            "Type": "RDS_INSTANCE",
            "Port": 3306,
            "RdsResourceId": "instance-8898"
        },
        {
            "Endpoint": "instance-1018.demo.us-east-1.rds.amazonaws.com",
            "Type": "RDS_INSTANCE",
            "Port": 3306,
            "RdsResourceId": "instance-1018"
        },
        {
            "Type": "TRACKED_CLUSTER",
            "Port": 0,
            "RdsResourceId": "cluster-56-2020-02-25-1399"
        },
        {
            "Endpoint": "instance-4330.demo.us-east-1.rds.amazonaws.com",
            "Type": "RDS_INSTANCE",
            "Port": 3306,
            "RdsResourceId": "instance-4330"
        }
    ]
}

aws rds deregister-db-proxy-targets --db-proxy-name the-proxy --db-cluster-identifier
 cluster-56-2020-02-25-1399

aws rds describe-db-proxy-targets --db-proxy-name the-proxy

{
    "Targets": []
}

aws rds register-db-proxy-targets --db-proxy-name the-proxy --db-cluster-identifier
 provisioned-cluster

{
    "DBProxyTargets": [
        {
            "Type": "TRACKED_CLUSTER",
            "Port": 0,
            "RdsResourceId": "provisioned-cluster"
        },
```

```
        {
            "Endpoint": "gkldje.demo.us-east-1.rds.amazonaws.com",
            "Type": "RDS_INSTANCE",
            "Port": 3306,
            "RdsResourceId": "gkldje"
        },
        {
            "Endpoint": "provisioned-1.demo.us-east-1.rds.amazonaws.com",
            "Type": "RDS_INSTANCE",
            "Port": 3306,
            "RdsResourceId": "provisioned-1"
        }
    ]
}
```

RDS API

要使用 RDS API 修改代理，请使用 ModifyDBProxy、ModifyDBProxyTargetGroup、DeregisterDBProxyTargets 和 RegisterDBProxyTargets 操作。

使用 `ModifyDBProxy`，您可以更改以下属性：

- 代理使用的一组 Secrets Manager 密钥。
- 是否需要 TLS。
- 空闲客户端超时。
- 是否记录 SQL 语句中的其他信息以进行调试。
- 用于检索 Secrets Manager 密钥的 IAM 角色。
- 代理使用的安全组。

使用 `ModifyDBProxyTargetGroup`，您可以修改与连接相关的设置。目前，所有代理都有一个名为 `default` 的目标组。在您使用此目标组时，请指定代理的名称，并指定 `default` 作为目标组的名称。您无法重命名默认目标组。

使用 `DeregisterDBProxyTargets` 和 `RegisterDBProxyTargets`，您可以通过相应的目标组来更改代理关联的 RDS 数据库实例。目前，每个代理都可以连接到一个 RDS 数据库实例。目标组可跟踪多可用区配置中的所有 RDS 数据库实例的连接详细信息。

使用 RDS 代理时添加新数据库用户

在某些情况下，您可能会将新数据库用户添加到与代理关联的 RDS 数据库实例或集群。如果是这样，请添加 Secrets Manager 密钥或调整其用途，以存储该用户的凭证。为此，请运行以下步骤：

1. 使用 在 Amazon Secrets Manager 中设置 RDS 代理的数据库凭证 中描述的过程创建新 Secrets Manager 密钥。
2. 更新 IAM 角色以授予 RDS Proxy 对新 Secrets Manager 密钥的访问权限。为此，请更新 IAM 角色策略的资源部分。
3. 修改 RDS 代理，以在 Secrets Manager 密钥下添加新的 Secrets Manager 密钥。
4. 如果新用户取代了现有用户，请更新存储在代理的 Secrets Manager 密钥中的现有用户的凭证。

使用 RDS 代理时向 PostgreSQL 数据库添加新数据库用户

将新用户添加到 PostgreSQL 数据库时，如果有，请运行以下命令：

```
REVOKE CONNECT ON DATABASE postgres FROM PUBLIC;
```

向 `rdsproxyadmin` 用户授予 `CONNECT` 权限，以便用户可以监控目标数据库上的连接。

```
GRANT CONNECT ON DATABASE postgres TO rdsproxyadmin;
```

还可以通过将上述命令中的 `rdsproxyadmin` 更改为其他目标数据库用户，来允许该数据库用户执行运行状况检查。

使用 RDS 代理时更改数据库用户的密码

在某些情况下，您可能会更改与代理关联的 RDS 数据库实例中的数据库用户的密码。如果是这样，请使用新密码更新相应的 Secrets Manager 密钥。

RDS 代理连接注意事项

配置连接设置

若要调整 RDS Proxy 的连接池，您可以修改以下设置：

- IdleClientTimeout

- MaxConnectionsPercent
- MaxIdleConnectionsPercent
- ConnectionBorrowTimeout

IdleClientTimeout

您可以指定在代理关闭客户端连接之前，客户端连接可以处于空闲状态的时长。默认值为 1800 秒（30 分钟）。

如果应用程序未在上一请求完成后的指定时间内提交新请求，则将客户端连接视为空闲。底层数据库连接保持打开状态并返回到连接池。因此，它可以重复用于新的客户端连接。如果希望代理主动删除过时的连接，则降低空闲客户端连接超时。如果工作负载与代理建立频繁的连接，则增加空闲客户端连接超时，以便节省建立连接的成本。

此设置由 RDS 控制台中的 Idle client connection timeout（空闲客户端连接超时）字段以及 Amazon CLI 和 API 中的 `IdleClientTimeout` 设置表示。要了解如何在 RDS 控制台中更改 Idle client connection timeout（空闲客户端连接超时）字段的值，请参阅 Amazon Web Services Management Console。要了解如何更改 `IdleClientTimeout` 设置的值，请参阅 CLI 命令 modify-db-proxy 或 API 操作 ModifyDBProxy。

MaxConnectionsPercent

您可以限制 RDS 代理可与目标数据库建立的连接数。请将限制指定为数据库可用的最大连接数的百分比。此设置由 RDS 控制台中的 Connection pool maximum connections（连接池最大连接数）字段以及 Amazon CLI 和 API 中的 `MaxConnectionsPercent` 设置表示。

对于目标组使用的 RDS 数据库实例，`MaxConnectionsPercent` 值以 `max_connections` 设置的百分比表示。代理不会提前创建所有连接。此设置允许代理在工作负载需要时建立这些连接。

例如，对于 `max_connections` 设置为 1000 且 `MaxConnectionsPercent` 设置为 95 的注册数据库目标，RDS 代理将 950 个连接设置为与该数据库目标的并行连接的上限。

工作负载达到允许的最大数据库连接数的一个常见副作用是总体查询延迟增加，同时 `DatabaseConnectionsBorrowLatency` 指标也会增加。您可以通过比较 `DatabaseConnections` 和 `MaxDatabaseConnectionsAllowed` 指标来监控当前使用的数据库连接数和允许的数据库连接总数。

设置此参数时，请注意以下最佳实践：

- 为工作负载模式的变化留出足够的连接余量。建议将该参数设置为比最近监控的最大使用量高出至少 30%。由于 RDS 代理在多个节点之间重新分配数据库连接限额，因此，内部容量更改可能需要至少 30% 的余量来增加连接，以避免增加借用延迟。
- RDS 代理保留一定数量的连接用于主动监控，以支持快速失效转移、流量路由和内部操作。`MaxDatabaseConnectionsAllowed` 指标不包括这些预留连接。该指标表示可用于为工作负载提供服务的连接数，可以低于从 `MaxConnectionsPercent` 设置得出的值。

  推荐的最低 `MaxConnectionsPercent` 值

  - db.t3.small：30
  - db.t3.medium 或更大：20

要了解如何在 RDS 控制台中更改 Connection pool maximum connections（连接池最大连接数）字段的值，请参阅 Amazon Web Services Management Console。要了解如何更改 `MaxConnectionsPercent` 设置的值，请参阅 CLI 命令 modify-db-proxy-target-group 或 API 操作 ModifyDBProxyTargetGroup。

有关数据库连接限制的更多信息，请参阅数据库最大连接数。

MaxIdleConnectionsPercent

您可以控制 RDS Proxy 可在连接池中保留的空闲数据库连接数。默认情况下，如果连接五分钟内没有活动，RDS 代理会将其池中的数据库连接视为空闲。

对于 RDS 数据库实例目标组，`MaxIdleConnectionsPercent` 值以 `max_connections` 设置的百分比表示。原定设置值为 `MaxConnectionsPercent` 的 50%，上限为 `MaxConnectionsPercent` 的值。例如，如果 `MaxConnectionsPercent` 为 80，则 `MaxIdleConnectionsPercent` 的默认值为 40。如果未指定 `MaxConnectionsPercent` 的值，则对于 RDS for SQL Server，`MaxIdleConnectionsPercent` 为 5；对于所有其他引擎，默认值为 50。

对于高值，代理会将较高百分比的空闲数据库连接保持在打开状态。对于低值，代理会关闭较高百分比的空闲数据库连接。如果您的工作负载不可预测，请考虑为 `MaxIdleConnectionsPercent` 设置较高的值。这样做意味着 RDS 代理可以在不打开大量新数据库连接的情况下适应活动的突增。

此设置由 Amazon CLI 和 API 中 `DBProxyTargetGroup` 的 `MaxIdleConnectionsPercent` 设置表示。要了解如何更改 `MaxIdleConnectionsPercent` 设置的值，请参阅 CLI 命令 modify-db-proxy-target-group 或 API 操作 ModifyDBProxyTargetGroup。

有关数据库连接限制的更多信息，请参阅数据库最大连接数。

ConnectionBorrowTimeout

您可以选择 RDS Proxy 在返回超时错误之前等待连接池中数据库连接可用的时间。默认值为 120 秒。此设置适用于连接数达到最大而导致连接池中没有可用连接的情况。当没有适当的数据库实例可用于处理请求时，例如当失效转移操作正在进行时，它也适用。通过使用此设置，您可以为应用程序设置最佳等待期，而无需更改应用程序代码中的查询超时。

此设置由 RDS Proxy 控制台中的 Connection borrow timeout（连接借用超时）字段、`ConnectionBorrowTimeout` 中 `DBProxyTargetGroup` 的 Amazon CLI 设置或 API 表示。要了解如何在 RDS 控制台中更改 Connection borrow timeout（连接借用超时）字段的值，请参阅 Amazon Web Services Management Console。要了解如何更改 `ConnectionBorrowTimeout` 设置的值，请参阅 CLI 命令 modify-db-proxy-target-group 或 API 操作 ModifyDBProxyTargetGroup。

客户端和数据库连接

从您的应用程序到 RDS 代理的连接称为客户端连接。从代理到数据库的连接则称为数据库连接。使用 RDS 代理时，客户端连接在代理处终止，而数据库连接则在 RDS 代理中进行管理。

应用程序端连接池可以减少在应用程序与 RDS 代理之间重复建立连接。

在实现应用程序端连接池之前，请考虑以下配置方面：

- 客户端连接最长期限：RDS 代理强制规定客户端连接的最长期限为 24 小时。此值不可配置。将池配置为最长连接期限小于 24 小时，以免客户端连接意外中断。
- 客户端连接空闲超时：RDS 代理强制规定客户端连接的最长空闲时间。为池配置的空闲连接超时值要小于 RDS 代理的客户端连接空闲超时设置，以免连接意外中断。

在应用程序端连接池中配置的最大客户端连接数不必限制为 RDS 代理的 max_connections 设置。

客户端连接池可延长客户端连接期限。如果您的连接遇到固定问题，则池化客户端连接可能会降低多路复用效率。已固定但在应用程序端连接池中处于空闲状态的客户端连接会继续保持数据库连接，并防止数据库连接被其它客户端连接重复使用。查看您的代理日志，以检查您的连接是否遇到固定问题。

> **Note**
> 当数据库连接不再使用时，RDS 代理会在 24 小时后的某个时间关闭数据库连接。无论最大空闲连接设置的值如何，代理都会执行此操作。

避免固定 RDS 代理

当数据库请求不依赖于先前请求的状态信息时，多路复用效率更高。在这种情况下，RDS 代理可以在每个事务结束时重用连接。此类状态信息的示例包括可通过 SET 或 SELECT 语句更改的大多数变量和配置参数。默认情况下，客户端连接上的 SQL 事务可以在底层数据库连接之间多路复用。

与代理的连接可以进入一种称为固定的状态。固定连接后，每个后续事务将使用相同的底层数据库连接，直到会话结束。在会话结束之前，其他客户端连接也不能重用该数据库连接。客户端连接断开时，会话结束。

当 RDS 代理检测到不适合其他会话的会话状态更改时，它会自动将客户端连接固定到特定的数据库连接。固定降低了连接重用的有效性。如果您的所有连接或几乎所有连接都遇到固定问题，请考虑修改应用程序代码或工作负载，以减少导致固定问题的情形。

例如，您的应用程序更改了会话变量或配置参数。在这种情况下，后面的语句可能依赖于新变量或参数来生效。因此，当 RDS 代理处理更改会话变量或配置设置的请求时，它会将该会话固定到数据库连接。这样，会话状态对于同一会话中的所有后续事务仍然有效。

对于某些数据库引擎，此规则并不适用于您可以设置的全部参数。RDS 代理会跟踪某些语句和变量。因此，在您修改这些内容时，RDS 代理不会固定会话。在这种情况下，RDS 代理仅将连接重用于具有相同设置值的其他会话。有关 RDS 代理针对数据库引擎跟踪的内容的详细信息，请参阅以下内容：

- RDS 代理针对 RDS for SQL Server 数据库跟踪的内容
- RDS 代理针对 RDS for MariaDB 数据库和 RDS for MySQL 数据库跟踪的内容

RDS 代理针对 RDS for SQL Server 数据库跟踪的内容

RDS 代理跟踪以下 SQL Server 语句：

- `USE`
- `SET ANSI_NULLS`
- `SET ANSI_PADDING`
- `SET ANSI_WARNINGS`
- `SET ARITHABORT`
- `SET CONCAT_NULL_YIELDS_NULL`
- `SET CURSOR_CLOSE_ON_COMMIT`

- `SET DATEFIRST`
- `SET DATEFORMAT`
- `SET LANGUAGE`
- `SET LOCK_TIMEOUT`
- `SET NUMERIC_ROUNDABORT`
- `SET QUOTED_IDENTIFIER`
- `SET TEXTSIZE`
- `SET TRANSACTION ISOLATION LEVEL`

RDS 代理针对 RDS for MariaDB 数据库和 RDS for MySQL 数据库跟踪的内容

RDS 代理跟踪以下 MariaDB 和 MySQL 语句：

- DROP DATABASE
- DROP SCHEMA
- USE

RDS 代理跟踪以下 MySQL 和 MariaDB 变量：

- `AUTOCOMMIT`
- `AUTO_INCREMENT_INCREMENT`
- `CHARACTER SET (or CHAR SET)`
- `CHARACTER_SET_CLIENT`
- `CHARACTER_SET_DATABASE`
- `CHARACTER_SET_FILESYSTEM`
- `CHARACTER_SET_CONNECTION`
- `CHARACTER_SET_RESULTS`
- `CHARACTER_SET_SERVER`
- `COLLATION_CONNECTION`
- `COLLATION_DATABASE`
- `COLLATION_SERVER`

- INTERACTIVE_TIMEOUT
- NAMES
- NET_WRITE_TIMEOUT
- QUERY_CACHE_TYPE
- SESSION_TRACK_SCHEMA
- SQL_MODE
- TIME_ZONE
- TRANSACTION_ISOLATION (or TX_ISOLATION)
- TRANSACTION_READ_ONLY (or TX_READ_ONLY)
- WAIT_TIMEOUT

> **Note**
>
> 当您在会话范围内设置 `TRANSACTION_ISOLATION` 和 `TRANSACTION_READ_ONLY` 变量时，RDS 代理会跟踪对这些变量的更改。但是，如果在下一个事务范围内设置它们，RDS 代理将固定连接。无论您使用 `SET` 语句还是 `SET TRANSACTION` 语句来配置这些值，此行为都适用。

显著减少固定

RDS 代理的性能优化包括尝试通过最小化固定来最大化事务级别连接重用（多路复用）。

您可以执行以下步骤以显著减少固定：

- 避免可能导致固定的不必要的数据库请求。
- 在所有连接中一致地设置变量和配置设置。这样，后续会话更有可能重用具有这些特定设置的连接。

  但是，对于 PostgreSQL 设置，变量会导致会话固定。
- 对于 MySQL 引擎系列数据库，可将会话固定筛选条件应用于代理。您可以免除某些类型的操作，使其不固定会话（如果您知道这样做不会影响应用程序的正确操作）。
- 通过监控 Amazon CloudWatch 指标 `DatabaseConnectionsCurrentlySessionPinned` 来查看固定的发生频率。有关该指标和其他 CloudWatch 指标的信息，请参阅 使用 Amazon CloudWatch 监控 RDS Proxy 指标。

- 如果您使用 SET 语句为每个客户端连接执行相同的初始化，则可以在保留事务级别多路复用的同时执行此操作。在这种情况下，您将设置初始会话状态的语句移动到由代理使用的初始化查询中。该属性是一个字符串，包含一个或多个 SQL 语句（用分号分隔）。

  例如，您可以为设置特定配置参数的代理定义初始化查询。然后，每当为该代理设置新连接时，RDS 代理都会应用这些设置。您可以从应用程序代码中删除相应的 SET 语句，这样，它们就不会干扰事务级别多路复用。

  有关代理的固定发生频率的指标，请参阅 使用 Amazon CloudWatch 监控 RDS Proxy 指标。

导致对所有引擎系列进行固定的条件

对于以下情况（其中多路复用可能会导致意外行为），代理将会话固定到当前连接：

- 文本大小大于 16 KB 的任何语句都会导致代理固定会话。

导致对 RDS for Microsoft SQL Server 进行固定的条件

对于 RDS for SQL Server，以下交互也会导致固定：

- 使用多个活动的结果集（MARS）。有关 MARS 的信息，请参阅 SQL Server 文档。
- 使用分布式事务协调器（DTC）通信。
- 创建临时表、事务、游标或预准备语句。
- 使用以下 SET 语句：
  - `SET ANSI_DEFAULTS`
  - `SET ANSI_NULL_DFLT`
  - `SET ARITHIGNORE`
  - `SET DEADLOCK_PRIORITY`
  - `SET FIPS_FLAGGER`
  - `SET FMTONLY`
  - `SET FORCEPLAN`
  - `SET IDENTITY_INSERT`
  - `SET NOCOUNT`
  - `SET NOEXEC`
  - `SET OFFSETS`

- SET PARSEONLY
- SET QUERY_GOVERNOR_COST_LIMIT
- SET REMOTE_PROC_TRANSACTIONS
- SET ROWCOUNT
- SET SHOWPLAN_ALL、SHOWPLAN_TEXT 和 SHOWPLAN_XML
- SET STATISTICS
- SET XACT_ABORT

导致对 RDS for MariaDB 和 RDS for MySQL 进行固定的条件

对于 MariaDB 和 MySQL，以下交互也会导致固定：

- 显式表锁定语句 LOCK TABLE、LOCK TABLES 或 FLUSH TABLES WITH READ LOCK 会导致代理固定会话。
- 通过使用 GET_LOCK 创建命名锁会导致代理固定会话。
- 设置用户变量或系统变量（有些例外）会将会话固定到代理。如果这种情况严重限制了连接的重用，您可以配置 SET 操作以避免固定。为此，请调整会话固定筛选条件属性。有关更多信息，请参阅为 Amazon RDS 创建代理 和修改 RDS 代理。
- 创建临时表会导致代理固定会话。这样，无论事务边界如何，临时表的内容都会在整个会话期间保留。
- 调用 ROW_COUNT 和 FOUND_ROWS 函数有时会导致固定问题。
- 预编译语句会导致代理固定会话。无论预编译语句使用 SQL 文本还是二进制协议，这项规则都适用。
- 使用 SET LOCAL 时，RDS 代理无法固定连接。
- 调用存储过程和存储函数不会导致固定。RDS 代理不会检测此类调用导致的任何会话状态更改。如果您希望跨事务持久保持存储例程中的会话状态，请确保您的应用程序不会更改该状态。例如，RDS 代理目前与创建跨所有事务持久存在的临时表的存储过程不兼容。

如果您拥有关于应用程序行为的专业知识，则可以跳过某些应用程序语句的固定行为。为此，请在创建代理时选择会话固定筛选条件选项。当前，您可以选择退出会话固定，以设置会话变量和配置设置。

导致对 RDS for PostgreSQL 进行固定的条件

对于 PostgreSQL，以下交互也会导致固定：

- 使用 SET 命令。
- 使用 PREPARE、DISCARD、DEALLOCATE、或 EXECUTE 命令管理准备好的语句。
- 创建临时序列、表或视图。
- 声明游标。
- 丢弃会话状态。
- 监听通知频道。
- 加载库模块，如 auto_explain。
- 使用函数操作序列，例如 nextval 和 setval。
- 使用函数与锁定交互，例如 pg_advisory_lock 和 pg_try_advisory_lock。

> **Note**
>
> RDS 代理未锁定事务级咨询锁，特别是 pg_advisory_xact_lock、pg_advisory_xact_lock_shared、pg_try_advisory_xact_l 和 pg_try_advisory_xact_lock_shared。

- 设置参数或将参数重置为其默认值。具体而言，就是使用 SET 和 set_config 命令为会话变量分配默认值。
- 调用存储过程和存储函数不会导致固定。RDS 代理不会检测此类调用导致的任何会话状态更改。如果您希望跨事务持久保持存储例程中的会话状态，请确保您的应用程序不会更改该状态。例如，RDS 代理目前与创建跨所有事务持久存在的临时表的存储过程不兼容。

删除 RDS 代理

如果您不再需要某个代理，可以将其删除。或者，如果您停止使用数据库实例或与其关联的集群，则可以删除代理。

Amazon Web Services Management Console

删除代理

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Proxies (代理)。
3. 从列表中选择要删除的代理。

4. 选择 Delete Proxy (删除代理)。

Amazon CLI

要删除数据库代理，请使用 Amazon CLI 命令 delete-db-proxy。要删除相关的关联，还可以使用 deregister-db-proxy-targets 命令。

```
aws rds delete-db-proxy --name proxy_name
```

```
aws rds deregister-db-proxy-targets
    --db-proxy-name proxy_name
    [--target-group-name target_group_name]
    [--target-ids comma_separated_list]       # or
    [--db-instance-identifiers instance_id]       # or
    [--db-cluster-identifiers cluster_id]
```

RDS API

要删除数据库代理，请调用 Amazon RDS API 函数 DeleteDBProxy。要删除相关的项目和关联，您还可以调用函数 DeleteDBProxyTargetGroup 和 DeregisterDBProxyTargets。

使用 Amazon RDS Proxy 终端节点

RDS 代理端点提供了灵活且高效地管理数据库连接的方式，从而提高了可扩展性、可用性和安全性。使用代理端点，您可以：

- 简化监控和故障排除：使用多个端点来独立跟踪和管理来自不同应用程序的连接。

主题

- 删除代理终端节点

代理终端节点概述

使用 RDS 代理终端节点所涉及的过程与使用 RDS 实例终端节点相同。如果您不熟悉 RDS 终端节点，请参阅连接到运行 MySQL 数据库引擎的数据库实例和连接到运行 PostgreSQL 数据库引擎的数据库实例以了解更多信息。

当您创建代理端点时，您可以将它与不同于代理虚拟私有云（VPC）的 VPC 相关联。这样，您就可以从其它 VPC 连接到代理，例如组织内其它应用程序使用的 VPC。

有关与代理终端节点关联的限制的信息，请参阅 代理端点的限制。

RDS 代理日志在每个条目前添加关联代理端点的名称作为前缀。这可以是您为用户定义的端点指定的名称，也可以是代理默认读/写端点的特殊名称 `default`。

每个代理终端节点都有自己的 CloudWatch 指标集合。监控所有代理端点、特定端点的指标，或者代理的所有读/写或只读端点的指标。有关更多信息，请参阅 使用 Amazon CloudWatch 监控 RDS Proxy 指标。

代理终端节点使用与其关联的代理相同的身份验证机制。RDS Proxy 会自动为用户定义的终端节点设置权限和授权，并与关联代理的属性保持一致。

代理端点的限制

RDS 代理端点具有以下限制：

- 无法修改 RDS 代理默认端点。
- 代理的用户定义终端节点的最大数量为 20 个。因此，代理最多可以有 21 个终端节点：默认终端节点加上您创建的 20 个终端节点。
- 当您将其他终端节点与代理关联时，RDS Proxy 会自动确定集群中的哪些数据库实例用于每个终端节点。

创建代理时，RDS 会自动创建 VPC 端点，以用于应用程序和数据库之间的安全通信。VPC 端点是可见的，可从 Amazon VPC 控制台访问该端点。

添加新的代理端点会预置一个 Amazon PrivateLink 接口端点。如果您向代理添加一个或多个端点，则需支付额外费用。有关更多信息，请参阅 RDS 代理定价。

多可用区数据库集群的代理端点

默认情况下，在将 RDS 代理与多可用区数据库集群结合使用时所连接的终端节点具有读/写功能。因此，此端点会将所有请求发送到集群的写入器实例。所有这些连接都计入写入器实例的 `max_connections` 值。如果您的代理与多可用区数据库集群关联，您可以为该代理创建额外的读/写或只读终端节点。

您可以将只读端点与代理一起用于只读查询。使用方式与将读取器终端节点用于多可用区数据库集群的方式相同。这样做有助于您利用具有一个或多个读取器数据库实例的多可用区数据库集群的读取可扩展性。通过使用只读终端节点并根据需要向多可用区数据库集群中添加更多读取器数据库实例，您可以同时运行更多的查询并同时建立更多的连接。这些读取器终端节点有助于提高查询密集型应用程序的读取可扩展性。如果集群中的读取器数据库实例变得不可用，读取器终端节点还有助于提高连接的可用性。

多可用区数据库集群的读取器终端节点

通过 RDS Proxy，您可以创建和使用读取器终端节点。但是，这些终端节点仅适用于与多可用区数据库集群关联的代理。如果您使用 RDS CLI 或 API，则可能会看到值为 `TargetRole` 的属性 `READ_ONLY`。您可以通过将代理的目标从 RDS 数据库实例更改为多可用区数据库集群来利用此类代理。

在将 RDS 代理与多可用区数据库集群结合使用时，可以创建并连接到称为读取器终端节点的只读终端节点。

读取器终端节点如何帮助提高应用程序可用性

在某些情况下，集群中的读取器实例可能会变得不可用。如果发生这种情况，使用数据库代理的读取器终端节点的连接比使用多可用区数据库集群读取器终端节点的连接恢复得更快。RDS 代理仅将连接路由到集群中的可用读取器实例。当实例变为不可用时，不会由于 DNS 缓存而造成延迟。

如果连接是多路复用的，则 RDS 代理会将后续查询定向到其它读取器实例，而不会中断您的应用程序。如果读取器实例处于不可用状态，则与该实例终端节点的所有客户端连接都将关闭。

如果连接是固定的，则该连接中的下一个查询将返回错误。但是，您的应用程序可以立即重新连接到同一代理端点。RDS Proxy 会将连接路由到处于 `available` 状态的其他读取器数据库实例。在手动重新连接时，RDS 代理不会检查新旧读取器实例之间的复制滞后。

如果您的多可用区数据库集群没有可用的读取器实例，则 RDS 代理会尝试在某个读取器终端节点可用时尝试连接到该终端节点。如果在连接借用超时期间没有读取器实例变为可用，则连接尝试将失败。如果读取器实例变为可用，则连接尝试成功。

读取器终端节点如何帮助提高查询可扩展性

代理的读取器终端节点通过以下方式帮助提高多可用区数据库集群查询可扩展性：

- 在可行的情况下，RDS Proxy 使用特定的读取器终端节点连接对所有查询问题使用相同的读取器数据库实例。这样，对相同表的一组相关查询就可以利用特定数据库实例上的缓存、计划优化等优势。
- 如果读取器数据库实例变为不可用，则对应用程序的影响取决于会话是多路复用的还是固定的。如果会话是多路复用的，则 RDS Proxy 会将所有后续查询路由到其他读取器数据库实例，而无需您执行任何操作。如果会话是固定的，则您的应用程序将出现错误，必须重新连接。您可以立即重新连接到读取器终端节点，RDS Proxy 会将连接路由到可用的读取器数据库实例。有关代理会话的多路复用和固定的更多信息，请参阅 RDS 代理概念概述。

访问 VPC 中的 RDS 数据库

默认情况下，RDS 技术堆栈的组件都在同一个 Amazon VPC 中。例如，假设一个在 Amazon EC2 实例上运行的应用程序连接到 Amazon RDS 数据库实例。在这种情况下，应用程序服务器和数据库必须都位于同一 VPC 内。

通过 RDS 代理，您可以设置从一个 VPC 中的资源（例如，EC2 实例）访问另一个 VPC 中的 Amazon RDS 数据库实例。例如，您的组织可能有多个应用程序访问同一数据库资源。每个应用程序都可能位于自己的 VPC 中。

要启用跨 VPC 访问，您需要为代理创建一个新的终端节点。代理本身与 Amazon RDS 数据库实例位于同一个 VPC 中。但是，跨 VPC 终端节点与 EC2 实例等其他资源一起位于另一个 VPC 中。跨 VPC 终端节点与 EC2 和其他资源所在的 VPC 中的子网和安全组关联。通过这些关联，您可以从原本由于 VPC 限制而无法访问数据库的应用程序连接到终端节点。

以下步骤说明了如何通过 RDS Proxy 创建和访问跨 VPC 终端节点：

1. 创建两个 VPC，或者选择已用于 RDS 工作的两个 VPC。每个 VPC 都应该有自己的关联网络资源，例如互联网网关、路由表、子网和安全组。如果您只有一个 VPC，可以查阅 开始使用 Amazon RDS，了解设置另一个 VPC 以成功使用 RDS 的步骤。您还可以查看 Amazon EC2 控制台中的现有 VPC，以了解要将哪些类型的资源连接起来。
2. 创建一个与要连接到的 Amazon RDS 数据库实例关联的数据库代理。按照 为 Amazon RDS 创建代理 中的过程操作。
3. 在 RDS 控制台中的代理的 Details（详细信息）页面上，在 Proxy endpoints（代理终端节点）部分下，选择 Create endpoint（创建终端节点）。按照 创建代理终端节点 中的过程操作。
4. 选择将跨 VPC 终端节点设置为读/写还是只读。

5. 不接受默认的与 Amazon RDS 数据库实例使用相同的 VPC，而是选择其它 VPC。此 VPC 必须与代理所在的 VPC 位于同一 Amazon 区域中。
6. 现在不接受来自 Amazon RDS 数据库实例所在 VPC 中的默认子网和安全组，而是进行新的选择。根据您选择的 VPC 中的子网和安全组进行这些选择。
7. 您无需更改 Secrets Manager 密钥的任何设置。相同的凭证适用于代理的所有终端节点，无论每个终端节点位于哪个 VPC 中均是如此。
8. 等待新终端节点达到 Available（可用）状态。
9. 记下终端节点的完整名称。这是以 *Region_name*`.rds.amazonaws.com` 为结尾的值，您提供此值作为数据库应用程序连接字符串的一部分。
10. 从与该终端节点位于同一 VPC 中的资源访问新的终端节点。测试此过程的一种简单方法是在此 VPC 中创建一个新的 EC2 实例。然后，登录 EC2 实例并运行 `mysql` 或 `psql` 命令，以使用连接字符串中的端点值进行连接。

创建代理终端节点

要创建代理端点，请按照以下说明操作：

控制台

创建代理端点

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Proxies (代理)。
3. 单击您要为其创建新终端节点的代理的名称。

   此时将显示该代理的详细信息页面。
4. 在 Proxy endpoints（代理终端节点）部分中，选择 Create proxy endpoint（创建代理终端节点）。

   此时将显示 Create proxy endpoint（创建代理终端节点）窗口。
5. 对于 Proxy endpoint name（代理终端节点名称），输入您选择的描述性名称。
6. 对于 Target role（目标角色），选择将终端节点设置为读/写还是只读。

   使用读/写端点的连接可以执行任何类型的操作，如数据定义语言（DDL）语句、数据操作语言（DML）语句和查询。这些终端节点始终连接到 RDS 数据库集群的主实例。仅在应用程序中使用

单个终端节点时，可以将读/写终端节点用于常规数据库操作。您还可以将读/写终端节点用于管理操作、联机事务处理 (OLTP) 应用程序和提取-转换-加载 (ETL) 作业。

使用只读终端节点的连接只能执行查询。RDS 代理可以为终端节点的每个连接使用其中一个读取器实例。这样，查询密集型应用程序就可以利用多可用区数据库集群的集群功能。这些只读连接不会在集群的主实例上增加任何开销。这样，您的报告和分析查询就不会减慢 OLTP 应用程序的写入操作。

7. 对于虚拟私有云（VPC），选择默认值，以便从您通常用于访问代理或其关联数据库的同一 EC2 实例或其它资源访问端点。要为此代理设置跨 VPC 访问，请选择原定设置值以外的 VPC。有关跨 VPC 访问的更多信息，请参阅 [访问 VPC 中的 RDS 数据库]。
8. 对于子网，预设情况下，RDS Proxy 会填充与关联代理相同的子网。要限制对端点的访问，以便只有 VPC 的地址范围的一部分能够与其连接，请删除一个或多个子网。
9. 对于 VPC security group (VPC 安全组)，您可以选择现有安全组或创建新安全组。默认情况下，RDS Proxy 会填充与关联代理相同的一个或多个安全组。如果代理的入站和出站规则适用于此端点，则保留默认选项。

   如果选择创建新的安全组，请在此页面上为该安全组指定一个名称。然后，从 EC2 控制台编辑安全组设置。
10. 选择 Create proxy endpoint（创建代理终端节点）。

Amazon CLI

要创建代理终端节点，请使用 Amazon CLIcreate-db-proxy-endpoint 命令。

包括以下必需参数：

- `--db-proxy-name` *`value`*
- `--db-proxy-endpoint-name` *`value`*
- `--vpc-subnet-ids` *`list_of_ids`*。用空格分隔子网 ID。您无需指定 VPC 本身的 ID。

您还可以包括以下可选参数：

- `--target-role { READ_WRITE | READ_ONLY }`此参数默认为 `READ_WRITE`。当代理与仅包含写入器数据库实例的多可用区数据库集群相关联时，您不能指定 `READ_ONLY`。有关将只读端点与多可用区数据库集群结合使用的详细信息，请参阅 多可用区数据库集群的读取器终端节点。

- `--vpc-security-group-ids` *value*。用空格分隔安全组 ID。如果省略此参数，则 RDS Proxy 使用 VPC 的默认安全组。RDS Proxy 根据您为 `--vpc-subnet-ids` 参数指定的子网 ID 确定 VPC。

Example

下面的示例用于创建一个名为 `my-endpoint` 的代理终端节点。

对于 Linux、macOS 或 Unix：

```
aws rds create-db-proxy-endpoint \
  --db-proxy-name my-proxy \
  --db-proxy-endpoint-name my-endpoint \
  --vpc-subnet-ids subnet_id subnet_id subnet_id ... \
  --target-role READ_ONLY \
  --vpc-security-group-ids security_group_id ]
```

对于 Windows：

```
aws rds create-db-proxy-endpoint ^
  --db-proxy-name my-proxy ^
  --db-proxy-endpoint-name my-endpoint ^
  --vpc-subnet-ids subnet_id_1 subnet_id_2 subnet_id_3 ... ^
  --target-role READ_ONLY ^
  --vpc-security-group-ids security_group_id
```

RDS API

要创建代理端点，请使用 RDS API CreateDBProxyEndpoint 操作。

查看代理终端节点

要查看现有代理端点，请按照以下说明进行操作：

控制台

查看代理终端节点的详细信息

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Proxies (代理)。

3. 在列表中，选择要查看其终端节点的代理。单击代理名称以查看其详细信息页面。
4. 在 Proxy endpoints（代理终端节点）部分中，选择要查看的终端节点。单击其名称以查看详细信息页面。
5. 检查您对其值感兴趣的参数。您可以检查以下属性：

   - 终端节点是读/写还是只读。
   - 您在数据库连接字符串中使用的终端节点地址。
   - 与终端节点关联的 VPC、子网和安全组。

Amazon CLI

要查看一个或多个代理端点，请使用 Amazon CLI describe-db-proxy-endpoints 命令。

您可以包括以下参数：

- `--db-proxy-endpoint-name`
- `--db-proxy-name`

以下示例描述了 `my-endpoint` 代理终端节点：

Example

对于 Linux、macOS 或 Unix：

```
aws rds describe-db-proxy-endpoints \
  --db-proxy-endpoint-name my-endpoint
```

对于 Windows：

```
aws rds describe-db-proxy-endpoints ^
  --db-proxy-endpoint-name my-endpoint
```

RDS API

要描述一个或多个代理终端节点，请使用 RDS API DescribeDBProxyEndpoints 操作。

修改代理终端节点

要修改代理端点，请按照以下说明进行操作：

控制台

修改一个或多个代理终端节点

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Proxies (代理)。
3. 在列表中，选择要修改其终端节点的代理。单击代理名称以查看其详细信息页面。
4. 在 Proxy endpoints（代理终端节点）部分中，选择要修改的终端节点。您可以在列表中选择它，也可以单击其名称以查看详细信息页面。
5. 在代理详细信息页面上的 Proxy endpoints（代理终端节点）部分中，选择 Edit（编辑）。或者，在代理端点详细信息页面上，对操作，选择编辑。
6. 更改要修改的参数的值。
7. 选择 Save changes（保存更改）。

Amazon CLI

要修改代理端点，请使用 Amazon CLI modify-db-proxy-endpoint 命令和以下必需参数：

- `--db-proxy-endpoint-name`

使用以下一个或多个参数指定对终端节点属性的更改：

- `--new-db-proxy-endpoint-name`
- `--vpc-security-group-ids`。用空格分隔安全组 ID。

以下示例用于将 `my-endpoint` 代理终端节点重命名为 `new-endpoint-name`。

Example

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-proxy-endpoint \
  --db-proxy-endpoint-name my-endpoint \
  --new-db-proxy-endpoint-name new-endpoint-name
```

对于 Windows：

```
aws rds modify-db-proxy-endpoint ^
  --db-proxy-endpoint-name my-endpoint ^
  --new-db-proxy-endpoint-name new-endpoint-name
```

RDS API

要修改代理终端节点，请使用 RDS API ModifyDBProxyEndpoint 操作。

删除代理终端节点

要删除代理的端点，请按照以下说明操作：

> **Note**
>
> 您无法删除 RDS 代理自动为每个代理创建的默认代理端点。
>
> 删除代理时，RDS Proxy 会自动删除所有关联的终端节点。

控制台

使用 Amazon Web Services Management Console 删除代理终端节点

1. 在导航窗格中，选择 Proxies (代理)。
2. 在列表中，选择要删除其终端节点的代理。单击代理名称以查看其详细信息页面。
3. 在 Proxy endpoints（代理终端节点）部分中，选择要删除的终端节点。您可以在列表中选择一个或多个终端节点，或单击单个终端节点的名称以查看详细信息页面。
4. 在代理详细信息页面上的 Proxy endpoints（代理终端节点）部分中，选择 Delete（删除）。或者，在代理端点详细信息页面上，对于操作，选择编辑。

Amazon CLI

要删除代理终端节点，请运行 delete-db-proxy-endpoint 命令并包含以下必需参数：

- `--db-proxy-endpoint-name`

以下命令用于删除名为 `my-endpoint` 的代理终端节点。

对于 Linux、macOS 或 Unix：

```
aws rds delete-db-proxy-endpoint \
  --db-proxy-endpoint-name my-endpoint
```

对于 Windows：

```
aws rds delete-db-proxy-endpoint ^
  --db-proxy-endpoint-name my-endpoint
```

RDS API

要使用 RDS API 删除代理终端节点，请运行 DeleteDBProxyEndpoint 操作。对于 `DBProxyEndpointName` 参数，指定代理终端节点的名称。

使用 Amazon CloudWatch 监控 RDS Proxy 指标

您可以使用 Amazon CloudWatch 监控 RDS Proxy。CloudWatch 可从代理收集和处理原始数据，并将数据处理为便于读取的近乎实时的指标。要在 CloudWatch 控制台中查找这些指标，请依次选择 Metrics (指标)、RDS 和 Per-Proxy Metrics (每个代理指标)。有关更多信息，请参阅 Amazon CloudWatch 用户指南 中的使用 Amazon CloudWatch 指标。

> **Note**
>
> RDS 为与代理关联的每个底层 Amazon EC2 实例发布这些指标。一个代理可能由多个 EC2 实例提供服务。使用 CloudWatch 统计数据可汇总所有关联实例的代理值。
> 其中一些指标可能在代理第一次成功连接之后才可见。

在 RDS Proxy 日志中，每个条目都以关联的代理终端节点的名称作为前缀。此名称可以是您为用户定义的端点指定的名称，或者是执行读/写请求的代理的默认端点的特殊名称 `default`。

所有 RDS Proxy 指标都在 `proxy` 组中。

每个代理终端节点都有自己的 CloudWatch 指标。您可以单独监控每个代理终端节点的使用情况。有关代理终端节点的更多信息，请参阅 使用 Amazon RDS Proxy 终端节点。

您可以使用以下维度集之一聚合每个指标的值。例如，通过使用 `ProxyName` 维度集，您可以分析特定代理的所有流量。通过使用其他维度集，您可以用不同的方式拆分指标。您可以根据每个代理的不同终端节点或目标数据库，或者每个数据库的读/写和只读流量来拆分指标。

- 维度集 1：`ProxyName`

- 维度集 2：ProxyName、EndpointName
- 维度集 3：ProxyName、TargetGroup、Target
- 维度集 4：ProxyName、TargetGroup、TargetRole

指标	描述	有效期	CloudWatch 维度集
AvailabilityPercentage	目标组在维度指示的角色中可用的时间百分比。每分钟报告一次此指标。此指标最有用的统计数据是 Average。	1 minute	Dimension set 4
ClientConnections	当前客户端连接数。每分钟报告一次此指标。此指标最有用的统计数据是 Average。	1 minute	Dimension set 1, Dimension set 2
ClientConnectionsClosed	已关闭的客户端连接数。此指标最有用的统计数据是 Sum。	1 分钟及更久	Dimension set 1, Dimension set 2
ClientConnectionsInSetup	已打开但尚未完成设置的当前客户端连接数量。每分钟报告一次此指标。该指标最有用的统计数据是 Sum。	1 minute	Dimension set 1, Dimension set 2
ClientConnectionsNoTLS	当前没有传输层安全性 (TLS) 的客户端连接的数量。每分钟报告一次此指标。此指标最有用的统计数据是 Average。	1 minute	Dimension set 1, Dimension set 2

指标	描述	有效期	CloudWatch 维度集
ClientConnectionsReceived	收到的客户端连接请求数。此指标最有用的统计数据是 Sum。	1 分钟及更久	Dimension set 1, Dimension set 2
ClientConnectionsSetupFailedAuth	由于身份验证或 TLS 配置错误而失败的客户端连接尝试次数。此指标最有用的统计数据是 Sum。	1 分钟及更久	Dimension set 1, Dimension set 2
ClientConnectionsSetupSucceeded	使用任何具有或不具有 TLS 的身份验证机制成功建立的客户端连接数。此指标最有用的统计数据是 Sum。	1 分钟及更久	Dimension set 1, Dimension set 2
ClientConnectionsTLS	当前具有 TLS 的客户端连接的数量。每分钟报告一次此指标。此指标最有用的统计数据是 Average。	1 minute	Dimension set 1, Dimension set 2
DatabaseConnectionRequests	创建数据库连接的请求数。此指标最有用的统计数据是 Sum。	1 分钟及更久	Dimension set 1, Dimension set 3, Dimension set 4
DatabaseConnectionRequestsWithTLS	创建具有 TLS 的数据库连接的请求数。此指标最有用的统计数据是 Sum。	1 分钟及更久	Dimension set 1, Dimension set 3, Dimension set 4

指标	描述	有效期	CloudWatch 维度集
DatabaseConnections	当前数据库连接数。每分钟报告一次此指标。此指标最有用的统计数据是 Average。	1 minute	Dimension set 1, Dimension set 3, Dimension set 4
DatabaseConnectionsBorrowLatency	被监视的代理建立数据库连接所花费的时间（以微秒为单位）。此指标最有用的统计数据是 Average。	1 分钟及更久	Dimension set 1, Dimension set 2
DatabaseConnectionsCurrentlyBorrowed	当前处于借用状态的数据库连接数。每分钟报告一次此指标。此指标最有用的统计数据是 Average。	1 minute	Dimension set 1, Dimension set 3, Dimension set 4
DatabaseConnectionsCurrentlyInTransaction	事务中的当前数据库连接数。每分钟报告一次此指标。此指标最有用的统计数据是 Average。	1 minute	Dimension set 1, Dimension set 3, Dimension set 4
DatabaseConnectionsCurrentlySessionPinned	由于客户端请求中更改会话状态的操作，当前固定的当前数据库连接数。每分钟报告一次此指标。此指标最有用的统计数据是 Average。	1 minute	Dimension set 1, Dimension set 3, Dimension set 4
DatabaseConnectionsSetupFailed	失败的数据库连接请求数。此指标最有用的统计数据是 Sum。	1 分钟及更久	Dimension set 1, Dimension set 3, Dimension set 4

指标	描述	有效期	CloudWatch 维度集
DatabaseConnectionsSetupSucceeded	使用或不使用 TLS 成功建立的数据库连接数。此指标最有用的统计数据是 Sum。	1 分钟及更久	Dimension set 1, Dimension set 3, Dimension set 4
DatabaseConnectionsWithTLS	具有 TLS 的当前数据库连接数。每分钟报告一次此指标。此指标最有用的统计数据是 Average。	1 minute	Dimension set 1, Dimension set 3, Dimension set 4
MaxDatabaseConnectionsAllowed	允许的最大数据库连接数。每分钟报告一次此指标。此指标最有用的统计数据是 Average。	1 minute	Dimension set 1, Dimension set 3, Dimension set 4
QueryDatabaseResponseLatency	数据库响应查询所花费的时间（以微秒为单位）。此指标最有用的统计数据是 Average。	1 分钟及更久	Dimension set 1, Dimension set 2, Dimension set 3, Dimension set 4
QueryRequests	收到的查询数。包含多个语句的查询被计为一个查询。此指标最有用的统计数据是 Sum。	1 分钟及更久	Dimension set 1, Dimension set 2
QueryRequestsNoTLS	从非 TLS 连接收到的查询数。包含多个语句的查询被计为一个查询。此指标最有用的统计数据是 Sum。	1 分钟及更久	Dimension set 1, Dimension set 2

指标	描述	有效期	CloudWatch 维度集
QueryRequestsTLS	从 TLS 连接收到的查询数。包含多个语句的查询被计为一个查询。此指标最有用的统计数据是 Sum。	1 分钟及更久	Dimension set 1, Dimension set 2
QueryResponseLatency	从获取查询请求到代理响应该请求的时间（以微秒为单位）。此指标最有用的统计数据是 Average。	1 分钟及更久	Dimension set 1, Dimension set 2

您可以在Amazon Web Services Management Console中的 CloudWatch 下找到 RDS Proxy 活动的日志。每个代理在 Log groups (日志组) 页面中都有一个条目。

> **Important**
>
> 这些日志供相关人员进行故障排除，而不是用于编程访问。日志的格式和内容可能会发生变化。
> 特别是，较旧的日志不包含指示每个请求的终端节点的任何前缀。在较新的日志中，每个条目都以关联的代理终端节点的名称作为前缀。此名称可以是您为用户定义的终端节点指定的名称，也可以是使用代理的默认终端节点的请求的特殊名称 default。

使用 RDS 代理事件

事件表示环境中发生的更改，例如 Amazon 环境或软件即服务（SaaS）合作伙伴中的服务或应用程序。也可以是您自己的一个自定义应用程序或服务。例如，Amazon RDS 在创建或修改 RDS Proxy 时生成事件。Amazon RDS 将事件近乎实时地传输到 Amazon EventBridge。接下来，您可以查找可以订阅的 RDS Proxy 事件列表以及 RDS Proxy 事件示例。

有关使用事件的详细信息，请参阅以下内容：

- 有关如何使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 查看事件的说明，请参阅 查看 Amazon RDS 事件。

- 若要了解如何配置 Amazon RDS 向 EventBridge 发送事件，请参阅 创建对 Amazon RDS 事件触发的规则。

RDS Proxy 事件

下表显示了 RDS Proxy 为源类型时的事件类别和事件列表。

类别	RDS 事件 ID	消息	注意
配置更改	RDS-EVENT-0204	RDS 修改了数据库代理 *name*。	无
配置更改	RDS-EVENT-0207	RDS 修改了数据库代理 *name* 的端点。	无
配置更改	RDS-EVENT-0213	RDS 检测到已添加数据库实例并自动将其添加到数据库代理 *name* 的目标组。	无
配置更改	RDS-EVENT-0214	RDS 检测到已删除数据库实例 *name* 并自动将其从数据库代理 *name* 的目标组 *name* 中删除。	无
配置更改	RDS-EVENT-0215	RDS 检测到已删除数据库集群 *name* 并自动将其从数据库代理 *name* 的目标组 *name* 中删除。	无
创建	RDS-EVENT-0203	RDS 创建了数据库代理 *name*。	无
创建	RDS-EVENT-0206	RDS 为数据库代理 *name* 创建了端点 *name*。	无
删除	RDS-EVENT-0205	RDS 删除了数据库代理 *name*。	无

类别	RDS 事件 ID	消息	注意
删除	RDS-EVENT-0208	RDS 删除了数据库代理 *name* 的端点 *name*。	无
失败	RDS-EVENT-0243	RDS 无法为代理 *name* 预调配容量，因为您的子网 *name* 中没有足够的 IP 地址可用。要解决此问题，请确保您的子网具有最少的未使用 IP 地址数（如 RDS 代理文档中所建议）。	要确定您的实例类的建议数量，请参阅[计划 IP 地址容量](#)。
失败	RDS-EVENT-0275	RDS 限制了一些与数据库代理*##*的连接。从客户端到代理的并发连接请求数已超出了限制。	无

以下是采用 JSON 格式的 RDS 代理事件示例。该事件显示 RDS 已修改名为 `my-rds-proxy` 的 RDS Proxy 的名为 `my-endpoint` 的端点。事件 ID 为 RDS-EVENT-0207。


```
{
  "version": "0",
  "id": "68f6e973-1a0c-d37b-f2f2-94a7f62ffd4e",
  "detail-type": "RDS DB Proxy Event",
  "source": "aws.rds",
  "account": "123456789012",
  "time": "2018-09-27T22:36:43Z",
  "region": "us-east-1",
  "resources": [
     "arn:aws:rds:us-east-1:123456789012:db-proxy:my-rds-proxy"
  ],
  "detail": {
    "EventCategories": [
      "configuration change"
    ],
    "SourceType": "DB_PROXY",
    "SourceArn": "arn:aws:rds:us-east-1:123456789012:db-proxy:my-rds-proxy",
    "Date": "2018-09-27T22:36:43.292Z",
    "Message": "RDS modified endpoint my-endpoint of DB Proxy my-rds-proxy.",
```

```
    "SourceIdentifier": "my-endpoint",
    "EventID": "RDS-EVENT-0207"
  }
}
```

RDS 代理故障排除

下面，您可以找到一些常见 RDS 代理问题的故障排除思路，以及有关 RDS Proxy 的 CloudWatch Logs 的信息。

在 RDS Proxy 日志中，每个条目都以关联的代理终端节点的名称作为前缀。此名称可以是您为用户定义的端点指定的名称。或者，对于执行读/写请求的代理的默认端点，它可以是特殊名称 `default`。有关代理终端节点的更多信息，请参阅 使用 Amazon RDS Proxy 终端节点。

主题

验证代理的连接

您可以使用以下命令验证连接中的所有组件（例如代理、数据库和计算实例）是否均可相互进行通信。

使用 describe-db-proxies 命令检查代理本身。此外，使用 describe-db-proxy-target-groups 命令检查关联的目标组。检查目标的详细信息是否与您打算与代理关联的 RDS 数据库实例匹配。使用如下命令。

```
aws rds describe-db-proxies --db-proxy-name $DB_PROXY_NAME
aws rds describe-db-proxy-target-groups --db-proxy-name $DB_PROXY_NAME
```

要确认代理可以连接到底层数据库，请使用 describe-db-proxy-targets 命令检查目标组中指定的目标。使用如下命令。

```
aws rds describe-db-proxy-targets --db-proxy-name $DB_PROXY_NAME
```

describe-db-proxy-targets 命令的输出包括一个 `TargetHealth` 字段。您可以检查 `State` 内部的字段 `Reason`、`Description` 和 `TargetHealth`，以检查代理是否可以与底层数据库实例进行通信。

- State 值 AVAILABLE 表示代理可以连接到数据库实例。
- State 值 UNAVAILABLE 表示存在临时或永久的连接问题。在这种情况下，请检查 Reason 和 Description 字段。例如，如果 Reason 的值为 PENDING_PROXY_CAPACITY，请在代理完成其扩展操作后，重试连接。如果 Reason 的值为 UNREACHABLE、CONNECTION_FAILED 或 AUTH_FAILURE，请使用 Description 字段中的说明来帮助您诊断问题。
- 在更改为 State 或 REGISTERING 之前，AVAILABLE 字段的值可能在短时间内为 UNAVAILABLE。

如果以下 Netcat 命令 (nc) 成功，您可以从登录的 EC2 实例或其他系统访问代理终端节点。如果您与代理和关联数据库不在同一 VPC 中，则该命令将报告失败。您可能未在同一 VPC 中就可以直接登录到数据库。但是，除非在同一 VPC 中，否则无法登录到代理。

```
nc -zx MySQL_proxy_endpoint 3306

nc -zx PostgreSQL_proxy_endpoint 5432
```

您可以使用以下命令来确保 EC2 实例具有所需属性。特别是，EC2 实例的 VPC 必须与代理连接到的的 VPC 相同。

```
aws ec2 describe-instances --instance-ids your_ec2_instance_id
```

检查用于代理的 Secrets Manager 密钥。

```
aws secretsmanager list-secrets
aws secretsmanager get-secret-value --secret-id your_secret_id
```

确保 get-secret-value 显示的 SecretString 字段被编码为包含 username 和 password 字段的 JSON 字符串。以下示例显示了 SecretString 字段的格式。

```
{
  "ARN": "some_arn",
  "Name": "some_name",
  "VersionId": "some_version_id",
  "SecretString": '{"username":"some_username","password":"some_password"}',
  "VersionStages": [ "some_stage" ],
  "CreatedDate": some_timestamp
}
```

常见问题和解决方案

本节介绍使用 RDS 代理时的一些常见问题和潜在的解决方案。

运行 `aws rds describe-db-proxy-targets` CLI 命令后，如果 `TargetHealth` 描述显示 `Proxy does not have any registered credentials`，请验证以下内容：

- 已经为用户注册了访问代理的凭证。
- 用于访问代理使用的 Secrets Manager 密钥的 IAM 角色是有效的。

创建或连接数据库代理时，您可能会遇到以下 RDS 事件。

类别	RDS 事件 ID	描述
失败	RDS-EVENT-0243	RDS 无法为代理预调配容量，因为您的子网中没有足够的 IP 地址可用。要解决此问题，请确保您的子网具有最少的未使用 IP 地址数。要确定您的实例类的建议数量，请参阅计划 IP 地址容量。
失败	RDS-EVENT-0275	RDS 限制了一些与数据库代理##的连接。从客户端到代理的并发连接请求数已超出了限制。

创建新代理或连接到代理时，您可能会遇到以下问题。

错误	原因或解决方法
`403: The security token included in the request is invalid`	选择现有 IAM 角色，而不是选择创建新角色。

排查 RDS for MySQL 的 RDS 代理问题

连接到 MySQL 代理时，您可能会遇到以下问题。

错误	原因或解决方法
`ERROR 1040 (HY000): Connections rate limit exceeded (`*`limit_value`*` )`	从客户端到代理的连接请求速率超过了限制。
`ERROR 1040 (HY000): IAM authentication rate limit exceeded`	从客户端到代理的具有 IAM 身份验证的并发请求数超过了限制。
`ERROR 1040 (HY000): Number simultaneous connections exceeded (`*`limit_value`*` )`	从客户端到代理的并发连接请求数超出了限制。
`ERROR 1045 (28000): Access denied for user '`*`DB_USER`*`'@'%' (usi password: YES)`	代理使用的 Secrets Manager 密钥与现有数据库用户的用户名和密码不匹配。更新 Secrets Manager 密钥中的凭证，或者确保数据库用户存在并且具有与密钥中的密码相同的密码。
`ERROR 1105 (HY000): Unknown error`	出现未知错误。
`ERROR 1231 (42000):`	为 `character_set_client` 参数设置的值无效。例如，值 `ucs2` 无效，因为它会导致 MySQL 服务器崩溃。

错误	原因或解决方法
Variable ''character_set_client'' can't be set to the value of *value*	
ERROR 3159 (HY000): This RDS Proxy requires TLS connections.	您在代理中启用了需要传输层安全性设置，但您的连接将参数 ssl-mode=DISABLED 包含在 MySQL 客户端中。请执行以下任一操作： • 禁用代理的需要传输层安全性设置。 • 在 MySQL 客户端中，使用最小设置 ssl-mode=REQUIRED 连接到数据库。
ERROR 2026 (HY000): SSL connection error: Internal Server *Error*	与代理的 TLS 握手失败。一些可能的原因包括： • SSL 是必需的，但服务器不支持它。 • 发生内部服务器错误。 • 发生握手错误。
ERROR 9501 (HY000): Timed-out waiting to acquire database connection	等待获取数据库连接时代理超时。一些可能的原因包括： • 代理无法建立数据库连接，因为已达到最大连接数 • 代理无法建立数据库连接，因为数据库不可用。

排查 RDS for PostgreSQL 的 RDS 代理问题

连接到 PostgreSQL 代理时，您可能会遇到以下问题。

错误	原因	解决方案
ERROR 28000: IAM authentication is	用户在 PostgreSQL 客户端中，尝试使用具有设置	用户需要在 PostgreSQL 客户端中使用最小设置 sslmode=require 连接到数据库。有关

错误	原因	解决方案
`allowed only with SSL connections.`	`sslmode=disable` 的 IAM 身份验证连接到数据库。	更多信息，请参阅 PostgreSQL SSL 支持文档。
`ERROR 28000: This RDS proxy has no credentials for the role` *`role_name`* `. Check the credentials for this role and try again.`	这个角色没有 Secrets Manager 密钥。	为此角色添加 Secrets Manager 密钥。有关更多信息，请参阅 为 RDS 代理配置 IAM 身份验证。
`ERROR 28000: RDS supports only IAM, MD5, or SCRAM authentication.`	用于连接到代理的数据库客户端正在使用代理当前不支持的身份验证机制。	如果您不使用 IAM 身份验证，请使用 MD5 或 SCRAM 密码身份验证。
`ERROR 28000: A user name is missing from the connection startup packet. Provide a user name for this connection.`	用于连接到代理的数据库客户端在尝试建立连接时未发送用户名。	请确保在使用您选择的 PostgreSQL 客户端设置与代理的连接时定义用户名。
`ERROR 28000: IAM is allowed only with SSL connections.`	客户端尝试使用 IAM 身份验证进行连接，但未启用 SSL。	在 PostgreSQL 客户端中启用 SSL。
`ERROR 28000: This RDS Proxy requires TLS connections.`	用户启用了需要传输层安全性选项，但尝试在 PostgreSQL 客户端中使用 `sslmode=disable` 进行连接。	要修复此错误，请执行以下操作之一： • 禁用代理的需要传输层安全性选项。 • 在 PostgreSQL 客户端中，使用最小设置 `sslmode=allow` 连接到数据库。

错误	原因	解决方案
`ERROR 28P01: IAM authentication failed for user` *user_name* `. Check the IAM token for this user and try again.`	该错误可能是由于以下原因引起的： • 客户端提供了错误的 IAM 用户名。 • 客户端向用户提供了错误的 IAM 授权令牌。 • 客户端在使用没有所需权限的 IAM 策略。 • 客户端向用户提供了过期的 IAM 授权令牌。	要修复此错误，请执行以下操作： 1. 确认提供的 IAM 用户存在。 2. 确认 IAM 授权令牌属于提供的 IAM 用户。 3. 确认 IAM 策略具有足够的 RDS 权限。 4. 检查所用 IAM 授权令牌的有效性。
`ERROR 28P01: The password that was provided for the role` *role_name* `is wrong.`	此角色的密码与 Secrets Manager 密钥不匹配。	检查 Secrets Manager 中此角色的密钥，以查看密码是否与 PostgreSQL 客户端中使用的密码相同。
`ERROR 28P01: The IAM authentication failed for the role` *role_name* `. Check the IAM token for this role and try again.`	用于 IAM 身份验证的 IAM 令牌存在问题。	生成新的身份验证令牌并在新连接中使用它。
`ERROR 0A000: Feature not supported: RDS Proxy supports only version 3.0 of the PostgreSQL messaging protocol.`	用于连接到代理的 PostgreSQL 客户端使用早于 3.0 的协议。	使用支持 3.0 消息传递协议的较新 PostgreSQL 客户端。如果您使用的是 PostgreSQL psql CLI，请使用大于或等于 7.4 的版本。

错误	原因	解决方案
`ERROR 0A000: Feature not supported: RDS Proxy currently doesn't support streaming replication mode.`	用于连接到代理的 PostgreSQL 客户端正在尝试使用流复制模式，该模式目前不受 RDS 代理支持。	在用于连接的 PostgreSQL 客户端中关闭流复制模式。
`ERROR 0A000: Feature not supported: RDS Proxy currently doesn't support the option` *`option_name`* `.`	通过启动消息，用于连接到代理的 PostgreSQL 客户端正在请求 RDS 代理当前不支持的选项。	在用于连接的 PostgreSQL 客户端中，关闭上述消息中显示为不支持的选项。
`ERROR 53300: The IAM authentication failed because of too many competing requests.`	从客户端到代理的具有 IAM 身份验证的并发请求数超过了限制。	降低使用 PostgreSQL 客户端中的 IAM 身份验证建立连接的速率。
`ERROR 53300: The maximum number of client connections to the proxy exceeded` *`number_value`* `.`	从客户端到代理的并发连接请求数超出了限制。	减少从 PostgreSQL 客户端到此 RDS 代理的活动连接数。
`ERROR 53300: Rate of connection to proxy exceeded` *`number_value`* `.`	从客户端到代理的连接请求速率超过了限制。	降低从 PostgreSQL 客户端建立连接的速率。
`ERROR XX000: Unknown error.`	出现未知错误。	请联系 Amazon Support，以便我们调查此问题。

错误	原因	解决方案
ERROR 08000: Timed-out waiting to acquire database connection.	代理在 ConnectionBorrowTimeout 设置指定的持续时间内等待获取数据库连接时超时。一些可能的原因包括： • 代理无法建立数据库连接，因为已达到最大连接数。 • 代理无法建立数据库连接，因为数据库不可用，或者建立到数据库的连接所花费的时间超过了所配置的 ConnectionBorrowTimeout 。	可能的解决方案包括以下各项： • 避免固定代理连接。请参阅避免固定 RDS 代理。 • 检查 ConnectionBorrowTimeout 和 MaxConnectionsPercent 设置。请参阅RDS 代理连接注意事项。 • 查看目标可用性。请参阅使用 Amazon CloudWatch 监控 RDS Proxy 指标 中的 AvailabilityPercentage 。
ERROR XX000: Request returned an error: *database_error* .	从代理建立的数据库连接返回错误。	解决方案取决于特定的数据库错误。示例为：Request returned an error: database "your-database-name" does not exist。这意味着数据库服务器上不存在指定的数据库名称。或者，这意味着服务器上不存在用作数据库名称（如果未指定数据库名称）的用户名。

对已删除的 **postgres** 数据库进行故障排除

如果您错误地从实例中删除了 postgres 数据库，则需要还原数据库以还原与实例的连接。在数据库实例中运行以下命令：

```
CREATE DATABASE postgres;
```

```
GRANT CONNECT ON DATABASE postgres TO rdsproxyadmin;
```

将 RDS Proxy 与 Amazon CloudFormation 一起使用

您可以将 RDS Proxy 与 Amazon CloudFormation 一起使用。这有助于您创建相关资源组。此类组可能包含可连接到新创建的 Amazon RDS 数据库实例的代理。Amazon CloudFormation 中的 RDS Proxy 支持涉及两种新的注册表类型：`DBProxy` 和 `DBProxyTargetGroup`。

以下列表显示了 RDS Proxy 的 Amazon CloudFormation 模板示例。

```
Resources:
 DBProxy:
   Type: Amazon::RDS::DBProxy
   Properties:
     DBProxyName: CanaryProxy
     EngineFamily: MYSQL
     RoleArn:
      Fn::ImportValue: SecretReaderRoleArn
     Auth:
       - {AuthScheme: SECRETS, SecretArn: !ImportValue ProxySecret, IAMAuth: DISABLED}
     VpcSubnetIds:
       Fn::Split: [",", "Fn::ImportValue": SubnetIds]

 ProxyTargetGroup:
   Type: Amazon::RDS::DBProxyTargetGroup
   Properties:
     DBProxyName: CanaryProxy
     TargetGroupName: default
     DBInstanceIdentifiers:
       - Fn::ImportValue: DBInstanceName
   DependsOn: DBProxy
```

有关此示例中的资源的更多信息，请参阅 [DBProxy](#) 和 [DBProxyTargetGroup](#)。

有关您可以使用 Amazon CloudFormation 创建的资源的更多信息，请参阅 [RDS 资源类型参考](#)。

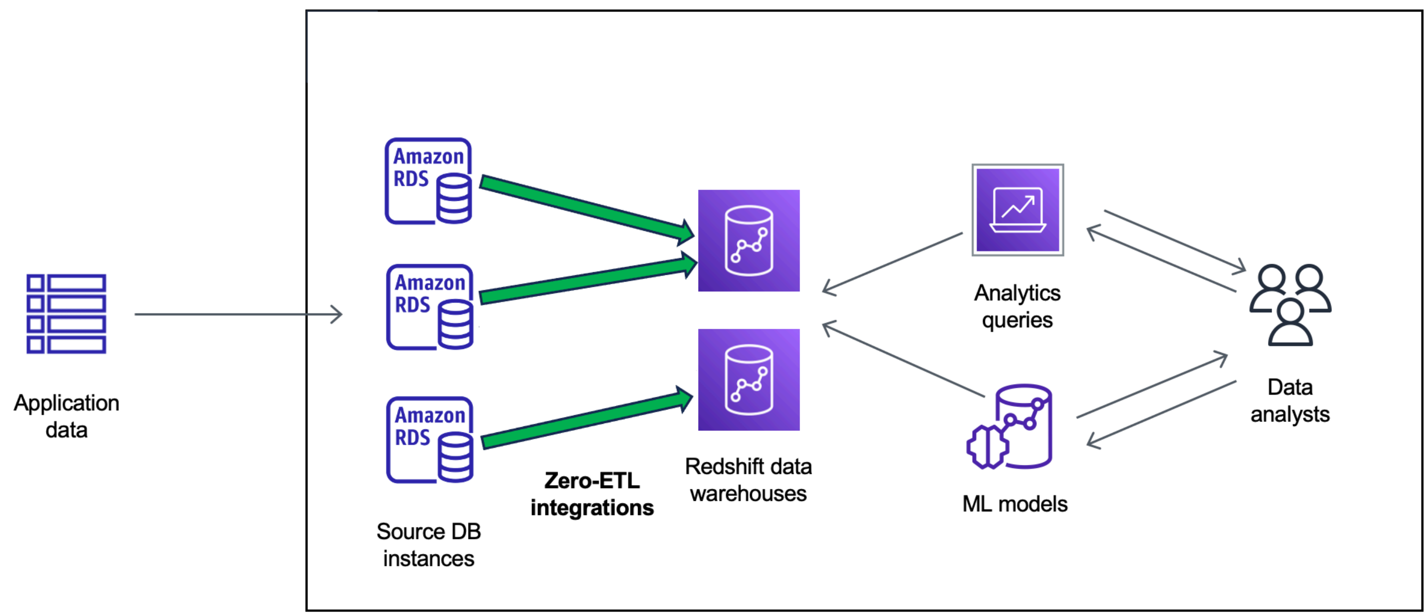
Amazon RDS 与 Amazon Redshift 的零 ETL 集成

Amazon RDS 与 Amazon Redshift 的零 ETL 集成支持使用 Amazon Redshift 对来自 RDS 的 PB 级事务数据进行近乎实时的分析和机器学习（ML）。这是一个完全托管式解决方案，用于在将事务数据写入 RDS 数据库后，使这些数据在 Amazon Redshift 中可用。提取、转换、加载（ETL）是将来自多个来源的数据合并到一个大型中央数据仓库的过程。

借助零 ETL 集成，您的 RDS 数据库中的数据在 Amazon Redshift 中随时可供使用。一旦这些数据进入 Amazon Redshift，您就可以使用 Amazon Redshift 的内置功能为分析、机器学习和人工智能工作负载提供支持，例如机器学习、实体化视图、数据共享、对多个数据存储和数据湖的联合访问以及与 Amazon SageMaker AI、QuickSight 及其他 Amazon Web Services 服务 的集成。

要创建零 ETL 集成，您需要将 RDS 数据库指定为源，并将 Amazon Redshift 数据仓库指定为目标。该集成会将数据从源数据库复制到目标数据仓库中。

下图阐明了此功能：



该集成还会监控数据管道的运行状况，并在可能的情况下从问题中恢复。您可以创建多个 RDS 数据库与单个 Amazon Redshift 命名空间的集成，从而使您能够获得跨多个应用程序的全面洞察。

主题

优势

RDS 与 Amazon Redshift 的零 ETL 集成有以下主要好处：

- 帮助您从多个数据来源获得整体见解。
- 无需构建和维护执行提取、转换、加载（ETL）操作的复杂数据管道。零 ETL 集成通过为您预调配和管理管道来消除构建和管理管道所带来的挑战。
- 减少运营负担和成本，让您专注于改进应用程序。
- 让您利用 Amazon Redshift 的分析和机器学习功能从事务数据和其他数据中获得见解，从而有效应对关键的、时间敏感的事件。

重要概念

在开始使用零 ETL 集成时，请考虑以下概念：

集成

> 完全管理的数据管道，可自动将事务数据和模式从 RDS 数据库复制到 Amazon Redshift 数据仓库。

源数据库

> 从中复制数据的 RDS 数据库。可以指定单可用区或多可用区数据库实例，或多可用区数据库集群。

目标数据仓库

将数据复制到的 Amazon Redshift 数据仓库。有两种类型的数据仓库：预调配集群数据仓库和无服务器数据仓库。预调配集群数据仓库是一个由称作节点的各种计算资源构成的集合，这些节点已整理到名为集群的组中。无服务器数据仓库由存储计算资源的工作组和存放数据库对象和用户的命名空间组成。两个数据仓库都运行 Amazon Redshift 引擎并包含一个或多个数据库。

多个源数据库可以写入同一个目标。

有关更多信息，请参阅《Amazon Redshift 开发人员指南》中的数据仓库系统架构。

限制

以下限制适用于 RDS 与 Amazon Redshift 的零 ETL 集成。

主题

一般限制

- 源数据库必须与目标 Amazon Redshift 数据仓库位于同一区域。
- 如果集群已有集成，则无法重命名数据库或其任何实例。
- 不能在相同的源数据库和目标数据库之间创建多个集成。
- 您无法删除已有集成的数据库。您必须先删除所有关联的集成。
- 如果您停止源数据库，则在恢复数据库之前，可能不会将最后几个事务复制到目标数据仓库。
- 如果源数据库实例已停止，则无法删除集成。
- 如果您的数据库是蓝绿部署的源，则在切换期间，蓝色和绿色环境无法具有现有的零 ETL 集成。您必须先删除集成，接着切换，然后重新创建集成。
- 如果源数据库正在积极创建另一个集成，则无法为其创建集成。
- 最初创建集成或重新同步表时，从源到目标的数据做种可能需要 20-25 分钟或更长时间，具体取决于源数据库的大小。这种延迟可能导致副本滞后延长。
- 某些数据类型不支持。有关更多信息，请参阅 the section called “数据类型差异”。

- 系统表、临时表和视图不会复制到 Amazon Redshift。

RDS for MySQL 限制

- 源数据库必须运行受支持的 RDS for MySQL 版本。有关受支持的版本的列表，请参阅 the section called "零 ETL 集成 "。
- 零 ETL 集成依赖于 MySQL 二进制日志（binlog）来捕获持续数据更改。请勿使用基于二进制日志的数据筛选，因为这可能会导致源数据库和目标数据库之间的数据不一致。
- 零 ETL 集成仅适用于配置为使用 InnoDB 存储引擎的数据库。
- 不支持带有预定义表更新的外键引用。具体而言，`CASCADE`、`SET NULL` 和 `SET DEFAULT` 操作不支持 `ON DELETE` 和 `ON UPDATE` 规则。尝试使用对另一个表的此类引用创建或更新表会使该表进入失败状态。
- `ALTER TABLE` 分区操作会导致您的表重新同步，以便将数据从 RDS 重新加载到 Amazon Redshift。该表在重新同步时将不可用于查询。有关更多信息，请参阅 the section called "我的一个或多个 Amazon Redshift 表需要重新同步"。

Amazon Redshift 限制

有关与零 ETL 集成相关的 Amazon Redshift 限制的列表，请参阅《Amazon Redshift 管理指南》中的将零 ETL 集成与 Amazon Redshift 结合使用时的注意事项。

限额

您的账户有以下关于 RDS 和 Amazon Redshift 的零 ETL 集成的限额。除非另行指定，否则每个限额将基于区域。

名称	默认值	描述
集成	100	Amazon Web Services 账户内的集成总数。
每个目标数据仓库的集成数	50	向单个目标 Amazon Redshift 数据仓库发送数据的集成数量。
每个源实例的集成数量	5	从单个源数据库实例发送数据的集成数量。

此外，Amazon Redshift 对每个数据库实例或集群节点中允许的表数量设置了某些限制。有关更多信息，请参阅《Amazon Redshift 管理指南》中的 Amazon Redshift 中的配额和限制。

支持的区域

RDS 与 Amazon Redshift 的零 ETL 集成在 Amazon Web Services 区域的子集中提供。有关受支持的区域的列表，请参阅the section called "零 ETL 集成 "。

开始使用 Amazon RDS 与 Amazon Redshift 的零 ETL 集成

在创建与 Amazon Redshift 的零 ETL 集成之前，请使用所需的参数和权限配置您的 RDS 数据库和 Amazon Redshift 数据仓库。在安装过程中，您将完成以下步骤：

1. 创建自定义数据库参数组。
2. 创建源数据库。
3. 创建目标 Amazon Redshift 数据仓库。

完成这些步骤后，请继续执行the section called "创建零 ETL 集成"。

> **Tip**
>
> 您可以在创建集成时让 RDS 为您完成这些设置步骤，而不必手动执行这些步骤。要立即开始创建集成，请参阅the section called "创建零 ETL 集成"。

步骤 1：创建自定义数据库参数组

Amazon RDS 与 Amazon Redshift 的零 ETL 集成需要为控制二进制日志记录（binlog）的数据库参数提供特定值。要配置二进制日志记录，必须先创建自定义数据库参数组，然后将其与源数据库关联。配置以下参数值。有关创建参数组的说明，请参阅the section called "数据库参数组"。我们建议您在同一个请求中配置所有参数值，以避免出现依赖项问题。

- `binlog_format=ROW`
- `binlog_row_image=full`

此外，请确保 `binlog_row_value_options` 参数未设置为 `PARTIAL_JSON`。如果源数据库为多可用区数据库集群，请确保 `binlog_transaction_compression` 参数未设置为 `ON`。

步骤 2：选择或创建源数据库

创建自定义数据库参数组后，选择或创建一个 RDS for MySQL 数据库。该数据库将成为向 Amazon Redshift 复制数据的源。有关创建单可用区或多可用区数据库实例的说明，请参阅 the section called “创建数据库实例”。有关创建多可用区数据库集群的说明，请参阅the section called “创建多可用区数据库集群”。

数据库必须运行受支持的数据库引擎版本。有关受支持的版本的列表，请参阅 the section called “零 ETL 集成 ”。

创建数据库时，在其他配置下，将默认的数据库参数组更改为您在上一步中创建的自定义参数组。

> **Note**
>
> 如果您在已创建数据库之后将参数组与数据库关联，则必须重启数据库以应用更改，然后才能创建零 ETL 集成。有关说明，请参阅the section called “重启数据库实例”或the section called “重启多可用区数据库集群”。

此外，请确保对数据库启用了自动备份。有关更多信息，请参阅 the section called “启用自动备份”。

步骤 3：创建目标 Amazon Redshift 数据仓库

创建源数据库后，必须在 Amazon Redshift 中创建和配置目标数据仓库。数据仓库必须满足以下要求：

- 使用 RA3 节点类型以及至少两个节点或 Redshift Serverless。
- 已加密（如果使用预置集群）。有关更多信息，请参阅 Amazon Redshift 数据库加密。

有关创建数据仓库的说明，请参阅预调配集群的创建集群或 Redshift Serverless 的创建带命名空间的工作组。

在数据仓库上启用区分大小写

要使集成成功，必须为数据仓库启用区分大小写参数（`enable_case_sensitive_identifier`）。默认情况下，所有预调配集群和 Redshift Serverless 工作组均禁用区分大小写。

要启用区分大小写，请根据您的数据仓库类型执行以下步骤：

- 预调配集群 – 要在预调配集群上启用区分大小写，请创建一个启用 `enable_case_sensitive_identifier` 参数的自定义参数组。然后，将该参数组与集群关联。有关说明，请参阅使用控制台管理参数组或使用 Amazon CLI 配置参数值。

> **Note**
>
> 将自定义参数组与集群关联后，请记得重启集群。

- 无服务器工作组 - 要在 Redshift Serverless 工作组上启用区分大小写，必须使用 Amazon CLI。Amazon Redshift 控制台目前不支持修改 Redshift Serverless 参数值。发送以下 update-workgroup 请求：

```
aws redshift-serverless update-workgroup \
  --workgroup-name target-workgroup \
  --config-parameters 
 parameterKey=enable_case_sensitive_identifier,parameterValue=true
```

  修改参数值后，无需重启工作组。

为数据仓库配置授权

创建数据仓库后，必须将源 RDS 数据库配置为授权的集成源。有关说明，请参阅为您的 Amazon Redshift 数据仓库配置授权。

使用 Amazon SDK 设置集成

您可以运行以下 Python 脚本来自动设置所需的资源，而不必手动设置每个资源。此代码示例使用适用于 Python (Boto3) 的 Amazon SDK 创建源 RDS for MySQL 数据库实例和目标 Amazon Redshift 数据仓库，其中每个都具有所需的参数值。然后，它会等待数据库变为可用后，再在它们之间创建零 ETL 集成。您可以根据需要设置的资源注释掉不同的函数。

要安装所需依赖项，请运行以下命令：

```
pip install boto3
pip install time
```

在脚本中，可以选择修改源组、目标组和参数组的名称。最后一个函数在设置资源后创建一个名为 `my-integration` 的集成。

Python 代码示例

```
import boto3
import time

# Build the client using the default credential configuration.
# You can use the CLI and run 'aws configure' to set access key, secret
# key, and default Region.

rds = boto3.client('rds')
redshift = boto3.client('redshift')
sts = boto3.client('sts')

source_db_name = 'my-source-db' # A name for the source database
source_param_group_name = 'my-source-param-group' # A name for the source parameter
 group
target_cluster_name = 'my-target-cluster' # A name for the target cluster
target_param_group_name = 'my-target-param-group' # A name for the target parameter
 group

def create_source_db(*args):
    """Creates a source RDS for MySQL DB instance"""

    response = rds.create_db_parameter_group(
        DBParameterGroupName=source_param_group_name,
        DBParameterGroupFamily='mysql8.0',
        Description='RDS for MySQL zero-ETL integrations'
    )
    print('Created source parameter group: ' + response['DBParameterGroup']
['DBParameterGroupName'])

    response = rds.modify_db_parameter_group(
        DBParameterGroupName=source_param_group_name,
        Parameters=[
            {
                'ParameterName': 'binlog_format',
                'ParameterValue': 'ROW',
                'ApplyMethod': 'pending-reboot'
            },
            {
                'ParameterName': 'binlog_row_image',
                'ParameterValue': 'full',
                'ApplyMethod': 'pending-reboot'
            }
```

```
        ]
    )
    print('Modified source parameter group: ' + response['DBParameterGroupName'])

    response = rds.create_db_instance(
        DBInstanceIdentifier=source_db_name,
        DBParameterGroupName=source_param_group_name,
        Engine='mysql',
        EngineVersion='8.0.32',
        DBName='mydb',
        DBInstanceClass='db.m5.large',
        AllocatedStorage=15,
        MasterUsername='username',
        MasterUserPassword='Password01**'
    )
    print('Creating source database: ' + response['DBInstance']
['DBInstanceIdentifier'])
    source_arn = (response['DBInstance']['DBInstanceArn'])
    create_target_cluster(target_cluster_name, source_arn, target_param_group_name)
    return(response)

def create_target_cluster(target_cluster_name, source_arn, target_param_group_name):
    """Creates a target Redshift cluster"""

    response = redshift.create_cluster_parameter_group(
        ParameterGroupName=target_param_group_name,
        ParameterGroupFamily='redshift-1.0',
        Description='RDS for MySQL zero-ETL integrations'
    )
    print('Created target parameter group: ' + response['ClusterParameterGroup']
['ParameterGroupName'])

    response = redshift.modify_cluster_parameter_group(
        ParameterGroupName=target_param_group_name,
        Parameters=[
            {
                'ParameterName': 'enable_case_sensitive_identifier',
                'ParameterValue': 'true'
            }
        ]
    )
    print('Modified target parameter group: ' + response['ParameterGroupName'])

    response = redshift.create_cluster(
```

```
        ClusterIdentifier=target_cluster_name,
        NodeType='ra3.4xlarge',
        NumberOfNodes=2,
        Encrypted=True,
        MasterUsername='username',
        MasterUserPassword='Password01**',
        ClusterParameterGroupName=target_param_group_name
    )
    print('Creating target cluster: ' + response['Cluster']['ClusterIdentifier'])

    # Retrieve the target cluster ARN
    response = redshift.describe_clusters(
        ClusterIdentifier=target_cluster_name
    )
    target_arn = response['Clusters'][0]['ClusterNamespaceArn']

    # Retrieve the current user's account ID
    response = sts.get_caller_identity()
    account_id = response['Account']

    # Create a resource policy granting access to source database and account ID
    response = redshift.put_resource_policy(
        ResourceArn=target_arn,
        Policy='''
        {
            \"Version\":\"2012-10-17\",
            \"Statement\":[
                {\"Effect\":\"Allow\",
                \"Principal\":{
                    \"Service\":\"redshift.amazonaws.com\"
                },
                \"Action\":[\"redshift:AuthorizeInboundIntegration\"],
                \"Condition\":{
                    \"StringEquals\":{
                        \"aws:SourceArn\":\"%s\"}
                    }
                },
                {\"Effect\":\"Allow\",
                \"Principal\":{
                    \"AWS\":\"arn:aws:iam::%s:root\"},
                \"Action\":\"redshift:CreateInboundIntegration\"}
            ]
        }
        ''' % (source_arn, account_id)
```

```
    )
    return(response)

def wait_for_db_availability(*args):
    """Waits for both databases to be available"""

    print('Waiting for source and target to be available...')

    response = rds.describe_db_instances(
        DBInstanceIdentifier=source_db_name
    )
    source_status = response['DBInstances'][0]['DBInstanceStatus']
    source_arn = response['DBInstances'][0]['DBInstanceArn']

    response = redshift.describe_clusters(
        ClusterIdentifier=target_cluster_name
    )
    target_status = response['Clusters'][0]['ClusterStatus']
    target_arn = response['Clusters'][0]['ClusterNamespaceArn']

    # Every 60 seconds, check whether the databases are available
    if source_status != 'available' or target_status != 'available':
        time.sleep(60)
        response = wait_for_db_availability(
            source_db_name, target_cluster_name)
    else:
        print('Databases available. Ready to create zero-ETL integration.')
        create_integration(source_arn, target_arn)
        return

def create_integration(source_arn, target_arn):
    """Creates a zero-ETL integration using the source and target databases"""

    response = rds.create_integration(
        SourceArn=source_arn,
        TargetArn=target_arn,
        IntegrationName='my-integration'
    )
    print('Creating integration: ' + response['IntegrationName'])

def main():
    """main function"""
    create_source_db(source_db_name, source_param_group_name)
    wait_for_db_availability(source_db_name, target_cluster_name)
```

```
if __name__ == "__main__":
    main()
```

后续步骤

借助源 RDS 数据库和 Amazon Redshift 目标数据仓库，您可以创建零 ETL 集成并复制数据。有关说明，请参阅the section called “创建零 ETL 集成”。

创建 Amazon RDS 与 Amazon Redshift 的零 ETL 集成

在创建 Amazon RDS 零 ETL 集成时，需要指定源 RDS 数据库和目标 Amazon Redshift 数据仓库。您还可以自定义加密设置和添加标签。Amazon RDS 在源数据库与其目标之间创建集成。集成激活后，您插入到源数据库中的任何数据都将复制到配置的 Amazon Redshift 目标中。

主题

- 先决条件
- 所需的权限
- 创建零 ETL 集成
- 使用客户自主管理型密钥加密集成
- 后续步骤

先决条件

在创建零 ETL 集成之前，必须创建源数据库和目标 Amazon Redshift 数据仓库。您还必须通过将数据库添加为授权集成源来允许复制到数据仓库。

有关完成其中每个步骤的说明，请参阅the section called “开始使用零 ETL 集成”。

所需的权限

创建零 ETL 集成需要具有某些 IAM 权限。至少您需要具有执行以下操作的权限：

- 为源 RDS 数据库创建零 ETL 集成。
- 查看和删除所有零 ETL 集成。

- 在目标数据仓库中创建入站集成。如果同一个账户拥有 Amazon Redshift 数据仓库并且该账户是该数据仓库的授权主体，则不需要此权限。有关添加授权主体的信息，请参阅为您的 Amazon Redshift 数据仓库配置授权。

以下示例策略演示了创建和管理集成所需的最低权限。如果您的用户或角色具有更广泛的权限（例如 `AdministratorAccess` 托管式策略），则可能不需要这些确切的权限。

Note

Redshift Amazon 资源名称（ARN）采用以下格式。请注意，在无服务器命名空间 UUID 之前使用了正斜杠（(/）而不是冒号（:）。

- 预调配集群 – `arn:aws:`**`redshift`**`:{region}:{account-id}:namespace:`*`namespace-uuid`*
- 无服务器 – `arn:aws:`**`redshift-serverless`**`:{region}:{account-id}:namespace`**`/`***`namespace-uuid`*

示例策略

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "rds:CreateIntegration"
        ],
        "Resource": [
            "arn:aws:rds:{region}:{account-id}:db:source-db",
            "arn:aws:rds:{region}:{account-id}:integration:*"
        ]
    },
    {
        "Effect": "Allow",
        "Action": [
            "rds:DescribeIntegrations"
        ],
        "Resource": ["*"]
    },
    {
```

```
            "Effect": "Allow",
            "Action": [
                "rds:DeleteIntegration",
                "rds:ModifyIntegration"
            ],
            "Resource": [
                "arn:aws:rds:{region}:{account-id}:integration:*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "redshift:CreateInboundIntegration"
            ],
            "Resource": [
                "arn:aws:redshift:{region}:{account-id}:namespace:namespace-uuid"
            ]
        }]
}
```

在不同的账户中选择目标数据仓库

如果您计划指定位于另一个 Amazon Web Services 账户中的目标 Amazon Redshift 数据仓库，则必须创建一个角色，以允许当前账户中的用户访问目标账户中的资源。有关更多信息，请参阅在您拥有的其他 Amazon Web Services 账户中向 IAM 用户提供访问权限。

该角色必须具有以下权限，这些权限允许用户查看目标账户中可用的 Amazon Redshift 预调配集群和 Redshift Serverless 命名空间。

必需的权限和信任策略

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "redshift:DescribeClusters",
            "redshift-serverless:ListNamespaces"
         ],
         "Resource":[
            "*"
         ]
```

```
        }
    ]
}
```

该角色必须具有以下信任策略，该策略指定目标账户 ID。

```
{
    "Version":"2012-10-17",
    "Statement":[
        {
            "Effect":"Allow",
            "Principal":{
                "AWS": "arn:aws:iam::{external-account-id}:root"
            },
            "Action":"sts:AssumeRole"
        }
    ]
}
```

有关创建角色的说明，请参阅使用自定义信任策略创建角色。

创建零 ETL 集成

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 创建零 ETL 集成。

默认情况下，RDS for MySQL 会立即清除二进制日志文件。由于零 ETL 集成依赖二进制日志将数据从源复制到目标，因此源数据库的保留期必须至少为一小时。创建集成后，Amazon RDS 会立即检查所选源数据库的二进制日志文件保留期。如果当前值为 0 小时，则 Amazon RDS 会自动将其更改为 1 小时。否则，该值将保持不变。

RDS 控制台

创建零 ETL 集成

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在左侧导航窗格中，选择零 ETL 集成。
3. 选择创建零 ETL 集成。
4. 在集成标识符中，输入集成的名称。该名称可包含最多 63 个字母数字字符，并且可以包含连字符。

5. 选择下一步。
6. 对于源，选择数据将源自其中的 RDS 数据库。

> **Note**
>
> 如果数据库参数配置不正确，RDS 会通知您。如果您收到此消息，可以选择为我修复，也可以手动配置它们。有关手动修复它们的说明，请参阅the section called “步骤 1：创建自定义数据库参数组”。
>
> 修改数据库参数需要重启。在创建集成之前，必须完成重启，并且必须成功地将新的参数值应用于数据库。

7. 成功配置源数据库后，选择下一步。
8. 对于目标，执行以下操作：
   1. （可选）要为 Amazon Redshift 目标使用不同的 Amazon Web Services 账户，请选择指定其他账户。然后，输入有权显示您数据仓库的 IAM 角色的 ARN。有关创建 IAM 角色的说明，请参阅the section called “在不同的账户中选择目标数据仓库”。
   2. 对于 Amazon Redshift 数据仓库，选择从源数据库中复制的数据的目标。您可以选择预调配的 Amazon Redshift 集群或 Redshift Serverless 命名空间作为目标。

> **Note**
>
> 如果指定数据仓库的资源策略或区分大小写设置配置不正确，RDS 会通知您。如果您收到此消息，可以选择为我修复，也可以手动配置它们。有关手动修复这些问题的说明，请参阅《Amazon Redshift 管理指南》中的为您的数据仓库开启区分大小写和为您的数据仓库配置授权。
>
> 修改预调配 Redshift 集群的区分大小写需要重启。在创建集成之前，必须完成重启，并且必须成功地将新的参数值应用于集群。
>
> 如果您选择的源和目标位于不同的 Amazon Web Services 账户，则 Amazon RDS 无法为您修复这些设置。您必须导航到另一个账户，然后在 Amazon Redshift 中手动修复这些问题。

9. 正确配置目标数据仓库后，选择下一步。
10. （可选）对于标签，向集成添加一个或多个标签。有关更多信息，请参阅 the section called “为 RDS 资源添加标签”。

11. 对于加密，请指定您希望如何加密集成。默认情况下，RDS 会加密所有与 Amazon 拥有的密钥 的集成。要改为选择客户自主管理型密钥，请启用自定义加密设置并选择用于加密的 KMS 密钥。有关更多信息，请参阅 the section called “加密 Amazon RDS 资源”。

    （可选）添加加密上下文。有关更多信息，请参阅 Amazon Key Management Service 开发人员指南中的加密内容。

    > **Note**
    >
    > 除了您添加的任何加密上下文对外，Amazon RDS 还会添加以下加密上下文对：
    >
    > - `aws:redshift:integration:arn` - `IntegrationArn`
    > - `aws:servicename:id` - `Redshift`
    >
    > 这会将您可以添加的加密上下文对总数从 8 减少到 6，并增加授予约束条件的总字符限制。有关更多信息，请参阅《Amazon Key Management Service 开发人员指南》中的 Using grant constraints。

12. 选择下一步。
13. 查看您的集成设置并选择创建零 ETL 集成。

    如果创建失败，请参阅the section called “我无法创建零 ETL 集成”以了解故障排除步骤。

集成在创建时状态为 `Creating`，而目标 Amazon Redshift 数据仓库的状态为 `Modifying`。在此期间，您无法查询数据仓库或对其进行任何配置更改。

成功创建集成后，集成和目标 Amazon Redshift 数据仓库的状态都更改为 `Active`。

Amazon CLI

要使用 Amazon CLI 创建零 ETL 集成，请使用带有以下选项的 create-integration 命令：

- `--integration-name` – 指定集成的名称。
- `--source-arn` – 指定将作为集成源的 RDS 数据库的 ARN。
- `--target-arn` – 指定将作为集成目标的 Amazon Redshift 数据仓库的 ARN。

Example

对于 Linux、macOS 或 Unix：

```
aws rds create-integration \
    --integration-name my-integration \
    --source-arn arn:aws:rds:{region}:{account-id}:my-db \
    --target-arn arn:aws:redshift:{region}:{account-id}:namespace:namespace-uuid
```

对于 Windows：

```
aws rds create-integration ^
    --integration-name my-integration ^
    --source-arn arn:aws:rds:{region}:{account-id}:my-db ^
    --target-arn arn:aws:redshift:{region}:{account-id}:namespace:namespace-uuid
```

RDS API

要使用 Amazon RDS API 创建零 ETL 集成，请结合以下参数使用 [CreateIntegration] 操作：

- `IntegrationName` – 指定集成的名称。
- `SourceArn` – 指定将作为集成源的 RDS 数据库的 ARN。
- `TargetArn` – 指定将作为集成目标的 Amazon Redshift 数据仓库的 ARN。

使用客户自主管理型密钥加密集成

如果您指定自定义 KMS 密钥而不是创建集成时的 Amazon 拥有的密钥，则密钥策略必须为 Amazon Redshift 服务主体提供对 `CreateGrant` 操作的访问权限。此外，它必须允许当前用户执行 `DescribeKey` 和 `CreateGrant` 操作。

以下示例策略演示了如何提供密钥策略中所需的权限。它包括用于进一步缩小权限范围的上下文键。

示例密钥策略

```
{
    "Version": "2012-10-17",
    "Id": "Key policy",
    "Statement": [
        {
            "Sid": "Enables IAM user permissions",
            "Effect": "Allow",
```

```
            "Principal": {
                "AWS": "arn:aws:iam::{account-ID}:root"
            },
            "Action": "kms:*",
            "Resource": "*"
        },
        {
            "Sid": "Allows the Redshift service principal to add a grant to a KMS key",
            "Effect": "Allow",
            "Principal": {
                "Service": "redshift.amazonaws.com"
            },
            "Action": "kms:CreateGrant",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "kms:EncryptionContext:{context-key}":"{context-value}"
                },
                "ForAllValues:StringEquals": {
                    "kms:GrantOperations": [
                        "Decrypt",
                        "GenerateDataKey",
                        "CreateGrant"
                    ]
                }
            }
        },
        {
            "Sid": "Allows the current user or role to add a grant to a KMS key",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::{account-ID}:role/{role-name}"
            },
            "Action": "kms:CreateGrant",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "kms:EncryptionContext:{context-key}":"{context-value}",
                    "kms:ViaService": "rds.us-east-1.amazonaws.com"
                },
                "ForAllValues:StringEquals": {
                    "kms:GrantOperations": [
                        "Decrypt",
                        "GenerateDataKey",
```

```
                        "CreateGrant"
                    ]
                }
            }
        },
        {
            "Sid": "Allows the current uer or role to retrieve information about a KMS
 key",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::{account-ID}:role/{role-name}"
            },
            "Action": "kms:DescribeKey",
            "Resource": "*"
        }
    ]
}
```

有关更多信息，请参阅《Amazon Key Management Service 开发人员指南》中的创建密钥策略。

后续步骤

成功创建零 ETL 集成后，您必须在目标 Amazon Redshift 集群或工作组中创建目标数据库。然后，您可以开始向源 RDS 数据库中添加数据，并在 Amazon Redshift 中对其进行查询。有关说明，请参阅在 Amazon Redshift 中创建目标数据库。

Amazon RDS 与 Amazon Redshift 零 ETL 集成的数据筛选

Amazon RDS 零 ETL 集成支持数据筛选，这使您可以控制哪些数据从源 Amazon RDS 数据库复制到目标 Amazon Redshift 数据仓库。您可以应用一个或多个筛选条件来有选择地包含或排除特定的表，而不是复制整个数据库。这可以通过确保只传输相关数据，来协助您优化存储和查询性能。目前，筛选仅限于数据库级和表级。不支持列级和行级筛选。

当您要执行以下操作时，数据筛选可能很有用：

- 联接来自两个或更多不同源数据库的某些表，而您不需要来自任一数据库的完整数据。
- 仅使用表的子集而不是整个数据库实例集来执行分析，从而节省成本。
- 从某些表中筛选掉敏感信息，例如电话号码、地址或信用卡详细信息。

您可以使用 Amazon Web Services Management Console、Amazon Command Line Interface（Amazon CLI）或 Amazon RDS API 向零 ETL 集成添加数据筛选条件。

如果集成将预置 Amazon Redshift 集群作为其目标，则该集群必须使用补丁 180 或更高版本，才能使用数据筛选。

主题

数据筛选条件的格式

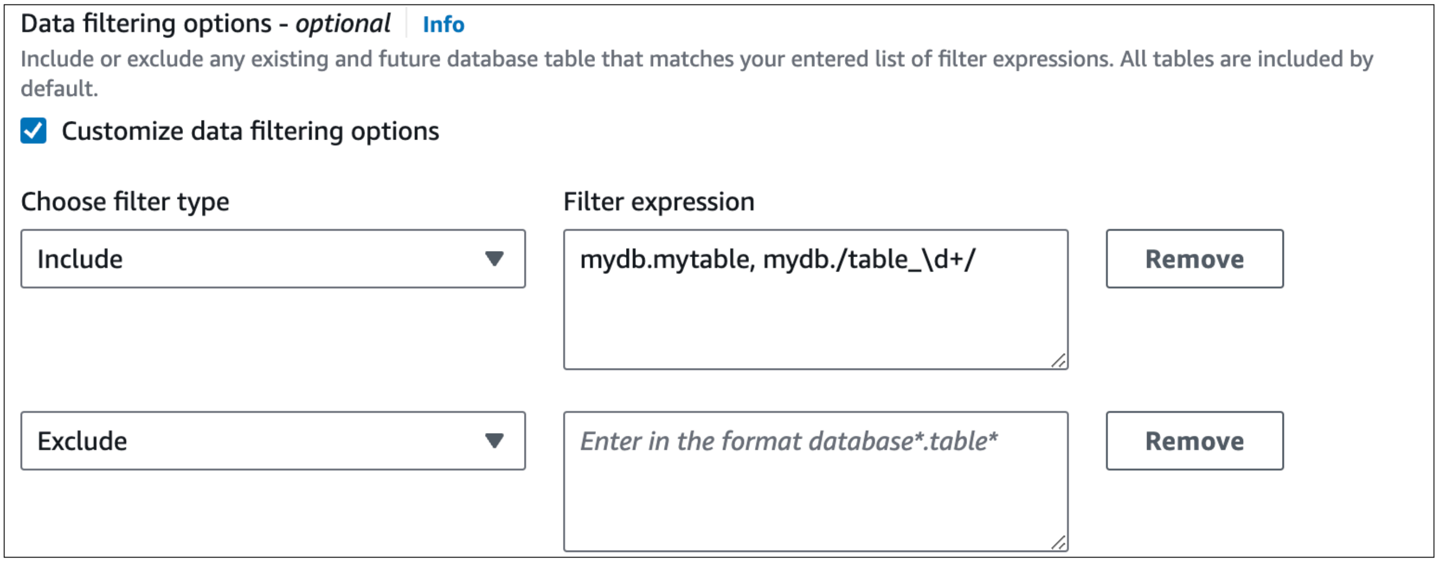
您可以为单个集成定义多个筛选条件。每个筛选条件要么包含、要么排除任何与筛选表达式中的模式之一匹配的现有和将来的数据库表。Amazon RDS 零 ETL 集成使用 Maxwell 筛选条件语法进行数据筛选。

每个筛选条件都包含以下元素：

元素	描述
筛选条件类型	`Include` 筛选条件类型包括与筛选表达式中的模式之一匹配的所有表。`Exclude` 筛选条件类型排除与模式之一匹配的所有表。
筛选表达式	逗号分隔的模式列表。表达式必须使用 Maxwell 筛选条件语法。
模式	一种筛选条件模式，格式为 *`database`*`.`*`table`*。您可以指定文字名称，也可以定义正则表达式。 您不能包含列级别筛选条件或黑名单。

元素	描述
	单个集成最多可以有总共 99 个模式。在控制台中，可以在单个筛选表达式中输入模式，也可以将它们分散在多个表达式中。单个模式的长度不能超过 256 个字符。

下图显示了控制台中 数据筛选条件的结构：



> **Important**
>
> 请勿在筛选条件模式中包含个人身份信息、机密信息或敏感信息。

Amazon CLI 中的数据筛选条件

使用 Amazon CLI 添加数据筛选条件时，语法与使用控制台略有不同。必须为每种模式分别指定筛选条件类型（`Include` 或 `Exclude`），因此不能将多个模式分组到一个筛选条件类型下。

例如，在控制台中，可以在单个 `Include` 语句下对以下逗号分隔的模式进行分组：

```
mydb.mytable, mydb./table_\d+/
```

但是，使用 Amazon CLI 时，相同的数据筛选条件必须采用以下格式：

```
'include: mydb.mytable, include: mydb./table_\d+/'
```

筛选条件逻辑

如果您在集成中未指定任何数据筛选条件，Amazon RDS 会采用默认筛选条件 `include:*.*`，这会将所有表复制到目标数据仓库。但是，如果您至少添加一个筛选条件，则默认逻辑会切换为 `exclude:*.*`，默认情况下这会排除所有表。这使您可以显式定义要在复制中包含哪些数据库和表。

例如，如果您定义以下筛选条件：

```
'include: db.table1, include: db.table2'
```

Amazon RDS 按以下方式评估筛选条件：

```
'exclude:*.*, include: db.table1, include: db.table2'
```

因此，Amazon RDS 仅将来自名为 db 的数据库的 `table1` 和 `table2` 复制到目标数据仓库。

筛选条件优先级

Amazon RDS 按您指定的顺序评估数据筛选条件。在 Amazon Web Services Management Console 中，它将从左到右、从上到下处理筛选表达式。第二个筛选条件或第一个筛选条件之后的单个模式可以覆盖它。

例如，如果第一个筛选条件是 `Include books.stephenking`，则它只包括 books 数据库中的 `stephenking` 表。但是，如果您添加第二个筛选条件 `Exclude books.*`，则它会覆盖第一个筛选条件。这可以防止 books 索引中的任何表复制到 Amazon Redshift。

当您指定至少一个筛选条件时，默认情况下，逻辑以采用 `exclude:*.*` 开始，这会自动将所有表排除 在复制范围之外。作为最佳实践，请按照从最广泛到最具体的顺序定义筛选条件。首先使用一个或多个 `Include` 语句来指定要复制的数据，然后添加 `Exclude` 筛选条件来有选择性地移除某些表。

同样的原则也适用于使用 Amazon CLI 定义的筛选条件。Amazon RDS 按您指定筛选条件模式的顺序评估这些模式，因此一个模式可能会覆盖您指定的在它之前的模式。

示例

以下示例演示了如何将数据筛选用于 零 ETL 集成：

- 包括所有数据库和所有表：

```
'include: *.*'
```

- 包括 books 数据库中的所有表：

```
'include: books.*'
```

- 排除任何名为 mystery 的表：

```
'include: *.*, exclude: *.mystery'
```

- 包括 books 数据库中的两个特定表：

```
'include: books.stephen_king, include: books.carolyn_keene'
```

- 包括 books 数据库中的所有表，但那些包含子字符串 mystery 的表除外：

```
'include: books.*, exclude: books./.*mystery.*/'
```

- 包括 books 数据库中的所有表，但那些以 mystery 开头的表除外：

```
'include: books.*, exclude: books./mystery.*/'
```

- 包括 books 数据库中的所有表，但那些以 mystery 结束的表除外：

```
'include: books.*, exclude: books./.*mystery/'
```

- 包括 books 数据库中的所有以 table_ 开头的表，但名为 table_stephen_king 的表除外。例如，将复制 table_movies 或 table_books，但不复制 table_stephen_king。

```
'include: books./table_.*/, exclude: books.table_stephen_king'
```

向集成添加数据筛选条件

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 Amazon RDS API 配置数据筛选。

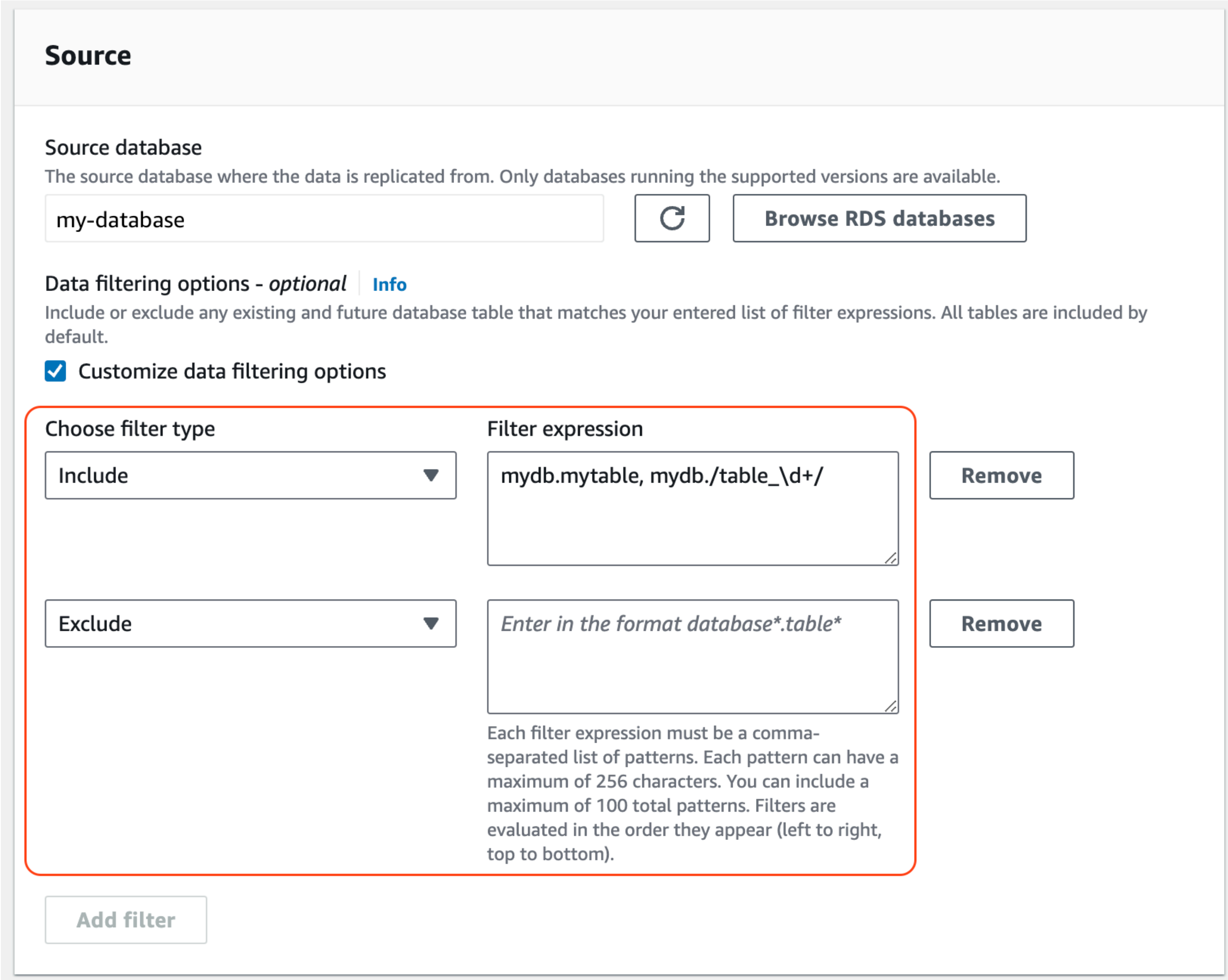
> **Important**
>
> 如果您在创建集成后添加筛选条件，Amazon RDS 会将其视为似乎始终存在。它会移除目标 Amazon Redshift 数据仓库中不符合新筛选条件的任何数据，并重新同步所有受影响的表。

RDS 控制台

将数据筛选条件添加到零 ETL 集成中

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择零 ETL 集成。选择要向其添加数据筛选条件的集成，然后选择修改。
3. 在源下，添加一个或多个 `Include` 和 `Exclude` 语句。

   下图显示了 MySQL 集成的数据筛选条件的示例：



4. 如果您对更改感到满意，请选择继续和保存更改。

Amazon CLI

要使用 Amazon CLI 向零 ETL 集成添加数据筛选条件，请调用用 modify-integration 命令。除了集成标识符外，还要使用逗号分隔的 `Include` 和 `Exclude` Maxwell 筛选条件列表来指定 `--data-filter` 参数。

Example

以下示例向 `my-integration` 添加筛选条件模式。

对于 Linux、macOS 或 Unix：

```
aws rds modify-integration \
```

```
    --integration-identifier my-integration \
    --data-filter 'include: foodb.*, exclude: foodb.tbl, exclude: foodb./table_\d+/'
```

对于 Windows：

```
aws rds modify-integration ^
    --integration-identifier my-integration ^
    --data-filter 'include: foodb.*, exclude: foodb.tbl, exclude: foodb./table_\d+/'
```

RDS API

要使用 RDS API 修改零 ETL 集成，请调用 ModifyIntegration 操作。指定集成标识符，并提供逗号分隔的筛选条件模式列表。

从集成中移除数据筛选条件

当您从集成中移除数据筛选条件时，Amazon RDS 会重新评估剩余的筛选条件，就好像移除的筛选条件从未存在过一样。然后，它将所有以前排除但现在符合条件的数据复制到目标 Amazon Redshift 数据仓库。这会触发所有受影响的表重新同步。

向源 RDS 数据库中添加数据并在 Amazon Redshift 中对其进行查询

要创建将数据从 Amazon RDS 复制到 Amazon Redshift 的零 ETL 集成，您必须在 Amazon Redshift 中创建目标数据库。

首先，连接到您的 Amazon Redshift 集群或工作组，并创建一个引用您的集成标识符的数据库。然后，您可以向 RDS 数据库中添加数据，并在 Amazon Redshift 中查看其复制的内容。

主题

- 在 Amazon Redshift 中创建目标数据库
- 向源数据库中添加数据
- 在 Amazon Redshift 中查询您的 Amazon RDS 数据
- RDS 和 Amazon Redshift 数据库之间的数据类型差异

在 Amazon Redshift 中创建目标数据库

在开始将数据复制到 Amazon Redshift 中之前，创建集成后，您必须在目标数据仓库中创建一个目标数据库。此目标数据库必须包含对集成标识符的引用。您可以使用 Amazon Redshift 控制台或查询编辑器 v2 来创建数据库。

有关创建目标数据库的说明，请参阅在 Amazon Redshift 中创建目标数据库。

向源数据库中添加数据

在配置集成后，您可以将一些数据添加到您希望复制到 Amazon Redshift 数据仓库的 RDS 数据库中。

> **Note**
>
> Amazon RDS 和 Amazon Redshift 中的数据类型存在差异。有关数据类型映射的表，请参阅the section called “数据类型差异”。

首先，使用您选择的 MySQL 客户端连接到源数据库。有关说明，请参阅 the section called “连接到运行 MySQL 的数据库实例”。

然后，创建一个表并插入一行示例数据。

> **Important**
>
> 确保该表有主键。否则，它无法复制到目标数据仓库。

以下示例使用 MySQL Workbench 实用程序。

```
CREATE DATABASE my_db;

USE my_db;

CREATE TABLE books_table (ID int NOT NULL, Title VARCHAR(50) NOT NULL, Author
 VARCHAR(50) NOT NULL,
Copyright INT NOT NULL, Genre VARCHAR(50) NOT NULL, PRIMARY KEY (ID));
```

```
INSERT INTO books_table VALUES (1, 'The Shining', 'Stephen King', 1977, 'Supernatural
 fiction');
```

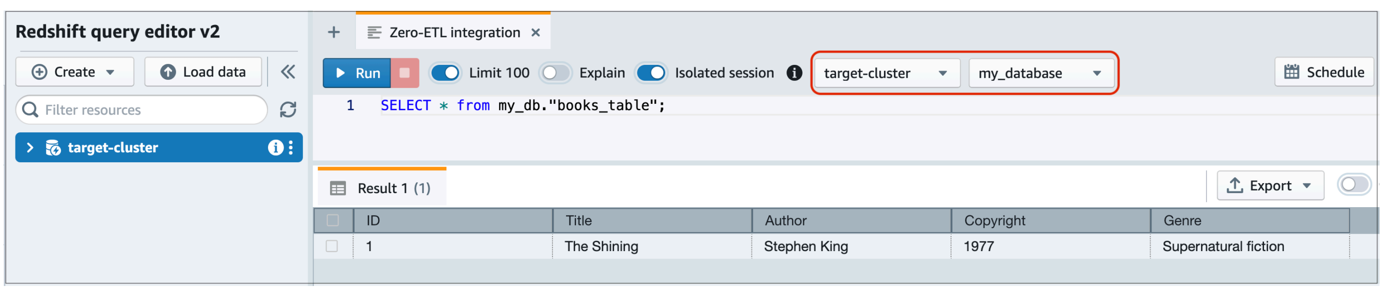
在 Amazon Redshift 中查询您的 Amazon RDS 数据

将数据添加到 RDS 数据库后，它会复制到 Amazon Redshift 中并准备好以供查询。

查询复制的数据

1. 导航到 Amazon Redshift 控制台，然后从左侧导航窗格中选择查询编辑器 v2。
2. 连接到您的集群或工作组，然后从下拉菜单中选择您通过集成创建的目标数据库（本示例中为 destination_database）。有关创建目标数据库的说明，请参阅[在 Amazon Redshift 中创建目标数据库](#)。
3. 使用 SELECT 语句来查询您的数据。在本例中，您可以运行以下命令，从您在源 RDS 数据库中创建的表中选择所有数据：

```
SELECT * from my_db."books_table";
```



- *my_db* 是 RDS 数据库模式名称。
- *books_table* 是 RDS 表名称。

也可以使用命令行客户端查询数据。例如：

```
destination_database=# select * from my_db."books_table";

 ID |       Title |          Author |   Copyright |                   Genre |  txn_seq |
 txn_id
----+-------------+----------------+-------------+-------------------------+----------
+--------+
```

```
  1 | The Shining |  Stephen King |         1977 |   Supernatural fiction |         2 |
12192
```

Note

为了区分大小写，请对架构、表和列名使用双引号（""）。有关更多信息，请参阅 enable_case_sensitive_identifier。

RDS 和 Amazon Redshift 数据库之间的数据类型差异

下表显示 RDS for MySQL 数据类型与相应 Amazon Redshift 数据类型的映射。Amazon RDS 目前仅支持将这些数据类型用于零 ETL 集成。

如果您的源数据库中的表包含不受支持的数据类型，则该表将不同步并且 Amazon Redshift 目标无法使用该表。从源到目标的流式传输仍在继续，但数据类型不受支持的表不可用。要修复该表并使其在 Amazon Redshift 中可用，您必须手动恢复重大更改，然后通过运行 `ALTER DATABASE...INTEGRATION REFRESH` 来刷新集成。

RDS for MySQL

RDS for MySQL 数据类型	Amazon Redshift 数据类型	描述	限制
INT	INTEGER	有符号的四字节整数	无
SMALLINT	SMALLINT	有符号的二字节整数	无
TINYINT	SMALLINT	有符号的二字节整数	无
MEDIUMINT	INTEGER	有符号的四字节整数	无
BIGINT	BIGINT	有符号的八字节整数	无

RDS for MySQL 数据类型	Amazon Redshift 数据类型	描述	限制
INT UNSIGNED	BIGINT	有符号的八字节整数	无
TINYINT UNSIGNED	SMALLINT	有符号的二字节整数	无
MEDIUMINT UNSIGNED	INTEGER	有符号的四字节整数	无
BIGINT UNSIGNED	DECIMAL(20,0)	可选精度的精确数字	无
DECIMAL(p,s) = NUMERIC(p ,s)	DECIMAL (p,s)	可选精度的精确数字	不支持精度大于 38 和比例大于 37
DECIMAL(p,s) UNSIGNED = NUMERIC(p,s) UNSIGNED	DECIMAL (p,s)	可选精度的精确数字	不支持精度大于 38 和比例大于 37
FLOAT4/REAL	REAL	单精度浮点数	无
FLOAT4/REAL UNSIGNED	REAL	单精度浮点数	无
DOUBLE/REAL/FLOAT8	DOUBLE PRECISION	双精度浮点数	无
DOUBLE/REAL/FLOAT8 UNSIGNED	DOUBLE PRECISION	双精度浮点数	无
BIT(n)	VARBYTE(8)	长度可变的二进制值	无
BINARY(n)	VARBYTE(n)	长度可变的二进制值	无
VARBINARY (n)	VARBYTE(n)	长度可变的二进制值	无

RDS for MySQL 数据类型	Amazon Redshift 数据类型	描述	限制
CHAR(n)	VARCHAR (n)	长度可变的字符串值	无
VARCHAR (n)	VARCHAR (n)	长度可变的字符串值	无
TEXT	VARCHAR(65535)	长度可变、最多 65535 个字符的字符串值	无
TINYTEXT	VARCHAR(255)	长度可变、最多 255 个字符的字符串值	无
MEDIUMTEXT	VARCHAR(65535)	长度可变、最多 65535 个字符的字符串值	无
LONGTEXT	VARCHAR(65535)	长度可变、最多 65535 个字符的字符串值	无
ENUM	VARCHAR(1020)	长度可变、最多 1020 个字符的字符串值	无
SET	VARCHAR(1020)	长度可变、最多 1020 个字符的字符串值	无
DATE	DATE	日历日期（年、月、日）	无
DATETIME	TIMESTAMP	日期和时间（没有时区）	无

RDS for MySQL 数据类型	Amazon Redshift 数据类型	描述	限制
TIMESTAMP(p)	TIMESTAMP	日期和时间（没有时区）	无
TIME	VARCHAR(18)	长度可变、最多 18 个字符的字符串值	无
YEAR	VARCHAR(4)	长度可变、最多 4 个字符的字符串值	无
JSON	SUPER	作为值的半结构化数据或文档	无

查看和监控 Amazon RDS 与 Amazon Redshift 的零 ETL 集成

您可以查看 Amazon RDS 零 ETL 集成的详细信息，以查看其配置信息和当前状态。您还可以通过在 Amazon Redshift 中查询特定的系统视图来监控集成的状态。此外，Amazon Redshift 向 Amazon CloudWatch 发布某些与集成相关的指标，您可以在 Amazon Redshift 控制台中查看这些指标。

主题

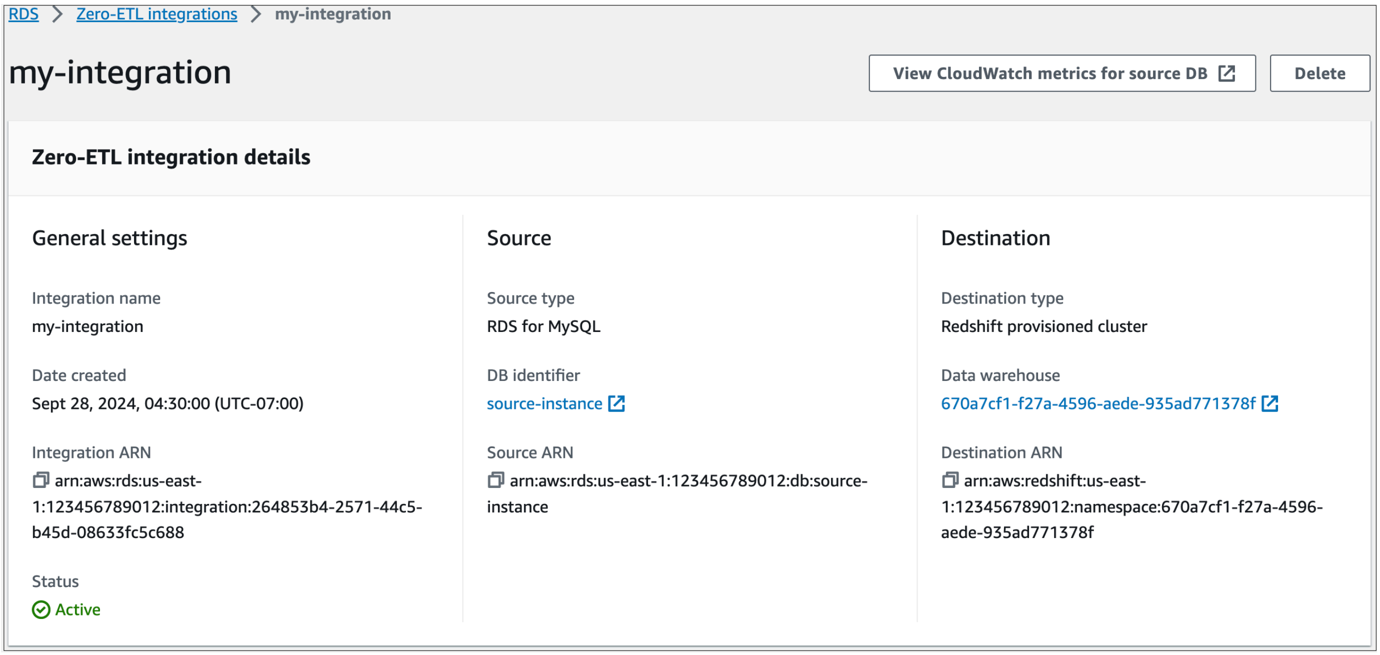
查看集成

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 查看 Amazon RDS 与 Amazon Redshift 的零 ETL 集成。

控制台

查看零 ETL 集成的详细信息

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 从左侧导航窗格中，选择零 ETL 集成。
3. 选择一个集成以查看其详细信息，例如其源数据库和目标数据仓库。



集成可以具有以下状态：

- `Creating` – 正在创建集成。
- `Active` – 集成正在将事务数据发送到目标数据仓库。
- `Syncing` – 集成遇到了可恢复的错误，正在重新设置数据种子。受影响的表在完成重新同步之前无法在 Amazon Redshift 中进行查询。
- `Needs attention` – 集成遇到了需要手动干预才能解决的事件或错误。要修复问题，请按照集成详细信息页面上错误消息中的说明修复问题。
- `Failed` – 集成遇到了无法恢复的事件或错误，无法修复。您必须删除并重新创建集成。
- `Deleting` – 正在删除集成。

Amazon CLI

要使用 Amazon CLI 查看当前账户中的所有 零 ETL 集成，请使用 describe-integrations 命令并指定 `--integration-identifier` 选项。

Example

对于 Linux、macOS 或 Unix：

```
aws rds describe-integrations \
    --integration-identifier ee605691-6c47-48e8-8622-83f99b1af374
```

对于 Windows：

```
aws rds describe-integrations ^
    --integration-identifier ee605691-6c47-48e8-8622-83f99b1af374
```

RDS API

要使用 Amazon RDS API 查看零 ETL 集成，请结合 `IntegrationIdentifier` 参数使用 `DescribeIntegrations` 操作。

使用系统表监控集成

Amazon Redshift 具有包含系统运行方式相关信息的系统表和视图。您可以像查询任何其他数据库表那样查询这些系统表和视图。有关 Amazon Redshift 中的系统表和视图的更多信息，请参阅《Amazon Redshift 数据库开发人员指南》中的系统表和视图参考。

您可以查询以下系统视图和表，以获取有关 与 Amazon Redshift 的零 ETL 集成的信息：

- SVV_INTEGRATION – 为您的集成提供配置详细信息。
- SVV_INTEGRATION_TABLE_STATE – 描述集成中每个表的状态。
- SYS_INTEGRATION_TABLE_STATE_CHANGE – 显示集成的表状态更改日志。
- SYS_INTEGRATION_ACTIVITY – 提供有关已完成的集成运行的信息。

所有集成相关的 Amazon CloudWatch 指标均源自 Amazon Redshift。有关更多信息，请参阅《Amazon Redshift 管理指南》中的零 ETL 集成的指标。目前，Amazon RDS 不向 Amazon CloudWatch 发布任何与集成相关的指标。

监控与 Amazon EventBridge 的集成

Amazon Redshift 会将集成相关的事件发送到 Amazon EventBridge。有关事件及其相应事件 ID 的列表，请参阅《Amazon Redshift 管理指南》中的与 Amazon EventBridge 的零 ETL 集成事件通知。

修改 Amazon RDS 与 Amazon Redshift 的零 ETL 集成

您只能修改与 Amazon Redshift 的零 ETL 集成的名称、描述和数据筛选选项。您无法修改用于加密集成的 Amazon KMS 密钥，也无法修改源数据库或目标数据库。

如果您将数据筛选条件添加到现有集成中，Amazon RDS 会重新评估筛选条件，就好像此筛选条件一直存在一样。它会移除当前位于目标 Amazon Redshift 数据仓库中，但与新筛选标准不匹配的所有数据。如果您从集成中移除 数据筛选条件，它会将以前不符合筛选标准（但现在符合）的所有数据复制到目标数据仓库中。有关更多信息，请参阅 the section called “零 ETL 集成的数据筛选”。

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 Amazon RDS API 修改零 ETL 集成。

RDS 控制台

修改零 ETL 集成

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择零 ETL 集成，然后选择要修改的集成。
3. 选择修改，然后对任何可用设置进行修改。
4. 当所有更改都达到您的要求时，选择修改。

Amazon CLI

要使用 Amazon CLI 修改零 ETL 集成，请调用 modify-integration 命令。与 `--integration-identifier` 一起，指定以下任意选项：

- `--integration-name` – 为集成指定新名称。
- `--description` – 为集成指定新描述。
- `--data-filter` – 为集成指定数据筛选选项。有关更多信息，请参阅 the section called “零 ETL 集成的数据筛选”。

Example

以下请求修改现有集成。

对于 Linux、macOS 或 Unix：

```
aws rds modify-integration \
    --integration-identifier ee605691-6c47-48e8-8622-83f99b1af374 \
    --integration-name my-renamed-integration
```

对于 Windows：

```
aws rds modify-integration ^
    --integration-identifier ee605691-6c47-48e8-8622-83f99b1af374 ^
    --integration-name my-renamed-integration
```

RDS API

要使用 RDS API 修改零 ETL 集成，请调用 ModifyIntegration 操作。指定集成标识符，以及您要修改的参数。

删除 Amazon RDS 与 Amazon Redshift 的零 ETL 集成

当您删除零 ETL 集成时，Amazon RDS 会将其从源数据库中删除。您的事务数据不会从 Amazon RDS 或 Amazon Redshift 中删除，但是 Amazon RDS 不会向 Amazon Redshift 发送新数据。

只有当集成的状态为 `Active`、`Failed`、`Syncing` 或 `Needs attention` 时，您才能将其删除。

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 删除零 ETL 集成。

控制台

删除零 ETL 集成

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 从左侧导航窗格中，选择零 ETL 集成。

3. 选择要删除的零 ETL 集成。
4. 依次选择操作和删除，然后确认删除。

Amazon CLI

要删除零 ETL 集成，请使用 delete-integration 命令并指定 `--integration-identifier` 选项。

Example

对于 Linux、macOS 或 Unix：

```
aws rds delete-integration \
    --integration-identifier ee605691-6c47-48e8-8622-83f99b1af374
```

对于 Windows：

```
aws rds delete-integration ^
    --integration-identifier ee605691-6c47-48e8-8622-83f99b1af374
```

RDS API

要使用 Amazon RDS API 删除零 ETL 集成，请结合 `IntegrationIdentifier` 参数使用 `DeleteIntegration` 操作。

Amazon RDS 与 Amazon Redshift 的零 ETL 集成问题排查

您可以通过在 Amazon Redshift 中查询 SVV_INTEGRATION 系统表来检查零 ETL 集成的状态。如果 `state` 列的值为 `ErrorState`，则表示有问题。有关更多信息，请参阅 the section called “使用系统表进行监控”。

使用以下信息来排查 Amazon RDS 与 Amazon Redshift 的零 ETL 集成的常见问题。

主题

- 我的一个或多个 Amazon Redshift 表需要重新同步

我无法创建零 ETL 集成

如果您无法创建零 ETL 集成，请确保源数据库的以下内容正确无误：

- 源数据库必须运行受支持的数据库引擎版本。有关受支持的版本的列表，请参阅 the section called "零 ETL 集成 "。
- 您正确配置了数据库参数。如果所需参数设置得不正确或与数据库不关联，则创建将失败。请参阅 the section called "步骤 1：创建自定义数据库参数组"。

此外，请确保您的数据仓库在以下方面正确无误：

- 已启用区分大小写。请参阅为您的数据仓库开启区分大小写。
- 您添加了正确的授权主体和集成源。请参阅为您的 Amazon Redshift 数据仓库配置授权。
- 数据仓库已加密（如果它是预调配集群）。请参阅 Amazon Redshift 数据库加密。

我的集成卡在 **Syncing** 状态

如果您更改其中一个必需的数据库集群参数的值，则集成状态可能会始终显示为 Syncing。

要修复此问题，请检查与源数据库关联的参数组中的参数值，并确保它们与所需值相匹配。有关更多信息，请参阅 the section called "步骤 1：创建自定义数据库参数组"。

如果您修改任何参数，请务必重启数据库来应用更改。

我的表未复制到 Amazon Redshift

如果您没有看到一个或多个表反映在 Amazon Redshift 中，可以运行以下命令来重新同步表：

```
ALTER DATABASE dbname INTEGRATION REFRESH TABLES table1, table2;
```

有关更多信息，请参阅《Amazon Redshift SQL 参考》中的 ALTER DATABASE。

您的数据可能未复制，因为一个或多个源表没有主键。Amazon Redshift 中的监控控制面板将这些表的状态显示为 Failed，而总体零 ETL 集成的状态更改为 Needs attention。要解决此问题，您可以在表中确定一个可以成为主键的现有键，也可以添加合成主键。有关详细解决方案，请参阅 Handle tables without primary keys while creating Amazon Aurora MySQL or Amazon RDS for MySQL zero-ETL integrations with Amazon Redshift。

我的一个或多个 Amazon Redshift 表需要重新同步

在源数据库上运行某些命令可能需要重新同步您的表。在这些情况下，SVV_INTEGRATION_TABLE_STATE 系统视图显示 `table_state` 为 `ResyncRequired`，这意味着集成必须将该特定表的数据从 MySQL 完全重新加载到 Amazon Redshift。

当表开始重新同步时，它进入 `Syncing` 状态。您无需执行任何手动操作即可重新同步表。在表数据重新同步时，您无法在 Amazon Redshift 中访问这些数据。

以下是一些可以使表进入 `ResyncRequired` 状态的示例操作，以及可供考虑的可能替代方案。

操作	示例	或者
在特定位置添加一列	`ALTER TABLE table_name ADD COLUMN column_name INTEGER NOT NULL first;`	Amazon Redshift 不支持使用 `first` 或 `after` 关键字将列添加到特定位置。如果目标表中列的顺序不重要，请使用更简单的命令将该列添加到表的末尾：`ALTER TABLE table_name ADD COLUMN column_name column_type ;`
使用默认 CURRENT_T	`ALTER TABLE table_name`	现有表行的 CURRENT_T

操作	示例	或者
IMESTAMP 添加时间戳列	`ADD COLUMN column_name  TIMESTAMP` `NOT NULL DEFAULT CURRENT_TIMESTAMP;`	IMESTAMP 值由 RDS for MySQL 计算，如果不进行完整表数据重新同步，就无法在 Amazon Redshift 中进行模拟。 如果可能，请将默认值切换为文本常量（例如 2023-01-01 00:00:15），以避免表可用性出现延迟。

在单个命令中执行多个列操作	`ALTER TABLE table_name`
	`  ADD COLUMN column_1,`
	`  RENAME COLUMN column_2 TO column_3;`
		考虑将命令拆分为两个单独的操作（ADD 和 RENAME），这不需要重新同步。

Amazon RDS for Db2

Amazon RDS 支持运行以下 IBM Db2 版本的数据库实例：

- Db2 Standard Edition
- Db2 Advanced Edition

Amazon RDS 支持运行以下 Db2 版本的数据库实例：

- Db2 11.5

有关次要版本支持的更多信息，请参阅 Amazon RDS 上的 Db2 版本。

在创建数据库实例之前，需要完成本用户指南的设置 Amazon RDS 环境部分中的步骤。当您使用主用户创建数据库实例时，该用户将获得 DBADM 权限，但有一些限制。将此用户用于管理任务，如创建其它数据库账户。您不能使用 SYSADM、SYSCTRL、SYSMAINT 实例级权限或 SECADM 数据库级权限。

您可执行以下操作：

- 数据库实例
- 数据库快照
- 时间点恢复
- 自动存储备份
- 手动存储备份

您可以在虚拟私有云（VPC）中使用运行 Db2 的数据库实例。还可以通过启用各种选项向 Amazon RDS for Db2 数据库实例添加功能。Amazon RDS 支持将 RDS for Db2 的多可用区部署作为一种高可用性失效转移解决方案。

> **Important**
> 为了提供托管服务体验，Amazon RDS 未提供数据库实例的 Shell 访问权限。它还限制了对需要提升权限的某些系统过程和表的访问。您可以使用标准 SQL 客户端（例如 IBM Db2 CLP）访问您的数据库。但是，您无法使用远程登录或安全外壳 (SSH) 直接访问主机。

主题

- Amazon RDS 上的 Db2 概述
- 创建 Amazon RDS for Db2 数据库实例的先决条件
- Amazon RDS for Db2 数据库实例上的多个数据库
- 连接到 Db2 数据库实例
- 保护 Amazon RDS for Db2 数据库实例连接
- 管理您的 Amazon RDS for Db2 数据库实例
- 将 Amazon RDS for Db2 数据库实例与 Amazon S3 集成
- 将数据迁移到 Amazon RDS for Db2
- Amazon RDS for Db2 联合身份验证
- 使用适用于 Amazon RDS for Db2 的副本
- Amazon RDS for Db2 数据库实例选项
- Amazon RDS for Db2 的外部存储过程
- Amazon RDS for Db2 的已知问题和限制
- Amazon RDS for Db2 存储过程参考
- Amazon RDS for Db2 用户定义的函数参考
- Amazon RDS for Db2 故障排除

Amazon RDS 上的 Db2 概述

您可以阅读以下部分以获取 Amazon RDS 上的 Db2 概述。

主题

- Amazon RDS for Db2 功能
- Amazon RDS 上的 Db2 版本
- Amazon RDS for Db2 许可选项
- Amazon RDS for Db2 实例类
- Amazon RDS for Db2 默认角色
- Amazon RDS for Db2 参数
- Amazon RDS 上 Db2 数据库的 EBCDIC 排序规则

- Amazon RDS for Db2 数据库实例的本地时区

Amazon RDS for Db2 功能

Amazon RDS for Db2 支持 IBM Db2 数据库的大部分功能。有些功能可能仅提供有限支持或受限制的权限。有关特定 Db2 版本的 Db2 数据库功能的更多信息，请参阅 IBM Db2 文档。

您可以在数据库的新增功能？页面上筛选新的 Amazon RDS 功能。对于 Products (产品)，请选择 Amazon RDS。然后，可以使用 **`Db2 2023`** 之类的关键词进行搜索。

> **Note**
>
> 以下列表并不详尽。

主题

- RDS for Db2 中支持的功能
- RDS for Db2 中不支持的功能

RDS for Db2 中支持的功能

RDS for Db2 支持的功能包括 IBM Db2 原生功能和 Amazon RDS 核心功能。

IBM Db2 的原生功能

RDS for Db2 支持以下 Db2 数据库功能：

- 创建使用客户定义的代码集、排序规则、页面大小和地区的标准数据库。使用 Amazon RDS rdsadmin.create_database 存储过程。
- 添加、删除或修改本地用户和组。将 Amazon RDS 存储过程用于用于对 RDS for Db2 授予和撤销权限的存储过程。
- 使用 Amazon RDS rdsadmin.create_role 存储过程创建角色。
- 支持按行排列的标准表。
- 支持按列排列的表的分析工作负载。
- 能够定义 Db2 兼容性功能，例如 Oracle 和 MySQL。
- 支持基于 Java 的外部存储过程。

- 支持使用 SSL/TLS 进行传输中数据加密。
- 监控数据库的状态（ALIVE、DOWN、STORAGE_FULL、UNKNOWN 和 STANDBY_CONNECTABLE）。
- 还原客户提供的脱机或联机 Linux (LE) 数据库。将 Amazon RDS 存储过程用于用于管理 RDS for Db2 数据库的存储过程。
- 应用客户提供的 Db2 存档日志，以使数据库与自行管理的 Db2 数据库保持同步。将 Amazon RDS 存储过程用于用于管理 RDS for Db2 数据库的存储过程。
- 支持 Db2 实例级和数据库级审核。
- 支持同构联合身份验证。
- 能够从 Amazon Simple Storage Service（Amazon S3）中的数据文件加载表。
- 授予用户、组或角色的授权，例如 CONNECT、SYSMON、ACCESSCTRL、DATAACCESS、SQLADM、WLMADM、EXPLAIN、LOAD 或 IMPLICIT_SCHEMA。
- 创建多个数据库。

> **Note**
>
> 一个 RDS for Db2 数据库实例可以包含多达 50 个数据库。有关更多信息，请参阅 the section called “多个 Db2 数据库”。

Amazon RDS 的核心功能

RDS for Db2 支持以下核心 Amazon RDS 功能：

- 要分配给数据库实例的自定义参数组
- 创建、修改和删除数据库实例
- 还原自行管理的 Db2 脱机或联机 Linux (LE) 数据库备份
- 支持 gp3、io2 和 io1 存储类型
- 使用 Amazon Managed Microsoft AD 对 RDS for Db2 进行 Kerberos 身份验证和 LDAP 组授权
- 修改现有 Db2 实例的安全组、端口、实例类型、存储、备份保留期和其它设置
- 数据库实例的删除保护
- 跨区域时间点故障恢复（PITR）
- 使用 Amazon Key Management Service（Amazon KMS）进行存储加密和静态加密
- 多可用区数据库实例，带有一个备用实例以实现高可用性

- 重启数据库实例
- 更新主密码
- 将数据库实例还原到特定时间
- 使用存储级备份来备份和还原数据库实例
- 启动和停止数据库实例
- 维护数据库实例

RDS for Db2 中不支持的功能

RDS for Db2 不支持以下 Db2 数据库功能：

- 主用户的 `SYSADM`、`SECADM` 和 `SYSMAINT` 访问权限。
- 用 C、C++ 或 Cobol 编写的外部存储过程。
- 单台主机上有多个 Db2 数据库实例。
- 用于身份验证的外部 GSS-API 插件。
- 用于备份或还原 Db2 数据库的外部第三方插件。
- 多节点大规模并行处理（MPP），例如 IBM Db2 Warehouse。
- IBM Db2 pureScale.
- 为于 RDS for Db2 手动设置高可用性灾难恢复（HADR）。

> **Note**
>
> Amazon RDS 通过副本为 RDS for Db2 支持和管理 HADR。有关更多信息，请参阅使用适用于 Amazon RDS for Db2 的副本。
> RDS for Db2 支持多可用区部署、跨区域自动备份和复制。有关更多信息，请参阅Amazon RDS 的多可用区数据库实例部署和将自动备份复制到其他 Amazon Web Services 区域。

- 原生数据库加密。
- 与 Informix、Sybase 和 Teradata 的异构联合身份验证。有关更多信息，请参阅 the section called "联合身份验证"。
- 加密备份的跨区域时间点故障恢复（PITR）。
- 通过备份和还原数据，创建非防护例程并迁移现有的非防护例程。有关更多信息，请参阅非防护例程。

- 创建新的非自动存储表空间。有关更多信息，请参阅迁移期间的非自动存储表空间。
- 外部表。

Amazon RDS 上的 Db2 版本

对于 Db2，版本号采用 major.minor.build.revision 的形式，例如 11.5.9.0.sb00000000.r1。我们的版本实现与 Db2 的版本实现一致。

major

主要版本号是版本号的整数和第一个小数点后的部分，例如 11.5。如果主要版本号更改，则将版本更改视为主要版本更改；例如，从版本 11.5 更改为 12.1。

minor

次要版本号是版本号的第三部分和第四部分，例如 11.5.9.0 中的 9.0。第三部分表示 Db2 模组包，例如 9.0 中的 9。第四部分表示 Db2 修复包，例如 9.0 中的 0。如果 Db2 模组包或 Db2 修复包发生变化（例如，从版本 11.5.9.0 更改为 11.5.9.1，或者从 11.5.9.0 更改为 11.5.10.0），则版本更改被视为次要更改，但提供目录表更新除外。（Amazon RDS 会处理这些例外情况。）

build

内部版本号是版本号的第五部分，例如 11.5.9.0.sb00000000 中的 sb00000000。编号部分全部为零的内部版本号表示标准内部版本。编号部分不全为零的内部版本号表示特殊内部版本。如果现有 Db2 版本有安全修复或特殊内部版本，则内部版本号会发生变化。内部版本号的更改还表明 Amazon RDS 自动应用了新的次要版本。

修订

修订号是版本号的第六部分，例如 11.5.9.0.sb00000000.r1 中的 r1。修订是对现有 Db2 版本的 Amazon RDS 修订。修订号更改表明 Amazon RDS 自动应用了新的次要版本。

主题

Amazon RDS 上支持的 Db2 次要版本

下表显示了 Amazon RDS 当前支持的 Db2 11.5 的次要版本。

> **Note**
> 只有月份和年份的发布日期均为大概日期，这些日期会在确切日期明确后进行更新。

Db2 引擎版本	IBM 发布日期	RDS 发布日期
11.5.9.0	2023 年 11 月 15 日	2023 年 11 月 27 日

在创建新数据库实例时，您可以指定目前支持的任何 Db2 版本。您可以指定主要版本（例如 Db2 11.5），以及所指定主要版本的任何受支持的次要版本。如果没有指定版本，Amazon RDS 会默认使用支持的版本，通常是最新的版本。如果指定了主要版本但未指定次要版本，Amazon RDS 将默认使用您指定的主要版本的最新版本。要查看受支持版本的列表以及新建数据库实例的默认值，请使用 describe-db-engine-versions Amazon Command Line Interface（Amazon CLI）命令。

例如，要列出 Amazon RDS for Db2 支持的引擎版本，请运行以下 Amazon CLI 命令。将*##*替换为您的 Amazon Web Services 区域。

对于 Linux、macOS 或 Unix：

```
aws rds describe-db-engine-versions \
    --filters Name=engine,Values=db2-ae,db2-se \
    --query "DBEngineVersions[].{Engine:Engine, EngineVersion:EngineVersion,
 DBParameterGroupFamily:DBParameterGroupFamily}" \
    --region region
```

对于 Windows：

```
aws rds describe-db-engine-versions ^
    --filters Name=engine,Values=db2-ae,db2-se ^
    --query "DBEngineVersions[].{Engine:Engine, EngineVersion:EngineVersion,
 DBParameterGroupFamily:DBParameterGroupFamily}" ^
    --region region
```

此命令生成类似于以下示例的输出：

```
[
    {
```

```
    "Engine": "db2-ae",
    "EngineVersion": "11.5.9.0.sb00000000.r1",
    "DBParameterGroupFamily": "db2-ae-11.5"
    },
    {
    "Engine": "db2-se",
    "EngineVersion": "11.5.9.0.sb00000000.r1",
    "DBParameterGroupFamily": "db2-se-11.5"
    }
]
```

默认 Db2 版本可能因 Amazon Web Services 区域而异。要创建具有特定次要版本的数据库实例，请在数据库实例创建过程中指定次要版本。您可以通过运行 `describe-db-engine-versions` 命令来确定适用于 Amazon Web Services 区域的 `db2-ae` 和 `db2-se` 数据库引擎的默认版本。以下示例返回美国东部（弗吉尼亚州北部）中 `db2-ae` 的默认版本。

对于 Linux、macOS 或 Unix：

```
aws rds describe-db-engine-versions \
    --default-only --engine db2-ae \
    --query "DBEngineVersions[].{Engine:Engine, EngineVersion:EngineVersion,
 DBParameterGroupFamily:DBParameterGroupFamily}" \
    --region us-east-1
```

对于 Windows：

```
aws rds describe-db-engine-versions ^
    --default-only --engine db2-ae ^
    --query "DBEngineVersions[].{Engine:Engine, EngineVersion:EngineVersion,
 DBParameterGroupFamily:DBParameterGroupFamily}" ^
    --region us-east-1
```

此命令生成类似于以下示例的输出：

```
[
    {
    "Engine": "db2-ae",
    "EngineVersion": "11.5.9.0.sb00000000.r1",
    "DBParameterGroupFamily": "db2-ae-11.5"
    }
]
```

借助于 Amazon RDS，您可以控制何时将 Db2 实例升级到 Amazon RDS 支持的新主要版本。可与 Db2 的特定版本保持兼容、对应用程序测试新版本，然后再部署到生产环境中以及在最适合计划的时间执行主要版本升级。

启用自动次要版本升级后，Amazon RDS 会自动将您的数据库实例升级到新的 Db2 次要版本，因为它们受到 Amazon RDS 支持。此修补在计划的维护时段内进行。您可以修改数据库实例以启用或禁用自动次要版本升级。

除了 Db2 版本 11.5.9.1 和 11.5.10.0 之外，自动升级到新 Db2 次要版本包括自动升级到新的内部版本和修订版。对于 11.5.9.1 和 11.5.10.0，请手动升级次要版本。

如果选择退出自动计划的升级，您可以按照与主要版本更新相同的步骤手动升级到支持的次要版本。有关信息，请参阅升级数据库实例引擎版本。

Amazon RDS 上支持的 Db2 主要版本

RDS for Db2 主要版本在标准支持下可用，至少到相应 IBM 版本的 IBM 支持结束日期（基本）为止。下表显示了可用于计划测试和升级周期的日期。如果 Amazon 对某个 RDS for Db2 版本的支持时间比原定时间更长，我们会计划更新此表以反映较晚的日期。

您可以参照下列日期规划您的测试和升级周期。

> **Note**
> 只有月份和年份的发布日期均为大概日期，这些日期会在确切日期明确后进行更新。
> 您还可以运行 describe-db-major-engine-versions Amazon CLI 命令或者使用 DescribeDBMajorEngineVersions RDS API 操作，来查看 Db2 数据库的主要版本。

Db2 主要版本	IBM 发行日期	RDS 发布日期	IBM 终止支持（标准版和高级版）	IBM 支持结束日期（扩展）
Db2 11.5	2019 年 6 月 27 日	2023 年 11 月 27 日	IBM 尚未宣布。[1]	支持结束后 4 年内

[1] 有关最新信息，请参阅 IBM Db2 文档中的 Db2 product end-of-support dates。

Amazon RDS for Db2 许可选项

Amazon RDS for Db2 提供两种许可选项：自带许可（BYOL）和通过 Amazon Web Services Marketplace 获得 Db2 许可证。

主题

- 自带 Db2 许可
- 通过 Amazon Web Services Marketplace 获得 Db2 许可证
- 在 Db2 许可证之间切换

自带 Db2 许可

在 BYOL 模式下，使用您现有的 Db2 数据库许可证在 Amazon RDS 上部署数据库。确认您拥有与要运行的数据库实例类和 Db2 数据库版本对应的 Db2 数据库许可证。您还必须遵循 IBM 对云计算环境中的 IBM 数据库软件授予许可的策略。

> **Note**
>
> 多可用区数据库实例是冷备用实例，因为 Db2 数据库已安装但未运行。备用数据库无法读取、运行或处理请求。有关更多信息，请参阅 IBM 网站上的 IBM Db2 许可信息。

在此模式下，您可以继续使用有效的 IBM 支持账户，并能直接联系 IBM 解决 Db2 数据库服务请求。如果您拥有可享受案例支持的Amazon Web Services 支持账户，可以联系 Amazon Web Services 支持 了解 Amazon RDS 问题。Amazon Web Services 和 IBM 为需要两个组织提供帮助的案例提供多供应商支持流程。

Amazon RDS 对于 Db2 Standard Edition 和 Db2 Advanced Edition 支持 BYOL 模型。

主题

- Db2 自带许可的 IBM ID
- 将 IBM ID 添加到 RDS for Db2 数据库实例的参数组中
- 集成 Amazon License Manager

Db2 自带许可的 IBM ID

在 BYOL 模式中，您需要 IBM Customer ID 和 IBM Site ID 来创建、修改或还原 RDS for Db2 数据库实例。在创建 RDS for Db2 数据库实例之前，必须使用 IBM Customer ID 和 IBM Site ID 创建自定义参数组。有关更多信息，请参阅 将 IBM ID 添加到 RDS for Db2 数据库实例的参数组中。您可以在同一 Amazon Web Services 账户或 Amazon Web Services 区域中运行多个具有不同 IBM Customer IDs 和 IBM Site IDs 的 RDS for Db2 数据库实例。

> **Important**
>
> 如果我们无法通过 IBM Customer ID 和 IBM Site ID 验证您的许可证，我们可能会终止使用这些未经验证的许可证运行的所有数据库实例。

如果您是新的 IBM Db2 客户，则必须先从 IBM 购买 Db2 软件许可证。购买 Db2 软件许可证后，您将收到一份来自 IBM 的权利证明，其中列出了您的 IBM Customer ID 和 IBM Site ID。

如果您是现有 IBM Db2 客户，则可以从 IBM 提供的权利证明证书上找到您的 IBM Customer ID 和 IBM Site ID。

也可以在您的 IBM Passport Advantage Online 账户中找到 IBM Customer ID 和 IBM Site ID。登录后，您可以在主页或软件下载页面上查看这两个 ID。

将 IBM ID 添加到 RDS for Db2 数据库实例的参数组中

由于您无法修改默认参数组，因此必须创建自定义参数组，然后对其进行修改以包含 IBM Customer ID 和 IBM Site ID 的值。有关参数组的信息，请参阅 Amazon RDS 数据库实例的数据库参数组。

> **Important**
>
> 在创建 RDS for Db2 数据库实例之前，必须使用 IBM Customer ID 和 IBM Site ID 创建自定义参数组。

使用下表中的参数设置。

参数	值
`rds.ibm_customer_id`	`<your IBM Customer ID>`

参数	值
rds.ibm_site_id	<your IBM Site ID>
ApplyMethod	immediate，pending-reboot

这些参数是动态的，这意味着对它们所做的任何更改都会立即生效，并且您无需重启数据库实例。如果您不希望更改立即生效，则可以将 ApplyMethod 设置为 pending-reboot，并安排在维护时段期间进行这些更改。

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 Amazon RDS API 创建和修改自定义参数组。

控制台

将您的 IBM Customer ID 和 IBM Site ID 添加到参数组

1. 创建新的数据库参数组。有关创建数据库参数组的详细信息，请参阅 在 Amazon RDS 中创建数据库参数组。
2. 修改您创建的参数组。有关修改参数组的更多信息，请参阅在 Amazon RDS 中修改数据库参数组中的参数。

Amazon CLI

将您的 IBM Customer ID 和 IBM Site ID 添加到参数组

1. 通过运行 create-db-parameter-group 命令创建自定义参数组。

   包括以下必需选项：

   - `--db-parameter-group-name` - 您正在创建的参数组的名称。
   - `--db-parameter-group-family` – Db2 引擎版本和主要版本。有效值：`db2-se-11.5`、`db2-ae-11.5`。
   - `--description` – 此参数组的描述。

   有关创建数据库参数组的详细信息，请参阅 在 Amazon RDS 中创建数据库参数组。
2. 通过运行 modify-db-parameter-group 命令修改您创建的自定义参数组中的参数。

包括以下必需选项：

- `--db-parameter-group-name` – 您创建的参数组的名称。
- `--parameters` – 参数名称、值和参数更新应用方法的数组。

有关修改参数组的更多信息，请参阅在 Amazon RDS 中修改数据库参数组中的参数。

RDS API

将您的 IBM Customer ID 和 IBM Site ID 添加到参数组

1. 使用 Amazon RDS API CreateDBParameterGroup 操作创建自定义数据库参数组。

   包括以下必需参数：

   - `DBParameterGroupName`
   - `DBParameterGroupFamily`
   - `Description`

   有关创建数据库参数组的详细信息，请参阅 在 Amazon RDS 中创建数据库参数组。
2. 修改您使用 RDS API ModifyDBParameterGroup 操作创建的自定义参数组中的参数。

   包括以下必需参数：

   - `DBParameterGroupName`
   - `Parameters`

   有关修改参数组的更多信息，请参阅在 Amazon RDS 中修改数据库参数组中的参数。

现在，您已准备好创建数据库实例并将自定义参数组附加到该数据库实例。有关更多信息，请参阅创建 Amazon RDS 数据库实例 和在 Amazon RDS 中将数据库参数组与数据库实例关联。

集成 Amazon License Manager

为了协助在 BYOL 模型中监控 RDS for Db2 许可证的使用情况，Amazon License Manager 与 RDS for Db2 相集成。License Manager 基于虚拟 CPU（vCPU）为跟踪 RDS for Db2 引擎版本提供支持。您也可以将 License Manager 与 Amazon Organizations 一起使用以集中管理组织的所有账户。

要跟踪 RDS for Db2 数据库实例的许可证使用情况，必须创建自管理许可证。可以使用 Amazon Web Services Management Console、Amazon License Manager CLI 和 Amazon License Manager API 创建自管理许可证。或者，也可以使用 Amazon CloudFormation 和 Terraform 模板自动创建自管理许可证。

与产品信息筛选条件匹配的 RDS for Db2 资源会自动与自管理许可证相关联。RDS for Db2 数据库实例的发现操作可能需要长达 24 小时。

下表显示了 RDS for Db2 的引擎版本产品信息筛选条件的可用值。

值	描述
`db2-se`	Db2 标准版
`db2-ae`	Db2 高级版

主题

- 术语
- 在 Amazon License Manager 中创建自管理许可证
- 使用模板在 Amazon License Manager 中自动创建自管理许可证
- 创建自管理许可证的相关设置

术语

本页使用以下术语讨论 Amazon RDS 与 Amazon License Manager 的集成。

自管理许可证

> 自管理许可证是 Amazon License Manager 中使用的术语。Amazon RDS 控制台将许可证称为 Amazon License Manager 配置。自管理许可证包含基于企业协议条款的许可规则。您创建的规则决定了 Amazon 如何处理使用许可证的命令。在创建自管理许可证时，请与组织的合规性团队密切合作来审核企业协议。有关更多信息，请参阅 Self-managed licenses in License Manager。

在 Amazon License Manager 中创建自管理许可证

您可以使用 Amazon Web Services Management Console、Amazon License Manager CLI 和 Amazon License Manager API 创建自管理许可证。

> **Note**
>
> 如果使用 Amazon Web Services Management Console 创建 RDS for Db2 数据库实例，您将通过输入自管理许可证的名称来创建该许可证。然后，Amazon RDS 将数据库实例与该许可证关联。（在 Amazon RDS 控制台中，此许可证称为 Amazon License Manager 配置。）如果要使用 Amazon License Manager CLI 或 Amazon License Manager API 创建 RDS for Db2 数据库实例，必须先使用以下步骤创建自管理许可证。同样的情况也适用于将 RDS for Db2 数据库实例还原到某个时间点或从快照还原。

控制台

创建自管理许可证来跟踪您的 RDS for Db2 数据库实例的许可证使用情况

1. 转到 https://console.aws.amazon.com/license-manager/。
2. 创建自管理许可证。

   有关说明，请参阅《Amazon License Manager 用户指南》中的 Create a self-managed license。

   在 Product Information (产品信息) 面板中为 RDS Product Information Filter (RDS 产品信息筛选器) 添加规则。

   有关更多信息，请参阅 Amazon License Manager API 参考 中的 产品信息。

Amazon License Manager CLI

> **Note**
>
> 此过程使用 Amazon License Manager CLI 命令。

要使用 Amazon CLI 创建自管理许可证，可运行 Amazon License Manager create-license-configuration 命令。使用 `--cli-input-json` 或 `--cli-input-yaml` 选项将选项传递给该命令。

有关更多信息，请参阅 the section called “创建自管理许可证的相关设置”。

下面的命令为 Db2 标准版创建自管理许可证。

```
aws license-manager create-license-configuration --cli-input-json file://rds-db2-
se.json
```

以下 JSON 是前一个命令中使用的 `rds-db2-se.json` 文件的内容。

```
{
    "Name": "rds-db2-se",
    "Description": "RDS Db2 Standard Edition",
    "LicenseCountingType": "vCPU",
    "LicenseCountHardLimit": false,
    "ProductInformationList": [
        {
            "ResourceType": "RDS",
            "ProductInformationFilterList": [
                {
                    "ProductInformationFilterName": "Engine Edition",
                    "ProductInformationFilterValue": ["db2-se"],
                    "ProductInformationFilterComparator": "EQUALS"
                }
            ]
        }
    ]
}
```

有关产品信息的更多信息，请参阅 Amazon License Manager 用户指南 中的资源清单自动发现。

有关 `--cli-input` 参数的更多信息，请参阅 Amazon CLI 用户指南 中的从 JSON 或 YAML 输入文件生成 Amazon CLI 骨架和输入参数。

Amazon License Manager API

> **Note**
>
> 此过程使用 Amazon License Manager API 命令。

要创建自管理许可证，请使用带有以下必需参数的 `CreateLicenseConfiguration` Amazon License Manager API 操作：

- Name

- `LicenseCountingType`
- `ProductInformationList`
- `ResourceType`
- `ProductInformationFilterList`
- `ProductInformationFilterName`
- `ProductInformationFilterValue`
- `ProductInformationFilterComparator`

有关这些参数的更多信息，请参阅the section called “创建自管理许可证的相关设置”。

使用模板在 Amazon License Manager 中自动创建自管理许可证

您可以使用 Amazon CloudFormation 和 Terraform 模板自动创建自管理许可证。

以下 Amazon CloudFormation 模板示例在 RDS for Db2 上为 Db2 标准版创建自管理许可证。对于 Db2 高级版模板，请更新 `Name`、`Description` 和 `ProductInformationFilter` 的值。

```
AWSTemplateFormatVersion: "2010-09-09"
Description: CloudFormation template to create a License Configuration for Db2 Standard
 Edition on RDS for Db2.

Resources:
  Db2LicenseConfiguration:
    Type: "AWS::LicenseManager::LicenseConfiguration"
    Properties:
      Name: "rds-db2-se"
      Description: "Db2 Standard Edition on RDS for Db2"
      LicenseCountingType: "vCPU"
      LicenseCountHardLimit: false
      ProductInformationList:
        - ResourceType: "RDS"
          ProductInformationFilterList:
            - ProductInformationFilterName: "Engine Edition"
              ProductInformationFilterValue:
                - "db2-se"
              ProductInformationFilterComparator: "EQUALS"
```

有关将 Amazon CloudFormation 与 Amazon RDS 结合使用的更多信息，请参阅 使用 Amazon CloudFormation 创建 Amazon RDS 资源。

以下 Terraform 模板示例在 RDS for Db2 上为 Db2 标准版创建自管理许可证。将 *us-east-1* 替换为您的 Amazon Web Services 区域。对于 Db2 高级版模板，请更新 `name`、`description` 和 `product_information_filter` 的值。

```
provider "aws" {
  region = "us-east-1"
}

resource "aws_licensemanager_license_configuration" "rds_db2_license_config" {
  name                     = "rds-db2-se"
  description              = "Db2 Standard Edition on RDS for Db2
  license_counting_type    = "vCPU"
  license_count_hard_limit = false

  product_information_list {
    resource_type = "RDS"

    product_information_filter {
      name       = "Engine Edition"
      comparator = "EQUALS"
      value      = ["db2-se"]
    }
  }
}
```

有关使用 Terraform 和 Amazon RDS 的更多信息，请参阅《Amazon 规范性指导》中的 Using Terraform as an IaC tool for the Amazon Web Services 云 和 Best practices for using the Terraform Amazon Provider。

创建自管理许可证的相关设置

在下表中，可以找到有关使用 Amazon License Manager CLI、Amazon License Manager API、Amazon CloudFormation 模板和 Terraform 模板创建自管理许可证的相关设置的详细信息。下表中的参数名称以 Amazon License Manager API 和 Amazon CloudFormation 模板中所使用名称的格式显示。

参数名称	数据类型	必需	描述
名称	字符串	是	许可证配置的名称。
描述	字符串	否	许可证配置的描述。

参数名称	数据类型	必需	描述
LicenseCountingType	字符串	是	用于跟踪许可证清单的维度。有效值：vCPU。
LicenseCountHardLimit	布尔值	否	指示是使用硬许可证实施还是软许可证实施。超过硬限制会阻止新实例的启动。
ProductInformationList	对象数组	是	许可证配置的产品信息列表。
ResourceType	字符串	是	资源类型。有效值：RDS。
ProductInformationFilterList	对象数组	是	许可证配置的产品信息筛选条件列表。
ProductInformationFilterName	字符串	是	所声明的筛选条件类型的名称。有效值：Engine Edition。
ProductInformationFilterValue	字符串数组	是	要筛选的值。必须只指定一个值。有效值：db2-se 或 db2-ae。
ProductInformationFilterComparator	字符串	是	ProductInformationFilterName 的逻辑运算符。有效值：EQUALS。

通过 Amazon Web Services Marketplace 获得 Db2 许可证

采用"通过 Amazon Web Services Marketplace 获得 Db2 许可证"模式时，您需要按小时费率订阅 Db2 许可证。此模式可帮助您快速开始使用 RDS for Db2，而无需购买许可证。

要采用"通过 Amazon Web Services Marketplace 获得 Db2 许可证"模式，您需要针对要使用的特定 IBM Db2 版本进行有效 Amazon Web Services Marketplace 订阅。如果您还没有订阅，请为该 IBM Db2 版本订阅 Amazon Web Services Marketplace。

Amazon RDS 支持对 IBM Db2 标准版和 IBM Db2 高级版采用"通过 Amazon Web Services Marketplace 获得 Db2 许可证"模式。

主题

术语

本页使用以下术语讨论 Amazon RDS 与 Amazon Web Services Marketplace 的集成。

SaaS 订阅

　在 Amazon Web Services Marketplace 中，软件即服务（SaaS）产品（如即用即付许可模式）采用基于使用情况的订阅模式。Db2 软件经销商 IBM 可以跟踪您的使用情况，您只需按实际使用量付费。

公开报价

　使用公开报价，您可以直接从 Amazon Web Services Management Console购买 Amazon Web Services Marketplace 产品。

专属优惠

　专属优惠是一项购买计划，允许卖家和买家协商在 Amazon Web Services Marketplace 中进行采购时的自定义价格和最终用户许可协议（EULA）条款。

Db2 市场费用

　IBM 为使用 Db2 软件许可证收取的费用 这些服务费通过 Amazon Web Services Marketplace 计量，并显示在您的 Amazon 账单上的 Amazon Web Services Marketplace 部分下。

Amazon RDS 费用

> Amazon 针对 RDS for Db2 服务收取的费用，其中不包括将 Amazon Web Services Marketplace 用于 Db2 许可证时的许可费用。费用通过正在使用的 Amazon RDS 服务计量并显示在您的 Amazon 账单上。

付款和计费

将 RDS for Db2 与 Amazon Web Services Marketplace 进行集成，为 Db2 提供按小时计费的即用即付许可证。Db2 Marketplace 费用包括 Db2 软件的许可费用，Amazon RDS 费用涵盖您的 RDS for Db2 数据库实例的使用费用。有关定价的更多信息，请参阅 [Amazon RDS for Db2 定价]。

要停止收取这些费用，您必须删除所有 RDS for Db2 数据库实例。此外，您还可以为 Db2 许可证取消对 Amazon Web Services Marketplace 的订阅。如果您在不删除数据库实例的情况下取消订阅，Amazon RDS 将继续向您收取数据库实例的使用费。有关更多信息，请参阅 the section called “删除数据库实例”。

您可以在 Amazon Billing 控制台中，查看采用“通过 Amazon Web Services Marketplace 获得 Db2 许可证”模式的 RDS for Db2 数据库实例的账单并管理其付款。您的账单包括两项费用：一项是您通过 Amazon Web Services Marketplace 使用 Db2 许可证的费用，另一项是您使用 Amazon RDS 的费用。有关更多信息，请参阅《Amazon 账单与成本管理 用户指南》中的查看您的账单。

订阅 Db2 Marketplace 清单并注册 IBM

要通过 Amazon Web Services Marketplace 使用 Db2 许可证，必须使用 Amazon Web Services Management Console完成以下两项任务。您无法通过 Amazon CLI 或 RDS API 完成这些任务。

> **Note**
>
> 如果您想使用 Amazon CLI 或 RDS API 创建数据库实例，则必须先完成这两项任务。

主题

任务 1：在 Amazon Web Services Marketplace 中订阅 Db2

要将 Db2 许可证与 Amazon Web Services Marketplace 一起使用，您需要拥有 Db2 的有效 Amazon Web Services Marketplace 订阅。由于订阅与特定 IBM Db2 版本相关联，因此您需要在 Amazon Web Services Marketplace 中为要使用的每个 Db2 版本（IBM Db2 高级版、IBM Db2 标准版）订阅 Db2：有关 Amazon Web Services Marketplace 订阅的信息，请参阅《Amazon Web Services Marketplace 买家指南》中的基于 Saas 使用量的订阅。

建议您在开始创建数据库实例之前，在 Amazon Web Services Marketplace 中订阅 Db2。

任务 2：向 IBM 注册订阅

在 Amazon Web Services Marketplace 中订阅 Db2 后，请从您选择的 Db2 订阅类型的 Amazon Web Services Marketplace 页面完成 IBM 订单注册。在 Amazon Web Services Marketplace 页面上，选择查看购买选项，然后选择设置账户。您可以使用现有 IBM 账户或通过创建免费 IBM 账户，进行注册。

获取专属优惠

您可以向 IBM 申请 Db2 的 Amazon Web Services Marketplace 专属优惠。有关更多信息，请参阅《Amazon Web Services Marketplace 买家指南》中的 Private offers。

> **Note**
>
> 如果您是 Amazon Organizations 用户并收到了发给您的付款人和成员账户的专属优惠，请按照以下步骤直接在组织中的每个账户上订阅 Db2。

获取 Db2 专属优惠

1. 发出专属优惠之后，登录到 Amazon Web Services Marketplace 控制台。
2. 打开带有 Db2 专属优惠链接的电子邮件。
3. 单击此链接，以直接访问专属优惠。

   > **Note**
   >
   > 当在登录到正确的账户之前单击此链接时，将产生一个找不到页面 (404) 错误。

4. 查看条款和条件。
5. 选择接受条款。

> **Note**
>
> 如果不接受 Amazon Web Services Marketplace 专属优惠，Amazon Web Services Marketplace 的 Db2 服务费将继续按公开小时费率计费。

6. 要验证优惠详细信息，请选择产品列表中的显示详细信息。

完成该过程后，您可以按照the section called "创建数据库实例"中的步骤创建数据库实例。在 Amazon Web Services Management Console中，在许可证下，确保选择通过 Amazon Web Services Marketplace。

在 Db2 许可证之间切换

您可以在 RDS for Db2 中，在 Db2 许可证之间进行切换。例如，您可以从"自带许可证"模式开始，然后切换到"通过 Amazon Web Services Marketplace 获得 Db2 许可证"模式。

> **Important**
>
> 如果要切换到"通过 Amazon Web Services Marketplace 获得 Db2 许可证"模式，请确保您对要使用的 IBM Db2 版本拥有有效的 Amazon Web Services Marketplace 订阅。如果没有，请在 Amazon Web Services Marketplace 中为该 Db2 版本订阅 Db2，然后完成还原过程。

控制台

在 Db2 许可证之间切换

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Automated backups（自动备份）。

   自动备份便会显示在 Current Region（当前区域）选项卡上。
3. 选择要还原的数据库实例。
4. 对于 Actions (操作)，选择 Restore to point in time (还原到时间点)。

   此时会显示还原到时间点窗口。
5. 选择最近可还原时间以还原到可能的最近时间，或选择自定义来选择时间。

如果选择自定义，请输入要将实例还原到的日期和时间。

> **Note**
>
> 时间以您的本地时区显示，表示为协调世界时 (UTC) 的偏移量。例如，UTC-5 是东部标准时间/中部夏令时。

6. 对于数据库引擎，请选择要使用的 Db2 许可证。
7. 对于数据库实例标识符，请输入目标还原后的数据库实例的名称。名称必须唯一。
8. 根据需要选择其他选项，例如数据库实例类、存储，以及是否要使用存储自动扩展。

   有关每项设置的信息，请参阅 数据库实例的设置。
9. 选择还原到时间点。

有关更多信息，请参阅 将 Amazon RDS 的数据库实例还原到指定时间。

Amazon CLI

要在 Db2 许可证之间切换，可运行 restore-db-instance-to-point-in-time 命令。以下示例还原最新的时间点版本，将数据库引擎设置为 IBM Db2 高级版，并将许可模式设置为“通过 Amazon Web Services Marketplace 获得 Db2 许可证”。

您可以指定其他设置。有关每项设置的信息，请参阅 数据库实例的设置。

Example

对于 Linux、macOS 或 Unix：

```
aws rds restore-db-instance-to-point-in-time \
    --source-db-instance-identifier my_source_db_instance \
    --target-db-instance-identifier my_target_db_instance \
    --use-latest-restorable-time \
    --engine db2-ae \
    --license-model marketplace-license
```

对于 Windows：

```
aws rds restore-db-instance-to-point-in-time ^
    --source-db-instance-identifier my_source_db_instance ^
```

```
    --target-db-instance-identifier my_target_db_instance ^
    --use-latest-restorable-time ^
    --engine db2-ae ^
    --license-model marketplace-license
```

有关更多信息，请参阅 将 Amazon RDS 的数据库实例还原到指定时间。

RDS API

要在 Db2 许可证之间切换，请使用以下参数调用 Amazon RDS API `RestoreDBInstanceToPointInTime` 操作：

- `SourceDBInstanceIdentifier`
- `TargetDBInstanceIdentifier`
- `RestoreTime`
- `Engine`
- `LicenseModel`

有关更多信息，请参阅 将 Amazon RDS 的数据库实例还原到指定时间。

Amazon RDS for Db2 实例类

数据库实例的计算和内存容量由其实例类决定。您需要的数据库实例类取决于您的处理能力和内存要求。

支持的 RDS for Db2 实例类

支持的 Amazon RDS for Db2 实例类是 Amazon RDS 数据库实例类的子集。有关 Amazon RDS 实例类的完整列表，请参阅数据库实例类。

主题

- Db2 Standard Edition 支持的 RDS for Db2 实例类
- Db2 Advanced Edition 支持的 RDS for Db2 实例类

Db2 Standard Edition 支持的 RDS for Db2 实例类

下表列出了 Db2 数据库版本 11.5.9.0 的 Db2 Standard Edition 支持的所有实例类。这些实例类同时适用于自带许可（BYOL）和通过 Amazon Web Services Marketplace 获得的 Db2 许可证。

实例类类型	实例类
具有第三代 Intel Xeon Scalable 处理器、SSD 存储和网络优化的通用实例类	db.m6idn.large–db.m6idn.8xlarge
由第三代 Intel Xeon Scalable 处理器提供支持的通用实例类	db.m6in.large–db.m6in.8xlarge
通用实例类	db.m7i.large–db.m7i.8xlarge db.m6i.large–db.m6i.8xlarge
使用基于本地 NVMe 的 SSD 的内存优化型实例类，由第三代 Intel Xeon Scalable 处理器提供支持	db.x2iedn.xlarge
由第三代 Intel Xeon Scalable 处理器提供支持的内存优化型实例类	db.r6idn.large–db.r6idn.4xlarge db.r6in.large–db.r6in.4xlarge
内存优化型实例类	db.r7i.large–db.r7i.8xlarge db.r6i.large–db.r6i.4xlarge
可突增性能实例类	db.t3.small–db.t3.2xlarge

Db2 Advanced Edition 支持的 RDS for Db2 实例类

下表列出了 Db2 数据库版本 11.5.9.0 的 Db2 Advanced Edition 支持的所有实例类。这些实例类同时适用于自带许可（BYOL）和通过 Amazon Web Services Marketplace 获得的 Db2 许可证。

实例类类型	实例类
具有第三代 Intel Xeon Scalable 处理器、SSD 存储和网络优化的通用实例类	db.m6idn.large–db.m6idn.32xlarge
由第三代 Intel Xeon Scalable 处理器提供支持的通用实例类	db.m6in.large–db.m6in.32xlarge
通用实例类	db.m6i.large–db.m7i.48xlarge

实例类类型	实例类
	db.m7i.large–db.m7i.48xlarge
使用基于本地 NVMe 的 SSD 的内存优化型实例类，由第三代 Intel Xeon Scalable 处理器提供支持	db.x2iedn.xlarge–db.x2iedn.32xlarge
由第三代 Intel Xeon Scalable 处理器提供支持的内存优化型实例类	db.r6idn.large–db.r6idn.32xlarge db.r6in.8xlarge–db.r6in.32xlarge
内存优化型实例类	db.r6i.large–db.r7i.48xlarge db.r7i.large–db.r7i.48xlarge
可突增性能实例类	db.t3.small–db.t3.2xlarge

Amazon RDS for Db2 默认角色

RDS for Db2 添加了以下六个角色，并使用 `ADMIN` 选项向 `master_user_role` 授予这些角色。预置数据库后，RDS for Db2 会向主用户授予 `master_user_role`。然后，主用户可以通过连接到数据库，使用原生 `GRANT` 语句向其它用户、组或角色授予这些角色。

- DBA – RDS for Db2 通过 `DATAACCESS` 授权创建此空角色。主用户可以向该角色添加更多授权或权限，然后向其它用户、组或角色授予该角色。
- DBA_RESTRICTED – RDS for Db2 创建这一空角色。主用户可以向该角色添加权限，然后向其它用户、组或角色授予该角色。
- DEVELOPER – RDS for Db2 通过 `DATAACCESS` 授权创建此空角色。主用户可以向该角色添加更多授权或权限，然后向其它用户、组或角色授予该角色。
- ROLE_NULLID_PACKAGES – 对于在运行 `CREATE DATABASE` 时由 Db2 绑定的 `ALL NULLID` 包，RDS for Db2 向此角色授予对这些包的 `EXECUTE` 权限。
- ROLE_PROCEDURES – RDS for Db2 向该角色授予对所有 `SYSIBM` 过程的 `EXECUTE` 权限。
- ROLE_TABLESPACES – RDS for Db2 授予对由 `CREATE DATABASE` 命令创建的表空间的 `USAGE` 权限。

Amazon RDS for Db2 参数

Amazon RDS for Db2 使用三种类型的参数：数据库管理器配置参数、注册表变量和数据库配置参数。可以通过参数组管理前两种类型，并通过 rdsadmin.update_db_param 存储过程管理最后一种类型。

默认情况下，RDS for Db2 数据库实例使用特定于 Db2 数据库和数据库实例的数据库参数组。此参数组包含 IBM Db2 数据库引擎的参数，特别是数据库管理器配置参数和注册表变量。有关使用参数的更多信息，请参阅Amazon RDS 的参数组。

RDS for Db2 数据库配置参数设置为您已选择的存储引擎的默认值。有关 Db2 参数的更多信息，请参阅 IBM Db2 文档中的 Db2 database configuration parameters。

主题

查看参数组中的参数

数据库管理器配置参数和注册表变量在参数组中设置。可以通过使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API，来查看特定 Db2 版本的数据库管理器配置参数和注册表变量。

控制台

查看数据库参数组的参数值

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择参数组。

   数据库参数组将显示在列表中。
3. 选择参数组名称以查看其参数列表。

Amazon CLI

可以通过运行 describe-engine-default-parameters 命令，来查看 Db2 版本的数据库管理器配置参数和注册表变量。为 `--db-parameter-group-family` 选项指定下列值之一：

- `db2-ae-11.5`
- `db2-se-11.5`

例如，要查看 Db2 Standard Edition 11.5 的参数，请运行以下命令：

```
aws rds describe-engine-default-parameters --db-parameter-group-family db2-se-11.5
```

此命令生成类似于以下示例的输出：

```
{
    "EngineDefaults": {
        "Parameters": [
            {
                "ParameterName": "agent_stack_sz",
                "ParameterValue": "1024",
                "Description": "You can use this parameter to determine the amount of
 memory that is allocated by Db2 for each agent thread stack.",
                "Source": "engine-default",
                "ApplyType": "static",
                "DataType": "integer",
                "AllowedValues": "256-32768",
                "IsModifiable": false
            },
            {
                "ParameterName": "agentpri",
                "ParameterValue": "-1",
                "Description": "This parameter controls the priority given to all
 agents and to other database manager instance processes and threads by the operating
 system scheduler. This priority determines how CPU time is allocated to the database
 manager processes, agents, and threads relative to other processes and threads running
 on the machine.",
                "Source": "engine-default",
                "ApplyType": "static",
                "DataType": "integer",
                "AllowedValues": "1-99",
                "IsModifiable": false
```

```
            },
            ...
        ]
    }
}
```

要仅列出 Db2 Standard Edition 11.5 的可修改参数，请运行以下命令：

对于 Linux、macOS 或 Unix：

```
aws rds describe-engine-default-parameters \
    --db-parameter-group-family db2-se-11.5 \
    --query 'EngineDefaults.Parameters[?IsModifiable==`true`].
{ParameterName:ParameterName, DefaultValue:ParameterValue}'
```

对于 Windows：

```
aws rds describe-engine-default-parameters ^
    --db-parameter-group-family db2-se-11.5 ^
    --query 'EngineDefaults.Parameters[?IsModifiable==`true`].
{ParameterName:ParameterName, DefaultValue:ParameterValue}'
```

RDS API

要查看数据库参数组的参数值，可执行带下列必需参数的 `DescribeDBParameters` 操作。

- `DBParameterGroupName`

使用 Db2 命令查看所有参数

可以通过使用 Db2 命令来查看数据库管理器配置参数、数据库配置参数和注册表变量的设置。

查看设置

1. 连接到 Db2 数据库。在以下示例中，用您自己的信息替换 *database_name*、*master_username* 和 *master_password*。

   ```
   db2 "connect to database_name user master_username using master_password"
   ```

2. 查找支持的 Db2 版本。

```
db2 "select service_level, fixpack_num from table(sysproc.env_get_inst_info()) as
 instanceinfo"
```

3. 查看特定 Db2 版本的参数。

   - 通过运行以下命令，来查看数据库管理器配置参数：

     ```
     db2 "select cast(substr(name,1,24) as varchar(24)) as name, case
         when value_flags = 'NONE' then '' else value_flags end flags,
         cast(substr(value,1,64) as varchar(64)) as current_value
         from sysibmadm.dbmcfg
         order by name asc with UR"
     ```

   - 通过运行以下命令，来查看所有数据库配置参数：

     ```
     db2 "select cast(substr(name,1,24) as varchar(24)) as name, case
         when value_flags = 'NONE' then '' else value_flags end flags,
         cast(substr(value,1,64) as varchar(64)) as current_value
         from table(db_get_cfg(null)) order by name asc, member asc with UR"
     ```

   - 通过运行以下命令，来查看当前设置的注册表变量：

     ```
     db2 "select cast(substr(reg_var_name,1,50) as varchar(50)) as reg_var_name,
         cast(substr(reg_var_value,1,50) as varchar(50)) as reg_var_value,
         level from table(env_get_reg_variables(null))
         order by reg_var_name,member with UR"
     ```

修改参数组中的参数

可以通过使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API，来修改自定义参数组中的数据库管理器配置参数和注册表变量。有关更多信息，请参阅 Amazon RDS 数据库实例的数据库参数组。

控制台

修改数据库管理器配置参数和注册表变量

1. 创建自定义参数组。有关创建数据库参数组的详细信息，请参阅 在 Amazon RDS 中创建数据库参数组。

2. 修改该自定义参数组中的参数。有关修改参数组的更多信息，请参阅在 Amazon RDS 中修改数据库参数组中的参数。

Amazon CLI

修改数据库管理器配置参数和注册表变量

1. 通过运行 create-db-parameter-group 命令创建自定义参数组。

   包括以下必需选项：

   • `--db-parameter-group-name` - 您正在创建的参数组的名称。
   • `--db-parameter-group-family` – Db2 引擎版本和主要版本。有效值：`db2-se-11.5`、`db2-ae-11.5`。
   • `--description` – 此参数组的描述。

   有关创建数据库参数组的详细信息，请参阅 在 Amazon RDS 中创建数据库参数组。
2. 通过运行 modify-db-parameter-group 命令修改您创建的自定义参数组中的参数。

   包括以下必需选项：

   • `--db-parameter-group-name` – 您创建的参数组的名称。
   • `--parameters` – 参数名称、值和参数更新应用方法的数组。

   有关修改参数组的更多信息，请参阅在 Amazon RDS 中修改数据库参数组中的参数。

RDS API

修改数据库管理器配置参数和注册表变量

1. 使用 CreateDBParameterGroup 操作创建自定义数据库参数组。

   包括以下必需参数：

   • `DBParameterGroupName`
   • `DBParameterGroupFamily`
   • `Description`

有关创建数据库参数组的详细信息，请参阅 在 Amazon RDS 中创建数据库参数组。

2. 使用 ModifyDBParameterGroup 操作修改您创建的自定义参数组中的参数。

   包括以下必需参数：

   - `DBParameterGroupName`
   - `Parameters`

   有关修改参数组的更多信息，请参阅在 Amazon RDS 中修改数据库参数组中的参数。

使用 Db2 命令修改数据库配置参数

可以使用 Db2 命令修改数据库配置参数。

修改数据库配置参数

1. 连接到 `rdsadmin` 数据库。在以下示例中，用您的信息替换 *master_username* 和 *master_password*。

```
db2 "connect to rdsadmin user master_username using master_password"
```

2. 通过调用 `rdsadmin.update_db_param` 存储过程来更改数据库配置参数。有关更多信息，请参阅 rdsadmin.update_db_param。

```
db2 "call rdsadmin.update_db_param(
    'database_name',
    'parameter_to_modify',
    'changed_value')"
```

Amazon RDS 上 Db2 数据库的 EBCDIC 排序规则

Amazon RDS for Db2 支持 Db2 数据库的 EBCDIC 排序规则。只有在使用 Amazon RDS the section called “rdsadmin.create_database” 存储过程创建数据库时，您才能为该数据库指定 EBCDIC 排序规则序列。

使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 创建 RDS for Db2 数据库实例时，可以指定数据库名称。如果您指定数据库名称，Amazon RDS 会创建一个默认排序规则为

SYSTEM 的数据库。如果您需要使用 EBCDIC 排序规则创建数据库，则在创建数据库实例时不要指定数据库名称。

RDS for Db2 中数据库的排序规则是在创建时设置的，并且不可更改。

使用 EBCDIC 排序规则创建 Db2 数据库

1. 如果您没有 RDS for Db2 数据库实例，请创建数据库实例但不要指定数据库名称。可以使用 Amazon Web Services Management Console、Amazon CLI、或 RDS API 创建数据库实例。有关更多信息，请参阅 创建数据库实例。
2. 通过调用 `rdsadmin.create_database` 存储过程创建 Db2 数据库并将排序规则选项设置为 EBCDIC 值。有关更多信息，请参阅 rdsadmin.create_database。

> **Important**
>
> 使用此存储过程创建数据库后，您将无法更改排序规则序列。如果您希望数据库使用不同的排序规则序列，请通过调用 the section called “rdsadmin.drop_database” 存储过程来删除该数据库。然后，使用所需的排序规则序列创建一个数据库。

Amazon RDS for Db2 数据库实例的本地时区

默认情况下，已设置运行 Db2 的 Amazon RDS 数据库实例的时区。默认值为协调世界时 (UTC)。要匹配应用程序的时区，您可改为将数据库实例的时区设置为本地时区。

首次创建数据库实例时将设置时区。您可以使用 Amazon Web Services Management Console、RDS API 或 Amazon CLI 创建数据库实例。有关更多信息，请参阅 创建数据库实例。

如果数据库实例是多可用区部署的一部分，则当它进行失效转移时，其时区将保持为您设置的本地时区。

您可以将数据库实例还原到您指定的某个时间点。时间以您的本地时区显示。有关更多信息，请参阅 将 Amazon RDS 的数据库实例还原到指定时间。

在数据库实例上设置本地时区具有以下限制：

- 您无法修改现有 Amazon RDS for Db2 数据库实例的时区。
- 您无法将快照从一个时区的数据库实例还原至另一个时区的数据库实例。
- 强烈建议您不要将备份文件从一个时区还原到另一个时区。如果您将备份文件从一个时区还原至为另一个时区，则必须审查您的查询和应用程序，来查看时区更改的效果。

可用时区

您可以将以下值用于时区设置。

区	时区
非洲	非洲/开罗、非洲/卡萨布兰卡、非洲/哈拉雷、非洲/拉各斯、非洲/罗安达、非洲/蒙罗维亚、非洲/内罗毕、非洲/的黎波里、非洲/温得和克
美洲	美洲/阿拉瓜伊纳、美洲/阿根廷/布宜诺斯艾利斯、美洲/亚森松、美洲/Bogota、美洲/加拉加斯、美洲/芝加哥、美洲/奇瓦瓦、美洲/库亚巴、美洲/丹佛、美洲/底特律、美洲/福塔莱萨、美洲/戈特霍布、美洲/危地马拉、美洲/哈利法克斯、美洲/利马、美洲/洛杉矶、美洲/玛瑙斯、美洲/马塔莫罗斯、美洲/墨西哥城、美洲/蒙特雷、美洲/蒙特维多、美洲/纽约、美洲/菲尼克斯、美洲/圣地亚哥、美洲/圣保罗、美洲/蒂华纳、美洲/多伦多
亚洲	亚洲/阿曼、亚洲/阿什哈巴德、亚洲/巴格达、亚洲/巴库、亚洲/曼谷、亚洲/贝鲁特、亚洲/加尔各答、亚洲/大马士革、亚洲/达卡、亚洲/香港、亚洲/伊尔库茨克、亚洲/雅加达、亚洲/耶路撒冷、亚洲/喀布尔、亚洲/卡拉奇、亚洲/加德满都、亚洲/加尔各答、亚洲/克拉斯诺亚尔斯克、亚洲/马加丹、亚洲/马尼拉、亚洲/马斯喀特、亚洲/新西伯利亚、亚洲/阳光、亚洲/利雅得、亚洲/首尔、亚洲/上海、亚洲/新加坡、亚洲/台北、亚洲/德黑兰、亚洲/东京、亚洲/乌兰巴托、亚洲/符拉迪沃斯托克、亚洲/雅库茨克、亚洲/埃里温
大西洋	大西洋/亚速尔、大西洋/佛得角
澳大利亚	澳大利亚/阿德莱德、澳大利亚/布里斯班、澳大利亚/达尔文、澳大利亚/尤克拉、澳大利亚/霍巴特、澳大利亚/豪勋爵岛、澳大利亚/珀斯、澳大利亚/悉尼
巴西	巴西/迪诺罗尼亚、巴西/东部
加拿大	加拿大/纽芬兰、加拿大/萨斯喀彻温省
ETC	ETC/GMT-3
欧洲	欧洲/阿姆斯特丹、欧洲/雅典、欧洲/柏林、欧洲/都柏林、欧洲/赫尔辛基、欧洲/加里宁格勒、欧洲/伦敦、欧洲/马德里、欧洲/莫斯科、欧洲/巴黎、欧洲/布拉格、欧洲/罗马、欧洲/萨拉热窝、欧洲/斯德哥尔摩

区	时区
太平洋	太平洋/阿皮亚、太平洋/奥克兰、太平洋/查塔姆、太平洋/斐济、太平洋/关岛、太平洋/檀香山、太平洋/圣诞岛、太平洋/玛贵斯、太平洋/萨摩亚、太平洋/汤加塔布岛、太平洋/韦克
US	美国/阿拉斯加、美国/中部、美国/印第安纳东部、美国/西部、美国/太平洋地区
UTC	UTC

创建 Amazon RDS for Db2 数据库实例的先决条件

以下各项是创建数据库实例之前的先决条件。

主题

管理员账户

创建数据库实例时，您必须为实例指定管理员账户。Amazon RDS 向该本地数据库管理员账户授予 `DBADM` 权限。

管理员账户具有以下特点、功能和限制：

- 是本地用户而不是 Amazon Web Services 账户。
- 没有 Db2 实例级权限，例如 `SYSADM`、`SYSMAINT` 或 `SYSCTRL`。
- 无法停止或启动 Db2 实例。
- 如果您在创建数据库实例时指定了名称，则无法删除 Db2 数据库。
- 拥有对 Db2 数据库（包括目录表和视图）的完全访问权限。
- 可以使用 Amazon RDS 存储过程创建本地用户和组。
- 可以授予和撤销权限和特权。

管理员账户可以执行以下任务：

- 创建、修改或删除数据库实例。
- 创建数据库快照。
- 启动时间点还原。
- 创建数据库快照的自动备份。
- 创建数据库快照的手动备份。
- 使用其它 Amazon RDS 功能。

额外注意事项

在创建数据库实例之前，请考虑以下事项：

- 每个 Amazon RDS for Db2 数据库实例可以托管多达 50 个 Db2 数据库。有关更多信息，请参阅 Amazon RDS for Db2 数据库实例上的多个数据库。
- Initial database name (初始数据库名称)
  - 如果您在创建数据库实例时没有提供数据库名称，则 Amazon RDS 不会创建数据库。
  - 在以下情况下不要提供数据库名称：
    - 您想修改 `db2_compatibility_vector` 参数。有关更多信息，请参阅 设置 db2_compatibility_vector 参数。
- 在自带许可（BYOL）模型中，必须先创建一个包含 IBM Customer ID 和 IBM Site ID 的自定义参数组。有关更多信息，请参阅 自带 Db2 许可。
- 在采用“通过 Amazon Web Services Marketplace 获得 Db2 许可证”模式时，您需要对要使用的特定 IBM Db2 版本进行有效 Amazon Web Services Marketplace 订阅。如果您还没有订阅，请在 Amazon Web Services Marketplace 中为要使用的 IBM Db2 版本订阅 Db2。有关更多信息，请参阅 通过 Amazon Web Services Marketplace 获得 Db2 许可证。

Amazon RDS for Db2 数据库实例上的多个数据库

通过调用 rdsadmin.create_database 存储过程，可以在单个 RDS for Db2 数据库实例上创建多个数据库。单个 RDS for Db2 数据库实例上限制为最多创建 50 个数据库，其中包括处于激活和停用状态的数据库。

> **Note**
>
> 如果您要在 2024 年 11 月 15 日之前创建的 RDS for Db2 数据库实例上创建多个数据库，则必须重启该数据库实例才能支持多个数据库。

默认情况下，Amazon RDS 会在您创建数据库时将其激活。要优化内存资源，可以停用不经常使用的数据库，然后在需要时将其激活。有关更多信息，请参阅the section called “停用数据库” 和the section called “激活数据库”。

数据库实例上已激活数据库的数量取决于服务器上的可用内存资源。内存资源因数据库实例类和为数据库配置的内存量而异。有关数据库实例类的信息，请参阅 the section called “数据库实例类”。有关如何更新 RDS for Db2 数据库内存的信息，请参阅 the section called “rdsadmin.update_db_param”。

建议您选择具有 2 GB 内存的数据库实例类，用于执行常见的数据库任务、满足操作系统要求以及执行其他 Amazon RDS 自动化任务（如备份）。有关数据库实例类的更多信息，请参阅the section called “修改数据库实例”。

此外，IBM 还建议每个活动数据库至少有 1 GB 的内存。有关更多信息，请参阅 IBM 文档中的 Disk and memory requirements。

可以使用以下公式来计算一个数据库实例能够拥有的最大活动数据库数量：

```
Active database limit = (total server memory - 2 GB) / 1 GB
```

以下示例显示了具有 db.m6i.xlarge 数据库实例类的数据库实例的最大活动数据库数量：

```
Active database limit = (total server memory - 2 GB) / 1 GB
                      = (16 GB - 2 GB) / 1 GB
                      = 14 databases
```

当 Amazon RDS 在崩溃后恢复数据库时，如果该数据库之前处于活动状态，它会激活该数据库。在某些情况下（例如将数据库实例类修改为具有较低的内存配置时），可能没有足够的内存来激活该数据库实例上的所有数据库。在这种情况下，Amazon RDS 会按数据库的创建顺序激活这些数据库。

> **Note**
>
> Amazon RDS 因内存不足而无法激活的所有数据库都将处于停用状态。

连接到 Db2 数据库实例

在 Amazon RDS 预调配 Amazon RDS for Db2 数据库实例后，您可以使用任何标准 SQL 客户端应用程序连接到该数据库实例。由于 Amazon RDS 是一项托管式服务，因此您无法以 SYSADM、SYSCTRL、SECADM 或 SYSMAINT 身份登录。

您可以使用 IBM Db2 CLP、IBM CLPPlus、DBeaver 或 IBM Db2 Data Management Console 连接到运行 IBM Db2 数据库引擎的数据库实例。

> **Note**
>
> 如果您的 RDS for Db2 数据库实例内存不足，则与 Db2 数据库的连接可能会失败。有关更多信息，请参阅 the section called "数据库连接错误"。

主题

查找 Amazon RDS for Db2 数据库实例的端点

每个 Amazon RDS 数据库实例都有一个终端节点，每个终端节点都包含数据库实例的 DNS 名称和端口号。要使用 SQL 客户端应用程序连接到 Amazon RDS for Db2 数据库实例，您需要该数据库实例的 DNS 名称和端口号。

可以使用 Amazon Web Services Management Console或 Amazon CLI 查找数据库实例的端点。

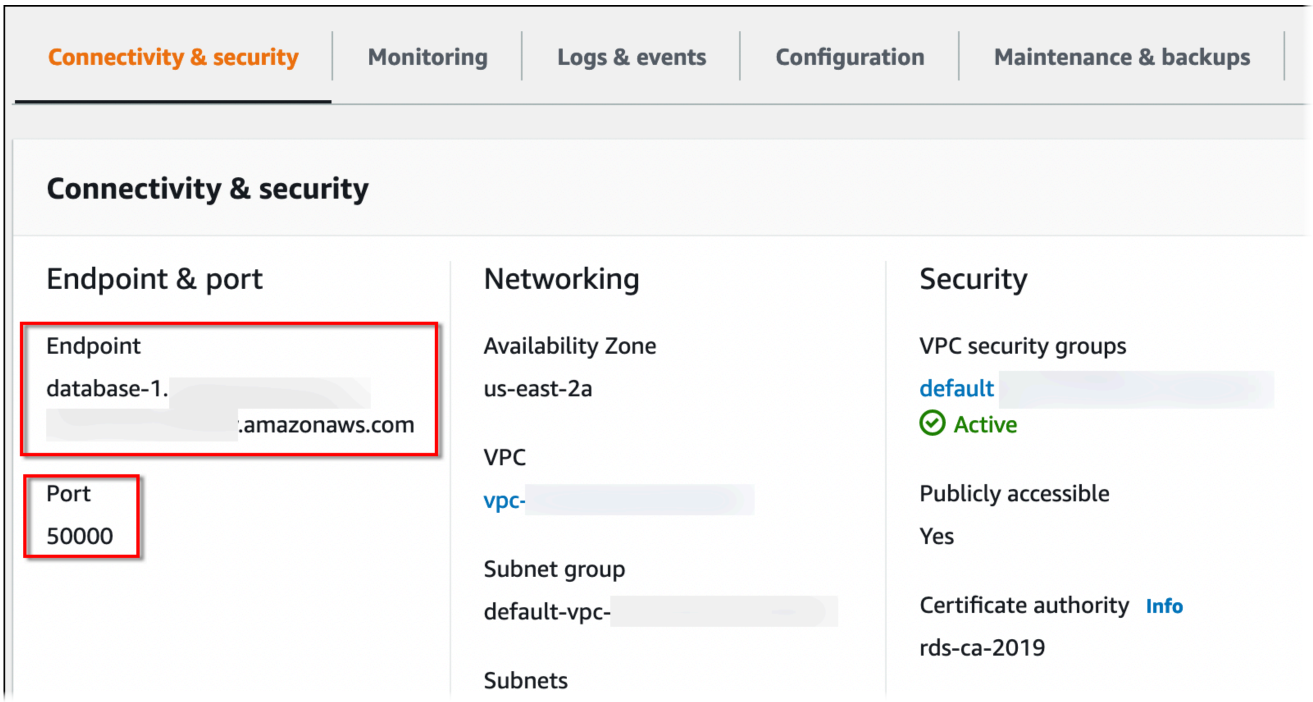
控制台

查找 RDS for Db2 数据库实例的端点

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。

2. 在控制台的右上角，选择数据库实例的 Amazon Web Services 区域。
3. 找到 RDS for Db2 数据库实例的 DNS 名称和端口号。

   a. 选择 Databases (数据库) 以显示您的数据库实例的列表。
   b. 选择 RDS for Db2 数据库实例名称以显示实例详细信息。
   c. 在连接和安全性选项卡上，复制端点。另请注意端口号。您需要端点和端口号才能连接到数据库实例。



Amazon CLI

要查找 RDS for Db2 数据库实例的端点，请运行 [describe-db-instances](https://docs.aws.amazon.com/cli/latest/reference/rds/describe-db-instances.html) 命令。在以下示例中，将 *database-1* 替换为数据库实例的名称。

对于 Linux、macOS 或 Unix：

```
aws rds describe-db-instances \
    --db-instance-identifier database-1 \
    --query 'DBInstances[].
{DBInstanceIdentifier:DBInstanceIdentifier,DBName:DBName,Endpoint:Endpoint}' \
    --output json
```

对于 Windows：

```
aws rds describe-db-instances ^
    --db-instance-identifier database-1 ^
    --query 'DBInstances[].
{DBInstanceIdentifier:DBInstanceIdentifier,DBName:DBName,Endpoint:Endpoint}' ^
    --output json
```

此命令生成类似于以下示例的输出。输出中的 `Address` 行包含 DNS 名称。

```
[
    {
        "DBInstanceIdentifier": "database-1",
        "DBName": "DB2DB",
        "Endpoint": {
            "Address": "database-1.123456789012.us-east-2.amazonaws.com",
            "Port": 50000,
            "HostedZoneId": "Z2OC4A7DETW6VH"
        }
    }
]
```

使用 IBM Db2 CLP 连接到 Amazon RDS for Db2 数据库实例

您可以使用命令行实用程序（如 IBM Db2 CLP）连接到 Amazon RDS for Db2 数据库实例。此实用程序是 IBM Data Server Runtime Client 的一部分。要从 IBM Fix Central 中下载客户端，请参阅 IBM 支持中的 IBM Data Server Client Packages Version 11.5 Mod 8 Fix Pack 0。

主题

术语

以下术语有助于解释连接到 RDS for Db2 数据库实例时使用的命令。

catalog tcpip node

此命令将远程数据库节点注册到本地 Db2 客户端，这样客户端应用程序就可以访问该节点。要对节点进行编目，您需要提供诸如服务器的主机名、端口号和通信协议之类的信息。然后，编目的节点表示一个或多个远程数据库所在的目标服务器。有关更多信息，请参阅 IBM Db2 文档中的 CATALOG TCPIP/TCPIP4/TCPIP6 NODE 命令。

catalog database

此命令将远程数据库注册到本地 Db2 客户端，这样客户端应用程序就可以访问该数据库。要对数据库进行编目，您需要提供诸如数据库别名、数据库所在的节点以及连接数据库所需的身份验证类型之类的信息。有关更多信息，请参阅 IBM Db2 文档中的 CATALOG DATABASE 命令。

安装客户端

downloading the package for Linux之后，使用根或管理员权限安装客户端。

> **Note**
>
> 要在 AIX 或 Windows 上安装客户端，请按照相同的步骤操作，但要修改适用于您的操作系统的命令。

在 Linux 上安装客户端

1. 运行 **`./db2_install -f sysreq`** 并选择 **`yes`** 以接受许可证。
2. 选择安装客户端的位置。
3. 运行 **`clientInstallDir/instance/db2icrt -s client`** ***`instance_name`***。将 *`instance_name`* 替换为 Linux 上有效的操作系统用户。在 Linux 中，Db2 数据库实例名称与操作系统的用户名相关联。

   此命令在 Linux 上指定用户的主目录中创建 **`sqllib`** 目录。

连接数据库实例

要连接到 RDS for Db2 数据库实例，您需要该实例的 DNS 名称和端口号。有关查找它们的信息，请参阅查找终端节点。您还需要知道您在创建 RDS for Db2 数据库实例时定义的数据库名称、主用户名和主密码。有关查找它们的更多信息，请参阅创建数据库实例。

使用 IBM Db2 CLP 连接到 RDS for Db2 数据库实例

1. 使用您在 IBM Db2 CLP 客户端安装期间指定的用户名登录。
2. 对 RDS for Db2 数据库实例进行编目。在以下示例中，将 *node_name*、*dns_name* 和 *port* 替换为本地目录中节点的名称、数据库实例的 DNS 名称以及端口号。

   ```
   db2 catalog TCPIP node node_name remote dns_name server port
   ```

   示例

   ```
   db2 catalog TCPIP node remnode remote database-1.123456789012.us-
   east-1.amazonaws.com server 50000
   ```

3. 对 `rdsadmin` 数据库和您的数据库进行编目。这将允许您连接到 `rdsadmin` 数据库以使用 Amazon RDS 存储过程执行一些管理任务。有关更多信息，请参阅 管理您的 RDS for Db2 数据库实例。

   在以下示例中，将 *database_alias*、*node_name* 和 *database_name* 替换为该数据库的别名、在上一步中定义的节点的名称以及数据库的名称。`server_encrypt` 通过网络加密您的用户名和密码。

   ```
   db2 catalog database rdsadmin [ as database_alias ] at node node_name
    authentication server_encrypt

   db2 catalog database database_name [ as database_alias ] at node node_name
    authentication server_encrypt
   ```

   示例

   ```
   db2 catalog database rdsadmin at node remnode authentication server_encrypt

   db2 catalog database testdb as rdsdb2 at node remnode authentication server_encrypt
   ```

4. 连接到您的 RDS for Db2 数据库。在以下示例中，将 *rds_database_alias*、*master_username* 和 *master_password* 替换为 RDS for Db2 数据库实例的数据库名称、主用户名和主密码。

   ```
   db2 connect to rds_database_alias user master_username using master_password
   ```

此命令生成类似于以下示例的输出：

```
Database Connection Information

    Database server        = DB2/LINUXX8664 11.5.9.0
    SQL authorization ID   = ADMIN
    Local database alias   = TESTDB
```

5. 运行查询并查看结果。以下示例显示了一条 SQL 语句，该语句选择了您创建的数据库。

```
db2 "select current server from sysibm.dual"
```

此命令生成类似于以下示例的输出：

```
1
    ------------------
    TESTDB

    1 record(s) selected.
```

排除与 RDS for Db2 数据库实例的连接故障

如果您收到以下 NULLID 错误，则通常表示您的客户端和 RDS for Db2 服务器版本不匹配。有关支持的 Db2 客户端版本，请参阅 IBM Db2 文档中的 Supported combinations of clients, drivers and server levels。

```
db2 "select * from syscat.tables"
SQL0805N Package "NULLID.SQLC2029 0X4141414141454A69" was not found.
SQLSTATE=51002
```

收到此错误后，必须将旧版 Db2 客户端中的软件包绑定到 RDS for Db2 支持的 Db2 服务器版本。

将旧版 Db2 客户端中的软件包绑定到较新的 Db2 服务器

1. 在客户端计算机上找到绑定文件。通常，这些文件位于 Db2 客户端安装路径的 bnd 目录中，扩展名为 .bnd。
2. 连接到 Db2 服务器。在以下示例中，将 *database_name* 替换为 Db2 服务器的名称。用您的信息替换 *master_username* 和 *master_password*。此用户拥有 DBADM 权限。

```
db2 connect to database_name user master_username using master_password
```

3. 运行 bind 命令以绑定这些软件包。

   a. 导航到客户端计算机上存在绑定文件的目录。

   b. 对每个文件运行 bind 命令。

      以下选项为必填：

      - `blocking all` – 在单个数据库请求中绑定该绑定文件中的所有软件包。
      - `grant public` – 授予 public 执行软件包的权限。
      - `sqlerror continue` – 指定即使发生错误，bind 过程仍会继续。

      有关 bind 命令的更多信息，请参阅 IBM Db2 文档中的 BIND 命令。

4. 通过查询 `syscat.package` 目录视图或检查 bind 命令后返回的消息，验证绑定是否成功。

有关更多信息，请参阅 IBM 支持中的 DB2 v11.5 Bind File and Package Name List。

使用 IBM CLPPlus 连接到 Amazon RDS for Db2 数据库实例

您可以使用诸如 IBM CLPPlus 之类的实用程序连接到 Amazon RDS for Db2 数据库实例。此实用程序是 IBM Data Server Runtime Client 的一部分。要从 IBM Fix Central 中下载客户端，请参阅 IBM 支持中的 IBM Data Server Client Packages Version 11.5 Mod 8 Fix Pack 0。

> **Important**
>
> 我们建议您在支持图形用户界面以及桌面的操作系统上运行 IBM CLPPlus，例如 macOS、Windows 或 Linux。如果运行无头 Linux，请将开关 -nw 与 CLPPlus 命令结合使用。

主题

- 安装客户端
- 连接数据库实例

安装客户端

下载 Linux 的软件包后，安装客户端。

> **Note**
> 要在 AIX 或 Windows 上安装客户端，请按照相同的步骤操作，但要修改适用于您的操作系统的命令。

在 Linux 上安装客户端

1. 运行 **./db2_install**。
2. 运行 **clientInstallDir/instance/db2icrt -s client *instance_name***。将 *instance_name* 替换为 Linux 上有效的操作系统用户。在 Linux 中，Db2 数据库实例名称与操作系统的用户名相关联。

   此命令在 Linux 上指定用户的主目录中创建 **sqllib** 目录。

连接数据库实例

要连接到 RDS for Db2 数据库实例，您需要该实例的 DNS 名称和端口号。有关查找它们的信息，请参阅查找终端节点。您还需要知道您在创建 RDS for Db2 数据库实例时定义的数据库名称、主用户名和主密码。有关查找它们的更多信息，请参阅创建数据库实例。

使用 IBM CLPPlus 连接到 RDS for Db2 数据库实例

1. 查看命令语法。在以下示例中，将 *clientDir* 替换为安装客户端的位置。

   ```
   cd clientDir/bin
       ./clpplus -h
   ```

2. 配置您的 Db2 服务器。在以下示例中，将 *dsn_name*、*database_name*、*endpoint* 和 *port* 替换为 RDS for Db2 数据库实例的 DSN 名称、数据库名称、端点和端口。有关更多信息，请参阅查找 Amazon RDS for Db2 数据库实例的端点。

   ```
   db2cli writecfg add -dsn dsn_name -database database_name -host endpoint -port port
    -parameter "Authentication=SERVER_ENCRYPT"
   ```

3. 连接到 RDS for Db2 数据库实例。在以下示例中，将 *master_username* 和 *dsn_name* 替换为主用户名和 DSN 名称。

```
./clpplus -nw master_username@dsn_name
```

4. 此时会打开一个 Java Shell 窗口。输入 RDS for Db2 数据库实例的主密码。

> **Note**
>
> 如果 Java Shell 窗口未打开，请运行 **./clpplus -nw** 以使用相同的命令行窗口。

```
Enter password: *********
```

此时会建立连接并生成类似于以下示例的输出：

```

Database Connection Information :
---------------------------------
Hostname = database-1.abcdefghij.us-east-1.rds.amazonaws.com
Database server = DB2/LINUXX8664  SQL110590
SQL authorization ID = admin
Local database alias = DB2DB
Port = 50000
```

5. 运行查询并查看结果。以下示例显示了一条 SQL 语句，该语句选择了您创建的数据库。

```
SQL > select current server from sysibm.dual;
```

此命令生成类似于以下示例的输出：

```
1
    --------------------
    DB2DB
    SQL>
```

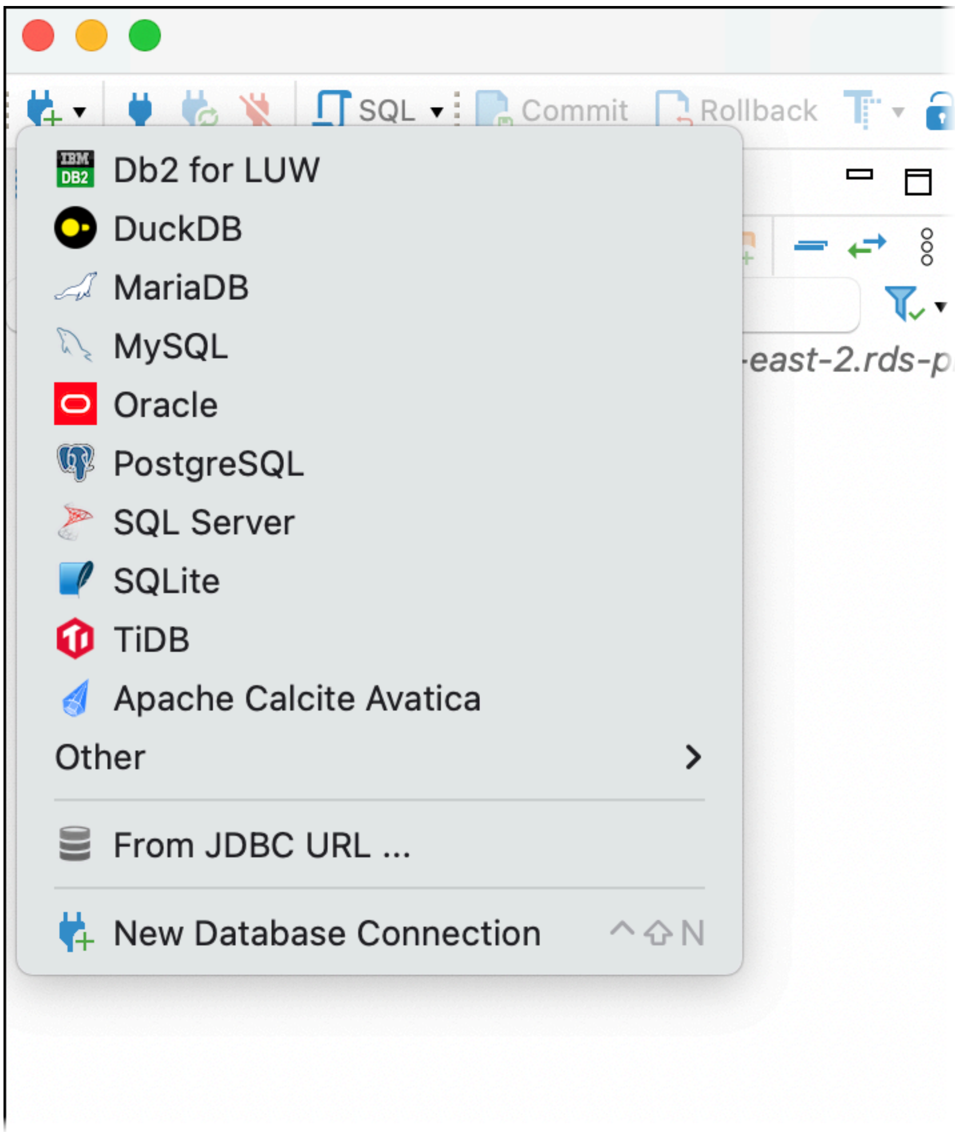
使用 DBeaver 连接到 Amazon RDS for Db2 数据库实例

您可以使用第三方工具（例如 DBeaver）连接到 Amazon RDS for Db2 数据库实例。要下载此实用程序，请参阅 DBeaver 社区。

要连接到 RDS for Db2 数据库实例，您需要该实例的 DNS 名称和端口号。有关查找它们的信息，请参阅查找终端节点。您还需要知道您在创建 RDS for Db2 数据库实例时定义的数据库名称、主用户名和主密码。有关查找它们的更多信息，请参阅创建数据库实例。

使用 DBeaver 连接到 RDS for Db2 数据库实例

1. 启动 DBeaver。
2. 在工具栏中选择新建连接图标，然后选择 Db2 for LUW。



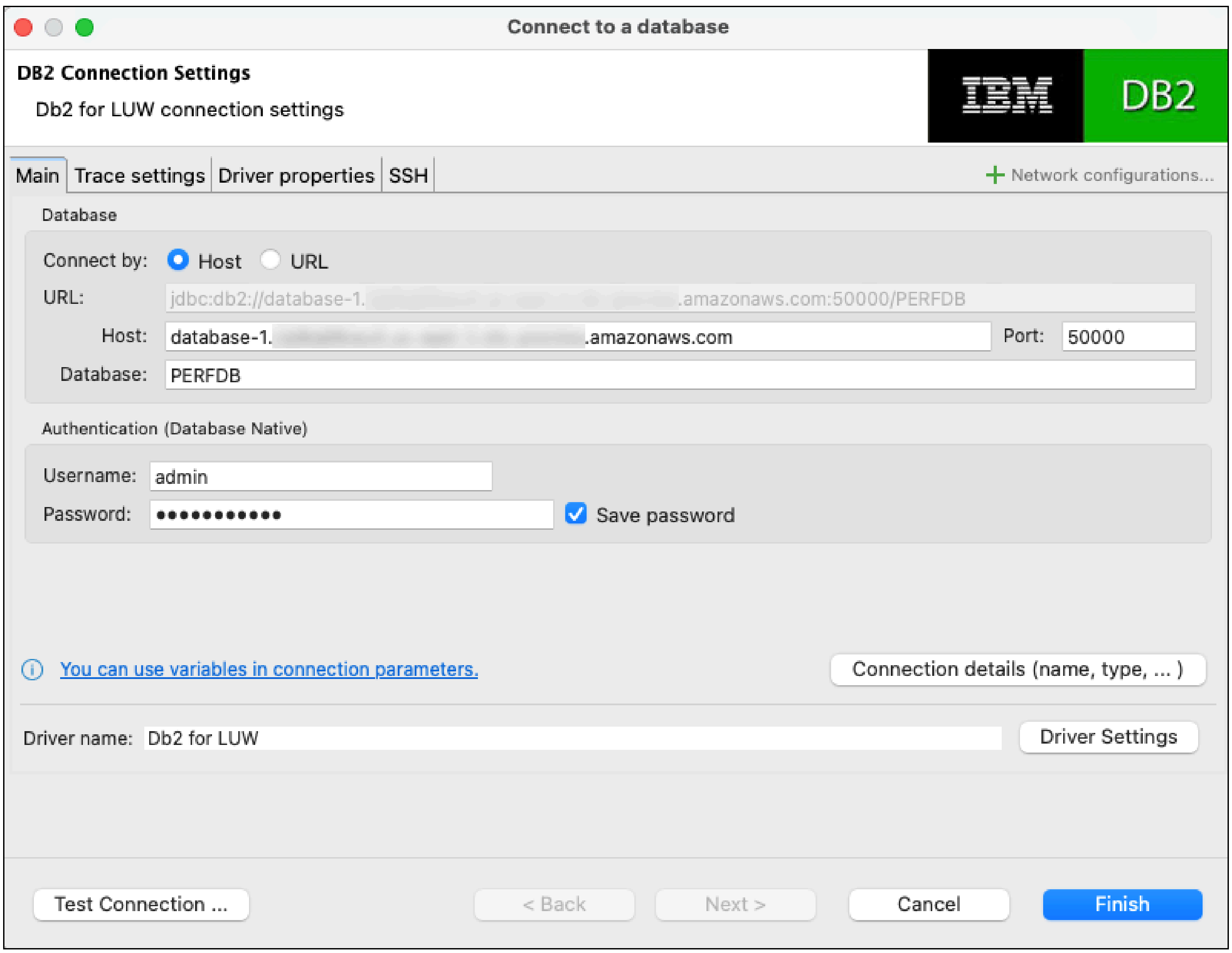
3. 在连接到数据库窗口中，提供 RDS for Db2 数据库实例的信息。

   a. 输入以下信息：

      - 对于主机，输入数据库实例的 DNS 名称。
      - 对于端口，输入数据库实例的端口号。
      - 对于数据库，输入数据库的名称。
      - 对于 Username (用户名)，输入数据库实例的数据库管理员的名称。
      - 对于密码，输入数据库实例的数据库管理员的密码。

   b. 选择保存密码。

   c. 选择驱动程序设置。



4. 在编辑驱动程序窗口中，指定其它安全属性。

a. 选择驱动程序属性选项卡。

b. 添加两个用户属性。

   i. 打开上下文（右键单击）菜单，然后选择添加新属性。

   ii. 对于属性名称，添加 encryptionAlgorithm，然后选择确定。

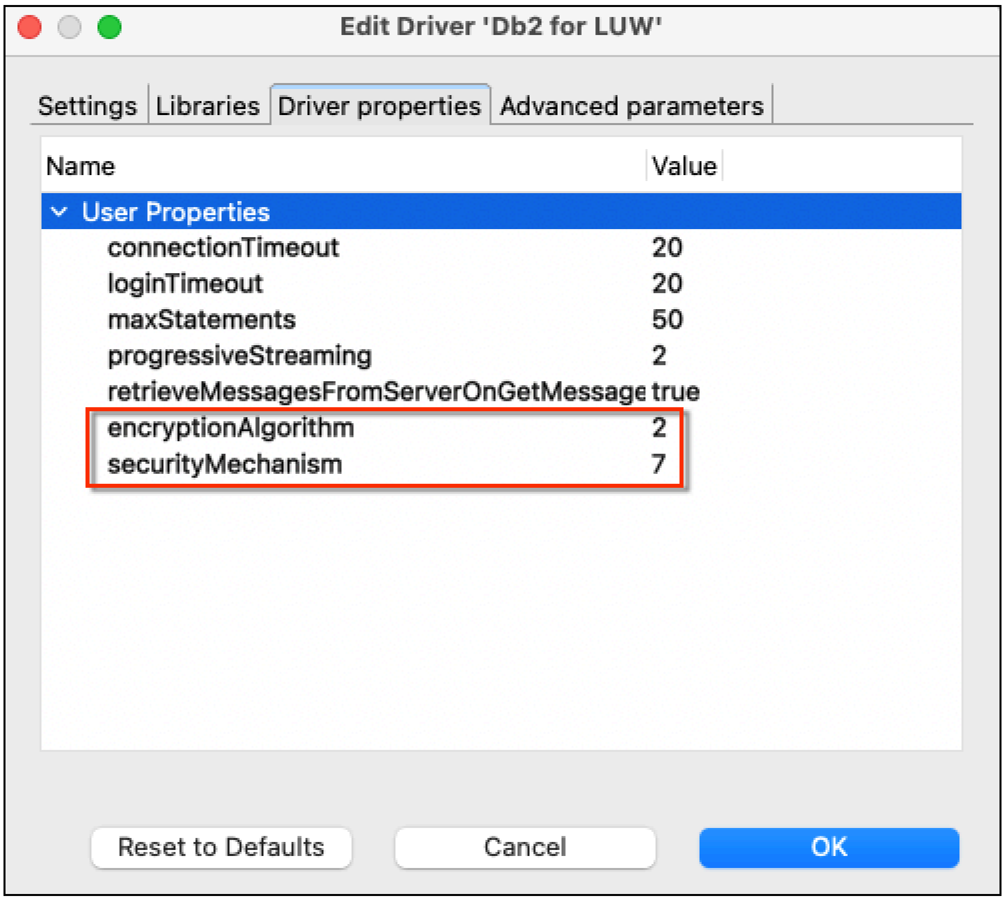
   iii. 选择 encryptionAlgorithm 行后，选择值列并添加 2。

   iv. 打开上下文（右键单击）菜单，然后选择添加新属性。

   v. 对于属性名称，添加 securityMechanism，然后选择确定。

   vi. 选择 securityMechanism 行后，选择值列并添加 7。

c. 选择 确定。



5. 在连接到数据库窗口中，选择测试连接。如果您的计算机上没有安装 DB2 JBDC 驱动程序，则该驱动程序会自动下载。

6. 选择 确定。

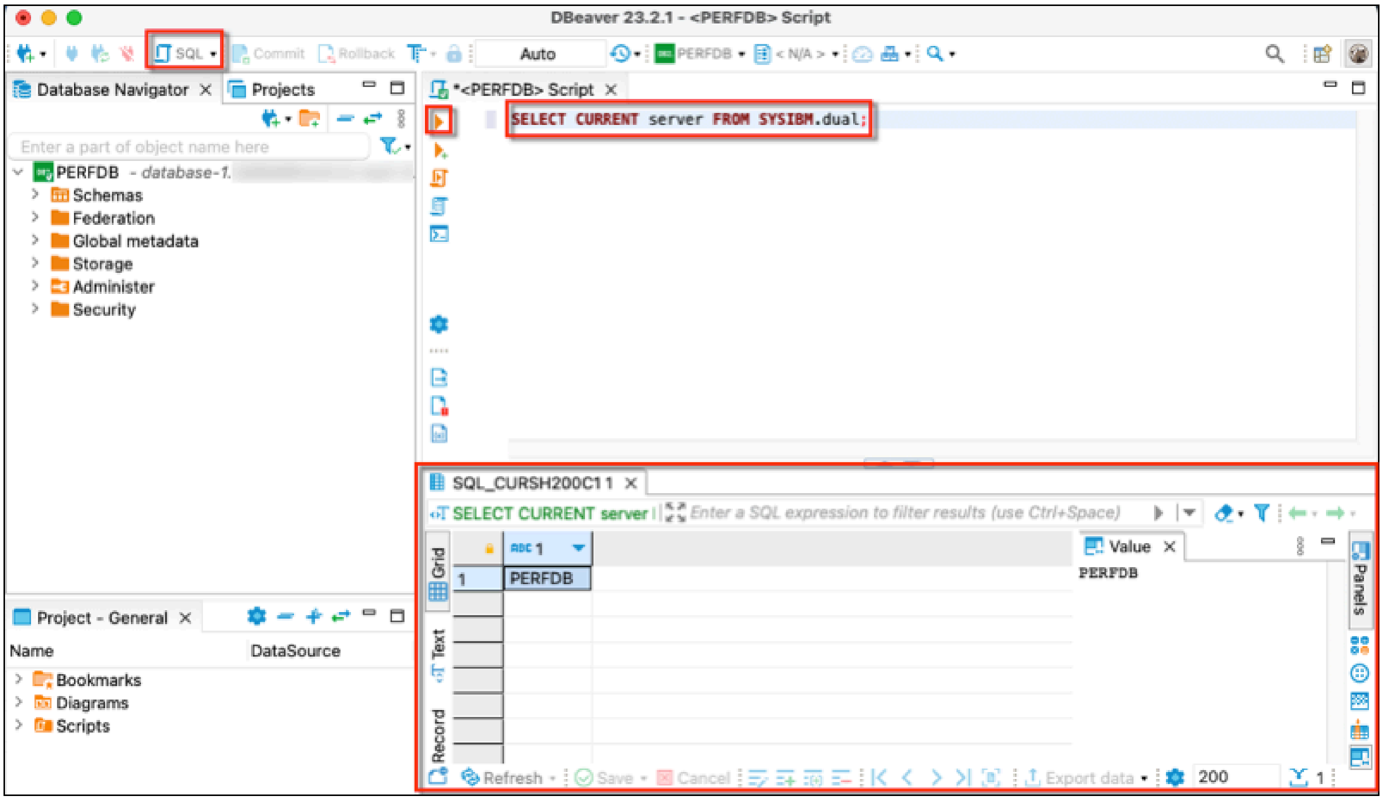
7. 选择完成。

8. 在数据库导航选项卡中，选择数据库的名称。您现在可以探索对象。

您现在可以运行 SQL 命令了。

运行 SQL 命令并查看结果

1. 在顶部菜单中，选择 SQL。这将打开 SQL 脚本面板。
2. 在脚本面板中，输入 SQL 命令。
3. 要运行该命令，请选择执行 SQL 查询按钮。
4. 在 SQL 结果面板中，查看 SQL 查询的结果。



使用 IBM Db2 Data Management Console 连接到 Amazon RDS for Db2 数据库实例

您可以使用 IBM Db2 Data Management Console 连接到 Amazon RDS for Db2 数据库实例。IBM Db2 Data Management Console 可以管理和监控多个 RDS for Db2 数据库实例。

> **Note**
>
> 您拥有的 Amazon EC2 Linux 或 Windows 计算机必须与 RDS for Db2 数据库实例位于同一 VPC 和安全组上。VPC 和安全组通过内部网络控制与数据库实例的连接。

IBM Db2 Data Management Console 需要存储库 Db2 数据库来存储元数据和性能指标，但无法自动为 RDS for Db2 创建存储库数据库。相反，必须先创建一个存储库数据库来监控一个或多个 RDS for Db2 数据库实例。然后，可以安装 IBM Db2 Data Management Console 并使用 IBM Db2 Data Management Console 连接到 RDS for Db2 数据库实例。

主题

步骤 1：创建存储库数据库来监控数据库实例

您可以使用现有大小合适的 RDS for Db2 数据库实例作为 IBM Db2 Data Management Console 的存储库，以监控其它 RDS for Db2 数据库实例。但是，由于管理员用户没有创建缓冲池和表空间的 SYSCTRL 权限，因此，使用 IBM Db2 Data Management Console 存储库创建操作来创建存储库数据库会失败。相反，必须创建一个存储库数据库。此存储库数据库监控 RDS for Db2 数据库实例。

可以通过两种不同的方式创建存储库数据库。可以创建 RDS for Db2 数据库，然后手动创建缓冲池、用户表空间和系统临时表空间。或者，可以创建一个单独的 Amazon EC2 实例来托管 IBM Db2 Data Management Console 存储库数据库。

主题

手动创建缓冲池、用户表空间和系统临时表空间

创建缓冲池、用户表空间和系统临时表空间

1. 连接到 `rdsadmin` 数据库。在以下示例中，用您自己的信息替换 *master_username* 和 *master_password*。

   ```
   db2 connect to rdsadmin user master_username using master_password
   ```

2. 为 IBM Db2 Data Management Console 创建缓冲池。在以下示例中，将 *database_name* 替换为您为 IBM Db2 Data Management Console 创建的用于监控 RDS for Db2 数据库实例的存储库名称。

```
db2 "call rdsadmin.create_bufferpool('database_name',
     'BP4CONSOLE', 1000, 'Y', 'Y', 32768)"
```

3. 为 IBM Db2 Data Management Console 创建用户表空间。在以下示例中，将 *database_name* 替换为您为 IBM Db2 Data Management Console 创建的用于监控 RDS for Db2 数据库实例的存储库名称。

```
db2 "call rdsadmin.create_tablespace('database_name',
     'TS4CONSOLE', 'BP4CONSOLE', 32768)"
```

4. 为 IBM Db2 Data Management Console 创建系统临时表空间。在以下示例中，将 *database_name* 替换为您为 IBM Db2 Data Management Console 创建的用于监控 RDS for Db2 数据库实例的存储库名称。

```
db2 "call rdsadmin.create_tablespace('database_name',
    'TS4CONSOLE_TEMP', 'BP4CONSOLE', 32768, 0, 0, 'S')"
```

现在您已准备就绪，可以安装 IBM Db2 Data Management Console 了。有关设置和安装的更多信息，请参阅[步骤 2：安装和设置 IBM Db2 Data Management Console]。

创建 Amazon EC2 实例来托管 IBM Db2 Data Management Console 存储库

您可以创建 Amazon Elastic Compute Cloud（Amazon EC2）实例来托管 IBM Db2 Data Management Console 存储库。有关创建 Amazon EC2 实例的信息，请参阅《Amazon EC2 用户指南》中的[教程：Amazon EC2 Linux 实例入门]。

步骤 2：安装和设置 IBM Db2 Data Management Console

创建缓冲池、用户表空间和系统临时表空间后，您已准备就绪，可以安装和设置 IBM Db2 Data Management Console 了。

> **Important**
>
> 您拥有的 Amazon EC2 Linux 或 Windows 计算机必须与 RDS for Db2 数据库实例位于同一 VPC 和安全组上。VPC 和安全组通过内部网络控制与数据库实例的连接。此外，您必须已经为 IBM Db2 Data Management Console 创建了存储库数据库。

安装和设置 IBM Db2 Data Management Console

1. 在 IBM 支持网站上，从 IBM Db2 Data Management Console Version 3.1x releases 下载 IBM Db2 Data Management Console。
2. 安装 IBM Db2 Data Management Console。
3. 打开 IBM Db2 Data Management Console，并使用 Amazon EC2 计算机的 IP 地址，以及用于与 Amazon EC2 实例进行 HTTP 或 HTTPS 连接的端口号。例如，使用 `http://xx.xx.xx.xx:11080` 或 `https://xx.xx.xx.xx.11081`。将 `xx.xx.xx.xx` 替换为 Amazon EC2 计算机的 IP 地址。`11080` 和 `11081` 是用于 HTTP 和 HTTPS 连接的默认端口。
4. （可选）如果您想在 Amazon EC2 实例上使用端口 80 或 443，可以使用 Apache httpd 或 Nginx HTTP 服务器将 IBM Db2 Data Management Console 端口代理到端口 80 或 443。有关更多信息，请参阅 Apache HTTP Server Project 和 the nginx website。

   要支持连接到 IBM Db2 Data Management Console，必须编辑安全组中的入站规则。如果您使用代理，请更改 TCP/IP 端口 80 或 443 以重定向到 IBM Db2 Data Management Console 端口。如果您未使用代理，请将 TCP/IP 端口 80 或 443 更改为默认端口 11080（HTTP）或 11081（HTTPS）。

现在，您已准备就绪，可以登录 IBM Db2 Data Management Console 来配置存储库数据库并连接到 RDS for Db2 数据库实例。有关更多信息，请参阅 配置存储库数据库并连接到数据库实例。

步骤 3：配置存储库数据库并连接到 RDS for Db2 数据库实例

当您首次连接到存储库数据库时，IBM Db2 Data Management Console 会自动配置存储库。配置存储库数据库后，可以向 IBM Db2 Data Management Console 添加数据库连接。

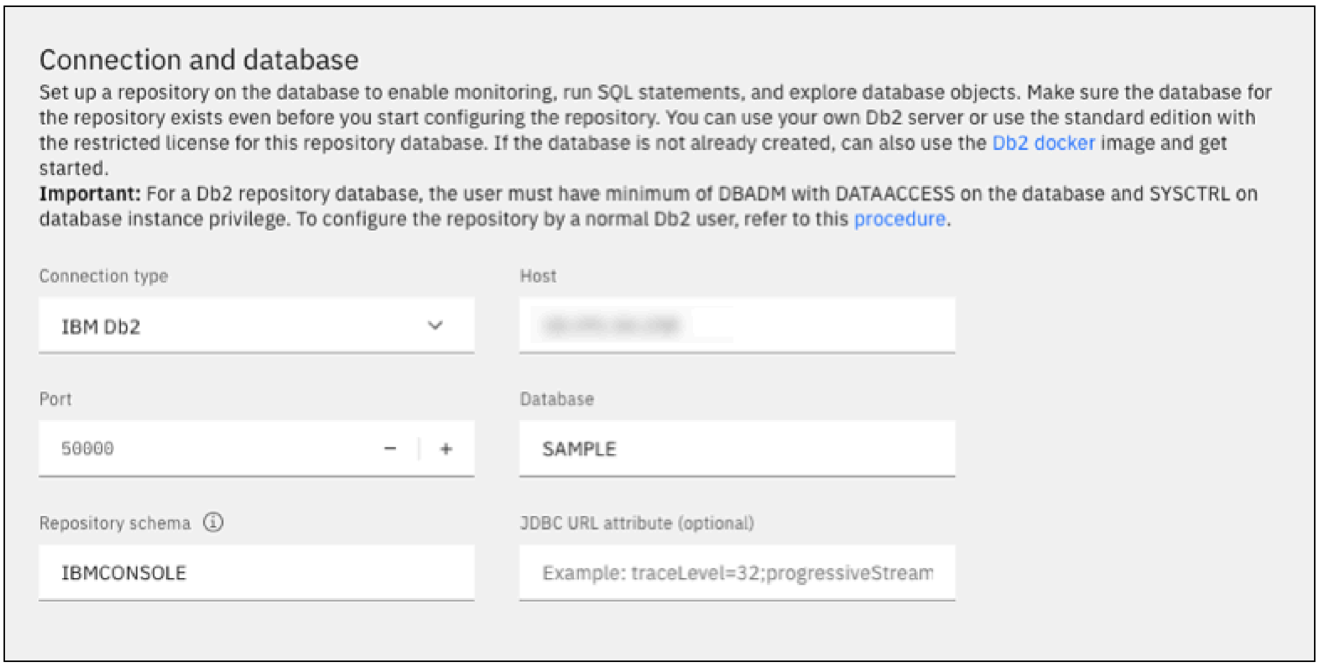
要连接到 RDS for Db2 数据库实例，您需要该实例的 DNS 名称和端口号。有关查找它们的信息，请参阅查找终端节点。您还需要知道您在创建 RDS for Db2 数据库实例时定义的数据库名称、主用户名和主密码。有关查找它们的更多信息，请参阅创建数据库实例。如果您通过互联网进行连接，请允许流量进入数据库端口。有关更多信息，请参阅 创建数据库实例。

使用 IBM Db2 Data Management Console 连接到 RDS for Db2 数据库实例

1. 使用您在安装期间设置的凭证登录 IBM Db2 Data Management Console。
2. 配置存储库。

   a. 在连接和数据库部分，输入您的 RDS for Db2 数据库实例的以下信息：

      - 对于主机，输入数据库实例的 DNS 名称。
      - 对于端口，输入数据库实例的端口号。
      - 对于数据库，输入数据库的名称。



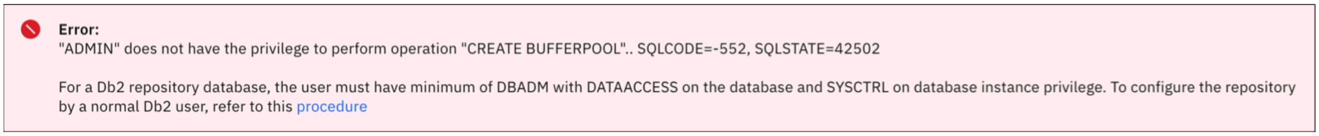
   b. 在安全和凭证部分，输入您的 RDS for Db2 数据库实例的以下信息：

      - 对于安全类型，选择加密的用户和密码。
      - 对于 Username (用户名)，输入数据库实例的数据库管理员的名称。
      - 对于密码，输入数据库实例的数据库管理员的密码。

   c. 选择 Test connection (测试连接)。

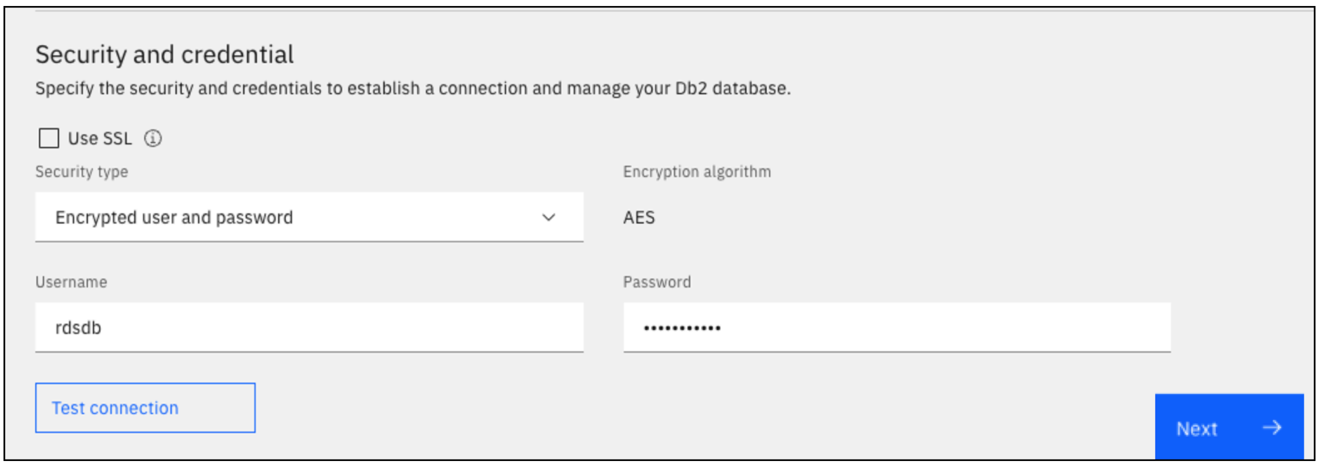
> **Note**
>
> 如果连接不成功，请通过安全组中的入站规则确认数据库端口处于打开状态。有关更多信息，请参阅 与 Amazon RDS for Db2 有关的安全组的注意事项。

如果您没有在 RDS for Db2 中手动创建缓冲池、用户表空间和系统临时表空间，则可能会看到以下错误消息：



确保为 IBM Db2 Data Management Console 存储库创建了缓冲区表、表空间和对象，以监控 RDS for Db2 数据库实例。或者，可以使用 Amazon EC2 Db2 数据库实例来托管 IBM Db2 Data Management Console 存储库，以监控 RDS for Db2 数据库实例。有关更多信息，请参阅 步骤 1：创建存储库数据库来监控数据库实例。

d. 成功测试连接后，选择下一步。



如果 IBM Db2 Data Management Console 在 RDS for Db2 数据库实例中找到了缓冲池、用户表空间和系统临时表空间，则 IBM Db2 Data Management Console 会自动配置存储库数据库。如果您将 Amazon EC2 实例上的 Db2 实例用作存储库数据库，则 IBM Db2 Data Management Console 会自动创建缓冲池和其它对象。

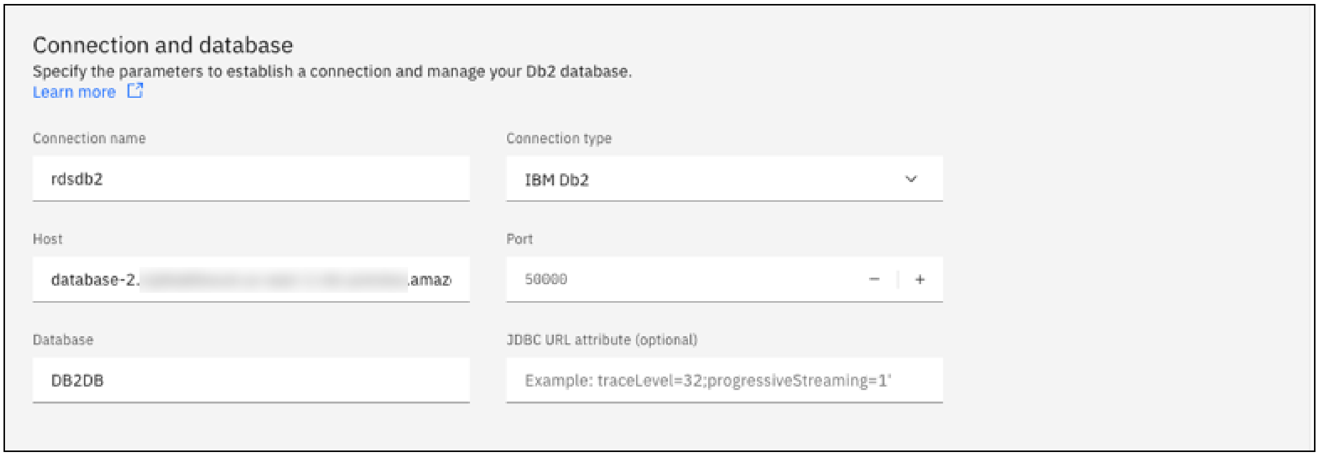
3. 在设置统计数据事件监视器选择加入窗口中，选择下一步。

4. （可选）添加新连接。如果您想使用不同的 RDS for Db2 数据库实例进行管理和监控，请添加与非存储库 RDS for Db2 数据库实例的连接。

   a. 在连接和数据库部分，输入用于进行管理和监控的 RDS for Db2 数据库实例的以下信息：

      - 对于连接名称，输入 Db2 数据库标识符。
      - 对于主机，输入数据库实例的 DNS 名称。
      - 对于端口，输入数据库实例的端口号。

- 对于数据库，输入数据库的名称。



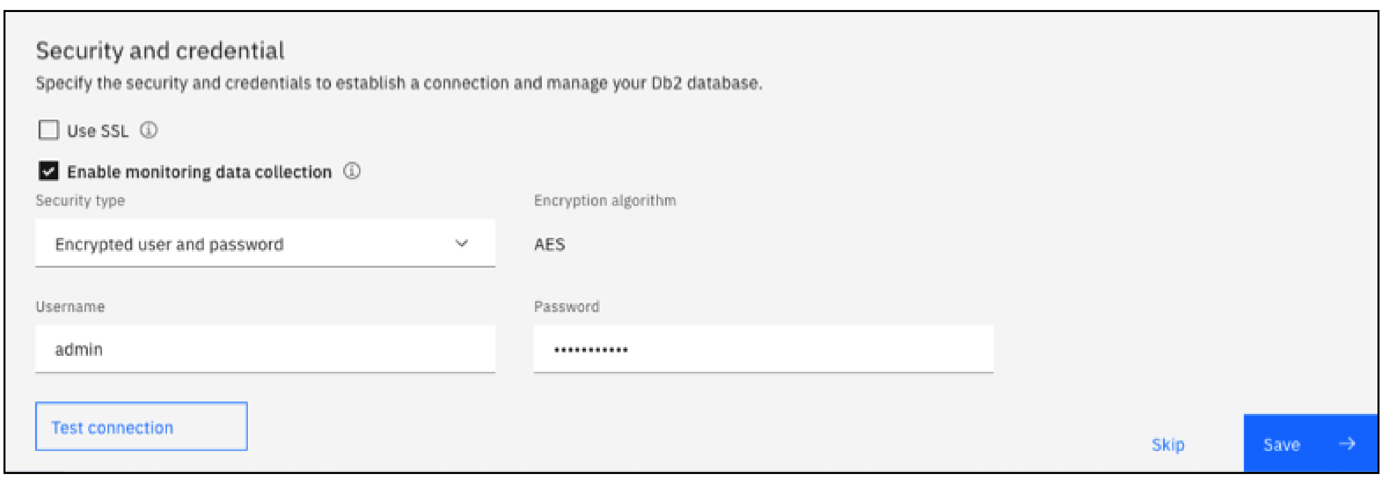
b. 在安全和凭证部分，选择启用监控数据收集。

c. 输入 RDS for Db2 数据库实例的以下信息：

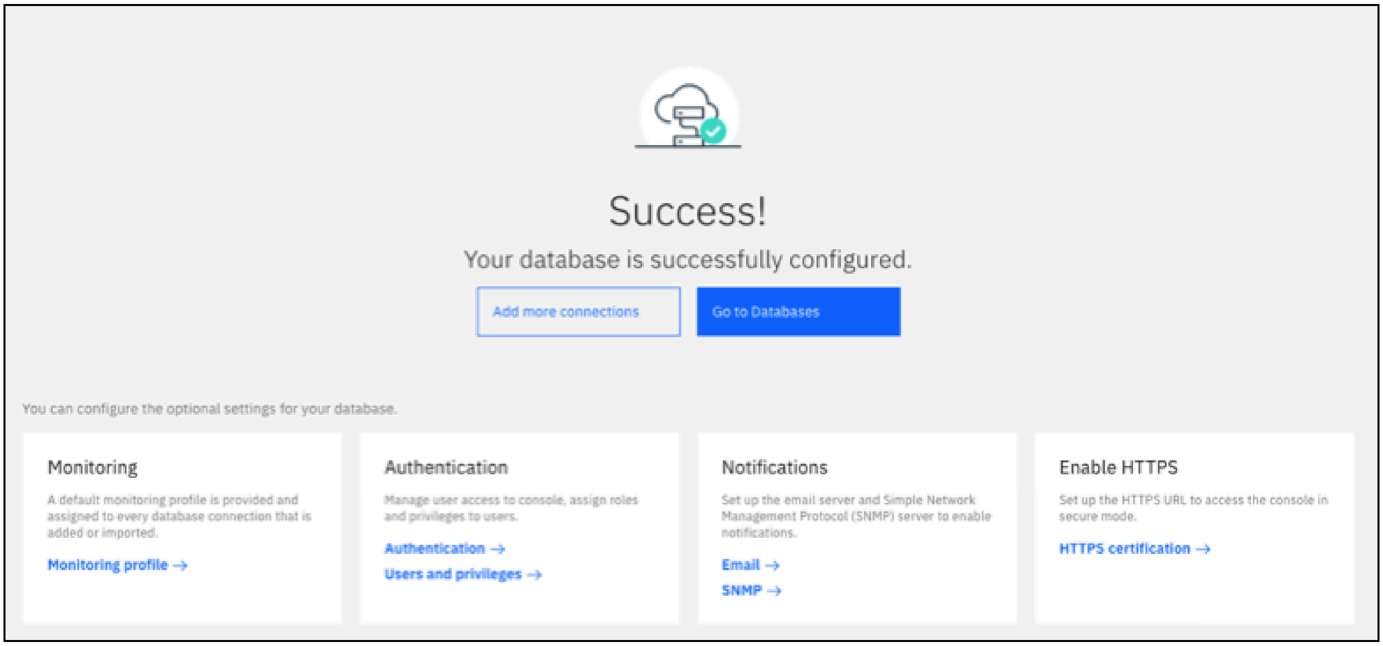
- 对于 Username (用户名)，输入数据库实例的数据库管理员的名称。
- 对于密码，输入数据库实例的数据库管理员的密码。

d. 选择 Test connection (测试连接)。

e. 成功测试连接后，选择保存。



添加连接后，将出现一个类似于以下内容的窗口。此窗口表示您的数据库已成功配置。



5. 选择转到数据库。将显示类似以下内容的“数据库”窗口。此窗口是一个显示指标、状态和连接的控制面板。

现在您就可以开始使用 IBM Db2 Data Management Console。

使用 IBM Db2 Data Management Console

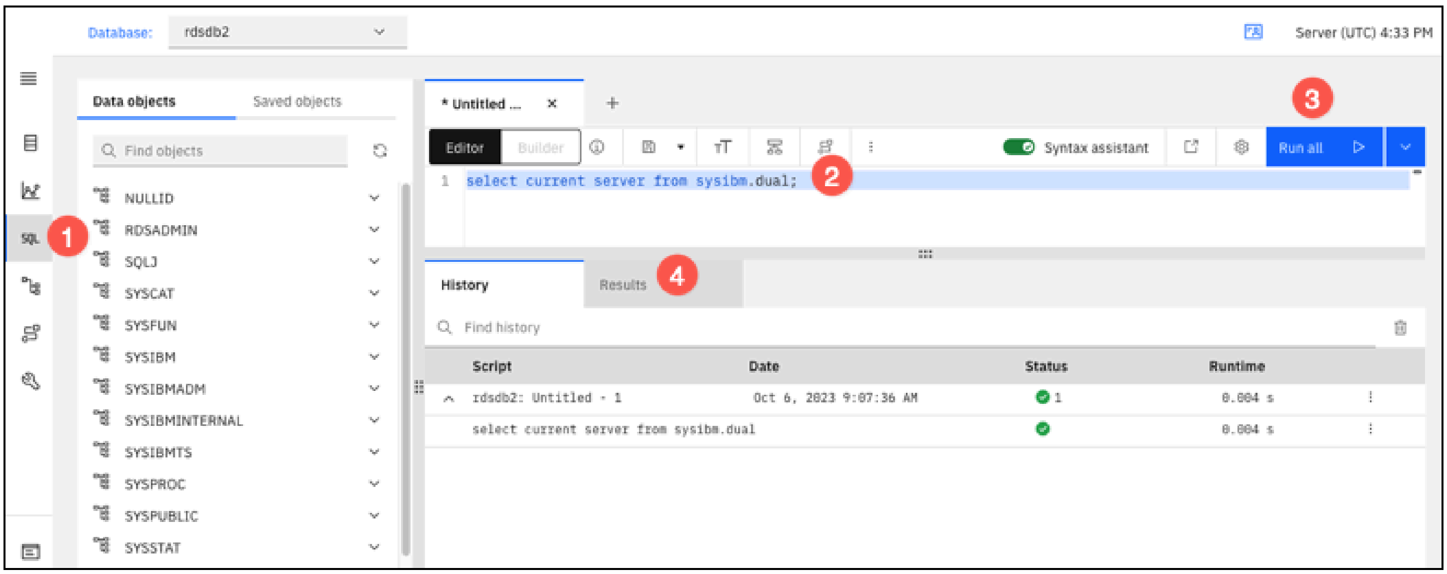
可以使用 IBM Db2 Data Management Console 执行以下类型的任务：

- 管理多个 RDS for Db2 数据库实例。
- 运行 SQL 命令。
- 浏览、创建或更改数据和数据库对象。
- 在 SQL 中创建 `EXPLAIN PLAN` 语句。

- 优化查询。

运行 SQL 命令并查看结果

1. 在左侧导航栏中，选择 SQL。
2. 输入 SQL 命令。
3. 选择全部运行。
4. 要查看结果，请选择结果选项卡。



与 Amazon RDS for Db2 有关的安全组的注意事项

要连接到 Amazon RDS for Db2 数据库实例，它必须与包含必要的 IP 地址和网络配置的安全组关联。您的 RDS for Db2 数据库实例可能使用默认安全组。如果在创建 RDS for Db2 数据库实例时分配了默认的未配置安全组，则防火墙将阻止互联网连接。有关创建新安全组的信息，请参阅 使用安全组控制访问权限。

在创建新安全组后，您应修改数据库实例以使其与该安全组关联。有关更多信息，请参阅"修改 Amazon RDS 数据库实例"。

可通过使用 SSL 加密与数据库实例的连接来增强安全性。有关更多信息，请参阅 对 Amazon RDS for Db2 数据库实例使用 SSL/TLS。

保护 Amazon RDS for Db2 数据库实例连接

Amazon RDS for Db2 支持旨在提高 RDS for Db2 数据库实例安全性的方法。

主题

- 对 Amazon RDS for Db2 数据库实例使用 SSL/TLS
- 将 Kerberos 身份验证用于 Amazon RDS for Db2

对 Amazon RDS for Db2 数据库实例使用 SSL/TLS

SSL 是用于保护客户端和服务器之间的网络连接的行业标准协议。在 SSL 3.0 版之后，此名称已更改为 TLS，但我们通常仍将此协议称为 SSL。Amazon RDS 支持对 Amazon RDS for Db2 数据库实例进行 SSL 加密。使用 SSL/TLS，您可以加密应用程序客户端和 RDS for Db2 数据库实例之间的连接。SSL/TLS 支持在 RDS for Db2 的所有 Amazon Web Services 区域中都可用。

要为 RDS for Db2 数据库实例启用 SSL/TLS 加密，请将 Db2 SSL 选项添加到与该数据库实例关联的参数组中。根据 Db2 的要求，Amazon RDS 使用另一个端口进行 SSL/TLS 连接。这样使数据库实例与 Db2 客户端之间能够同时进行明文通信和 SSL 加密通信。例如，您可以使用用于明文通信的端口与 VPC 内的其他资源通信，同时使用用于 SSL 加密通信的端口与 VPC 外部的资源通信。

主题

- 创建 SSL/TLS 连接
- 连接到您的 Db2 数据库服务器

创建 SSL/TLS 连接

要创建 SSL/TLS 连接，请选择证书颁发机构（CA），下载适用于所有 Amazon Web Services 区域的证书捆绑包，然后向自定义参数组添加参数。

步骤 1：选择 CA 并下载证书

选择证书颁发机构（CA），然后为所有 Amazon Web Services 区域下载证书捆绑包。有关更多信息，请参阅 使用 SSL/TLS 加密与数据库实例或集群的连接。

步骤 2：更新自定义参数组中的参数

> **Important**
>
> 如果您对 RDS for Db2 使用自带许可（BYOL）模型，请修改您为 IBM Customer ID 和 IBM Site ID 创建的自定义参数组。如果您对 RDS for Db2 使用不同的许可模式，请按照此过程将参数添加到自定义参数组。有关更多信息，请参阅 Amazon RDS for Db2 许可选项。

您无法修改 RDS for Db2 数据库实例的默认参数组。因此，您必须创建一个自定义参数组，对其进行修改，然后将其附加到 RDS for Db2 数据库实例。有关参数组的信息，请参阅 Amazon RDS 数据库实例的数据库参数组。

使用下表中的参数设置。

参数	值
`DB2COMM`	`TCPIP,SSL` 或 `SSL`
`SSL_SVCENAME`	`<any port number except the number used for the non-SSL port>`

更新自定义参数组中的参数

1. 通过运行 create-db-parameter-group 命令创建自定义参数组。

   包括以下必需选项：

   - `--db-parameter-group-name` - 您正在创建的参数组的名称。
   - `--db-parameter-group-family` – Db2 引擎版本和主要版本。有效值：`db2-se-11-5`、`db2-ae-11.5`。
   - `--description` – 此参数组的描述。

   有关创建数据库参数组的详细信息，请参阅 在 Amazon RDS 中创建数据库参数组。
2. 通过运行 modify-db-parameter-group 命令修改您创建的自定义参数组中的参数。

   包括以下必需选项：

- `--db-parameter-group-name` – 您创建的参数组的名称。
- `--parameters` – 参数名称、值和参数更新应用方法的数组。

有关修改参数组的更多信息，请参阅在 Amazon RDS 中修改数据库参数组中的参数。

3. 将参数组与 RDS for Db2 数据库实例相关联。有关更多信息，请参阅 在 Amazon RDS 中将数据库参数组与数据库实例关联。

连接到您的 Db2 数据库服务器

连接到 Db2 数据库服务器的说明因语言而异。

Java

使用 Java 连接到 Db2 数据库服务器

1. 下载 JDBC 驱动程序。有关更多信息，请参阅 IBM 支持文档中的 DB2 JDBC Driver Versions and Downloads。
2. 使用以下内容创建 Shell 文件。此脚本会将捆绑包中的所有证书添加到 Java KeyStore。

> **Important**
>
> 验证 `keytool` 在脚本中的路径上是否存在，以便脚本可以找到它。如果您使用 Db2 客户端，则可以在 `~sqlib/java/jdk64/jre/bin` 下找到 `keytool`。

```
#!/bin/bash
PEM_FILE=$1
PASSWORD=$2
KEYSTORE=$3
# number of certs in the PEM file
CERTS=$(grep 'END CERTIFICATE' $PEM_FILE| wc -l)
for N in $(seq 0 $(($CERTS - 1))); do
    ALIAS="${PEM_FILE%.*}-$N"
    cat $PEM_FILE |
    awk "n==$N { print }; /END CERTIFICATE/ { n++ }" |
    keytool -noprompt -import -trustcacerts -alias $ALIAS -keystore $KEYSTORE -
storepass $PASSWORD
```

```
done
```

3. 要运行 Shell 脚本并将包含证书捆绑包的 PEM 文件导入到 Java KeyStore 中，请运行以下命令。将 *shell_file_name.sh* 替换为 Shell 脚本文件的名称，并将 *password* 替换为 Java KeyStore 的密码。

```
./shell_file_name.sh global-bundle.pem password truststore.jks
```

4. 要连接到 Db2 服务器，请运行以下命令。将示例中的以下占位符替换为 RDS for Db2 数据库实例信息。

   - *ip_address* – 您的数据库实例端点的 IP 地址。
   - *port* – SSL 连接的端口号。这可以是任何端口号，但用于非 SSL 端口的端口号除外。
   - *database_name* – 数据库实例中数据库的名称。
   - *master_username* – 数据库实例的主用户名。
   - *master_password* – 数据库实例的主密码。

```
export trustStorePassword=MyPassword
java -cp ~/dsdriver/jdbc_sqlj_driver/linuxamd64/db2jcc4.jar \
com.ibm.db2.jcc.DB2Jcc -url \
"jdbc:db2://ip_address:port/database_name:\
sslConnection=true;sslTrustStoreLocation=\
~/truststore.jks;\
sslTrustStorePassword=${trustStorePassword};\
sslVersion=TLSv1.2;\
encryptionAlgorithm=2;\
securityMechanism=7;" \
-user master_username -password master_password
```

Node.js

使用 Node.js 连接到 Db2 数据库服务器

1. 安装 node-ibm_db 驱动程序。有关更多信息，请参阅 IBM Db2 文档中的 [Installing the node-ibm_db driver on Linux and UNIX systems]。
2. 基于以下内容创建 JavaScript 文件。将示例中的以下占位符替换为 RDS for Db2 数据库实例信息。

- *ip_address* – 您的数据库实例端点的 IP 地址。
- *master_username* – 数据库实例的主用户名。
- *master_password* – 数据库实例的主密码。
- *database_name* – 数据库实例中数据库的名称。
- *port* – SSL 连接的端口号。这可以是任何端口号，但用于非 SSL 端口的端口号除外。

```
var ibmdb = require("ibm_db");
const hostname = "ip_address";
const username = "master_username";
const password = "master_password";
const database = "database_name";
const port = "port";
const certPath = "/root/qa-bundle.pem";
ibmdb.open("DRIVER={DB2};DATABASE=" + database + ";HOSTNAME=" +
 hostname + ";UID=" + username + ";PWD=" + password + ";PORT=" + port +
 ";PROTOCOL=TCPIP;SECURITY=SSL;SSLServerCertificate=" + certPath + ";", function
 (err, conn){
 if (err) return console.log(err);
 conn.close(function () {
 console.log('done');
 });
});
```

3. 要运行 JavaScript 文件，请运行以下命令。

```
node ssl-test.js
```

Python

使用 Python 连接到 Db2 数据库服务器

1. 创建 Python 文件并输入以下内容。将示例中的以下占位符替换为 RDS for Db2 数据库实例信息。

   - *port* – SSL 连接的端口号。这可以是任何端口号，但用于非 SSL 端口的端口号除外。
   - *master_username* – 数据库实例的主用户名。
   - *master_password* – 数据库实例的主密码。

- *database_name* – 数据库实例中数据库的名称。
- *ip_address* – 您的数据库实例端点的 IP 地址。

```
import click
import ibm_db
import sys

port = port;
master_user_id = "master_username" # Master id used to create your DB instance
master_password = "master_password" # Master password used to create your DB
 instance
db_name = "database_name" # If not given "db-name'
vpc_customer_private_ip = "ip_address" # Hosts end points - Customer private IP
 Addressicert_path = "/root/ssl/global-bundle.pem" # cert path

@click.command()
@click.option("--path", help="certificate path")
def db2_connect(path):

    try:
        conn =
 ibm_db.connect(f"DATABASE={db_name};HOSTNAME={vpc_customer_private_ip};PORT={port};

 PROTOCOL=TCPIP;UID={master_user_id};PWD={master_password};SECURITY=ssl;SSLServerCertifi
 "", "")
        try:
            ibm_db.exec_immediate(conn, 'create table tablename (a int);')
            print("Query executed successfully")
        except Exception as e:
            print(e)
        finally:
            ibm_db.close(conn)
            sys.exit(1)
    except Exception as ex:
        print("Trying to connect...")

if __name__ == "__main__":
    db2_connect()
```

2. 创建以下 Shell 脚本，该脚本将运行您创建的 Python 文件。将 *python_file_name.py* 替换为 Python 脚本文件的名称。

```
#!/bin/bash
PEM_FILE=$1
# number of certs in the PEM file
CERTS=$(grep 'END CERTIFICATE' $PEM_FILE| wc -l)

for N in $(seq 0 $(($CERTS - 1))); do
    ALIAS="${PEM_FILE%.*}-$N"
    cert=`cat $PEM_FILE | awk "n==$N { print }; /END CERTIFICATE/ { n++ }"`
    cat $PEM_FILE | awk "n==$N { print }; /END CERTIFICATE/ { n++ }" >
 $ALIAS.pem
    python3 <python_file_name.py> --path $ALIAS.pem
    output=`echo $?`
    if [ $output == 1 ]; then
        break
    fi
done
```

3. 要导入包含证书捆绑包的 PEM 文件并运行 Shell 脚本，请运行以下命令。将 *shell_file_name.sh* 替换为您的 Shell 脚本文件的名称。

```
./shell_file_name.sh global-bundle.pem
```

Db2 CLP

使用 Db2 CLP 连接到 Db2 数据库服务器

1. 要使用 Db2 CLP 连接到您的 Db2 实例，您需要使用 GSKit，可从 IBM Fix Central 下载它。要使用 Db2 CLP，您还需要使用 IBM Db2 客户端，可从 IBM Support 的 Download initial Version 11.5 clients and drivers 下载该客户端。
2. 创建密钥库。

```
gsk8capicmd_64 -keydb -create -db "directory/keystore-filename" -pw
 "changeThisPassword" -type pkcs12 -stash
```

3. 将证书捆绑包导入到该密钥库。

```
gsk8capicmd_64 -cert -import -file global-bundle.pem -target directory/keystore-
filename> -target_stashed
```

4. 更新 Db2 实例配置。

```
db2 update dbm cfg using SSL_CLNT_KEYDB keystore-filename
 SSL_CLNT_STASH keystore stash file immediate
```

5. 对节点和数据库进行编目。

```
db2 catalog tcpip node ssluse1 REMOTE endpoint SERVER ssl_svcename security ssl

db2 catalog database testdb as ssltest at node ssluse1
```

6. 连接到 数据库。

```
db2 connect to ssltest user username using password
```

将 Kerberos 身份验证用于 Amazon RDS for Db2

当用户连接到 Amazon RDS for Db2 数据库实例时，您可以使用 Kerberos 身份验证来验证用户的身份。您的数据库实例与 Amazon Directory Service for Microsoft Active Directory（Amazon Managed Microsoft AD）结合使用以启用 Kerberos 身份验证。当用户对加入到信任域的 RDS for Db2 数据库实例进行身份验证时，身份验证请求将转发至您使用 Amazon Directory Service 创建的目录。有关更多信息，请参阅《Amazon Directory Service 管理员指南》中的什么是 Amazon Directory Service？。

首先，创建一个 Amazon Managed Microsoft AD 目录来存储用户凭证。然后，将您的 Amazon Managed Microsoft AD 目录中的域和其它信息添加到 RDS for Db2 数据库实例。当用户使用 RDS for Db2 数据库实例进行身份验证时，身份验证请求将转发到 Amazon Managed Microsoft AD 目录。

将所有凭证保存在同一目录中可以节省您的时间和精力。使用这种方法，您具有一个集中位置用于存储和管理多个数据库实例的凭证。使用目录还可以改善您的整体安全概要。

有关 Kerberos 身份验证的信息，请参阅以下主题。

主题

- 为 Amazon RDS for Db2 数据库实例设置 Kerberos 身份验证
- 通过 Kerberos 身份验证连接到 Amazon RDS for Db2

区域和版本可用性

功能可用性和支持因每个数据库引擎的特定版本以及 Amazon Web Services 区域而异。有关使用 Kerberos 身份验证的 RDS for Db2 的版本和区域可用性的更多信息，请参阅支持 Amazon RDS 中 Kerberos 身份验证功能的区域和数据库引擎。

> **Note**
>
> 对于 RDS for Db2 数据库实例已弃用的数据库实例类，不支持 Kerberos 身份验证。有关更多信息，请参阅 Amazon RDS for Db2 实例类。

RDS for Db2 数据库实例的 Kerberos 身份验证概述

要为 RDS for Db2 数据库实例设置 Kerberos 身份验证，请完成以下一般步骤，稍后将详细介绍这些步骤：

1. 使用 Amazon Managed Microsoft AD 创建 Amazon Managed Microsoft AD 目录。您可以使用 Amazon Web Services Management Console、 Amazon Command Line Interface（Amazon CLI）或 Amazon Directory Service 创建目录。有关更多信息，请参阅《Amazon Directory Service 管理指南》中的创建您的 Amazon Managed Microsoft AD 目录。
2. 创建使用托管式 IAM 策略 `AmazonRDSDirectoryServiceAccess` 的 Amazon Identity and Access Management（IAM）角色。此 IAM 角色允许 Amazon RDS 调用您的目录。

   为了让 IAM 角色允许访问，Amazon Security Token Service（Amazon STS）端点必须在您的 Amazon Web Services 账户的正确 Amazon Web Services 区域中激活。Amazon STS 端点默认为在所有 Amazon Web Services 区域中处于活动状态，您无需执行任何进一步操作即可使用它们。有关更多信息，请参阅《IAM 用户指南》中的在 Amazon Web Services 区域中激活和停用 Amazon STS。
3. 通过以下方法之一使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 创建或修改 RDS for Db2 数据库实例：
   - 使用控制台、create-db-instance 命令或 CreateDBInstance API 操作创建新的 RDS for Db2 数据库实例。有关说明，请参阅创建 Amazon RDS 数据库实例。
   - 使用控制台、modify-db-instance 命令或 ModifyDBInstance API 操作修改现有 RDS for Db2 数据库实例。有关说明，请参阅修改 Amazon RDS 数据库实例。

- 使用控制台、restore-db-instance-from-db-snapshot 命令或 RestoreDBInstanceFromDBSnapshot API 操作从数据库快照还原 RDS for Db2 数据库实例。有关说明，请参阅还原到数据库实例。
- 使用控制台、restore-db-instance-to-point-in-time 命令或 RestoreDBInstanceToPointInTime API 操作将 RDS for Db2 数据库实例还原到某个时间点。有关说明，请参阅将 Amazon RDS 的数据库实例还原到指定时间。

  您可以在与目录相同的 Amazon Virtual Private Cloud（VPC）中或在不同的 Amazon Web Services 账户或 VPC 中找到数据库实例。创建或修改 RDS for Db2 数据库实例时，请执行以下任务：

- 请提供创建目录时生成的域标识符（d-* 标识符）。
- 还要提供您创建的 IAM 角色的名称。
- 验证数据库实例安全组可以从目录安全组接收入站流量。

4. 配置 Db2 客户端，并验证对于以下端口，流量是否可以在客户端主机与 Amazon Directory Service 之间流动：

   - TCP/UDP 端口 53 – DNS
   - TCP 88 – Kerberos 身份验证
   - TCP 389 – LDAP
   - TCP 464 – Kerberos 身份验证

在域中管理数据库实例

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 来管理数据库实例及其与 Microsoft Active Directory 的关系。例如，您可以关联 Active Directory 以启用 Kerberos 身份验证。您也可以删除 Active Directory 的关联以禁用 Kerberos 身份验证。您也可以将由一个 Microsoft Active Directory 在外部进行身份验证的数据库实例移到另一个 Active Directory。

例如，通过运行 modify-db-instance CLI 命令，您可以执行以下操作：

- 重新尝试为失败的成员资格启用 Kerberos 身份验证，方法是为 `--domain` 选项指定当前成员资格的目录 ID。
- 通过为 `--domain` 选项指定 `none` 来禁用数据库实例的 Kerberos 身份验证。
- 通过为 `--domain` 选项指定新域的域标识符，将数据库实例从一个域移动到另一个域。

了解域成员资格

在创建或修改数据库实例后，它将成为域的成员。您可以在控制台中查看域成员资格的状态，也可以通过运行 describe-db-instances 命令来查看。数据库实例的状态可以是以下状态之一：

- `kerberos-enabled` – 数据库实例已启用 Kerberos 身份验证。
- `enabling-kerberos` – Amazon 正在此数据库实例上启用 Kerberos 身份验证。
- `pending-enable-kerberos` – 启用 Kerberos 身份验证正在此数据库实例上等待处理。
- `pending-maintenance-enable-kerberos` – Amazon 将尝试在下一个计划的维护时段在数据库实例上启用 Kerberos 身份验证。
- `pending-disable-kerberos` – 禁用 Kerberos 身份验证正在此数据库实例上等待处理。
- `pending-maintenance-disable-kerberos` – Amazon 将尝试在下一个计划的维护时段在数据库实例上禁用 Kerberos 身份验证。
- `enable-kerberos-failed` – 出现一个配置问题，导致 Amazon 无法在数据库实例上启用 Kerberos 身份验证。在重新发出命令以修改数据库实例之前纠正配置问题。
- `disabling-kerberos` – Amazon 正在此数据库实例上禁用 Kerberos 身份验证。

启用 Kerberos 身份验证的请求可能因网络连接问题或不正确的 IAM 角色而失败。在某些情况下，在创建或修改数据库实例时，尝试启用 Kerberos 身份验证可能会失败。如果发生这种情况，请确保使用正确的 IAM 角色，然后修改数据库实例以加入域。

为 Amazon RDS for Db2 数据库实例设置 Kerberos 身份验证

您可以使用 Amazon Directory Service for Microsoft Active Directory（Amazon Managed Microsoft AD）为 RDS for Db2 数据库实例设置 Kerberos 身份验证。要设置 Kerberos 身份验证，请按照以下步骤操作：

主题

步骤 1：使用 Amazon Managed Microsoft AD 创建目录

Amazon Directory Service 在 Amazon Web Services 云 中创建完全托管式 Active Directory。在创建 Amazon Managed Microsoft AD 目录时，Amazon Directory Service 将创建两个域控制器和 DNS 服务器。目录服务器在 VPC 中的不同子网中创建。这种冗余有助于确保即使发生故障，目录仍可访问。

创建 Amazon Managed Microsoft AD 目录时，Amazon Directory Service 代表您执行以下任务：

- 在您的 VPC 中设置 Active Directory。
- 创建具有用户名 `Admin` 和指定密码的目录管理员账户。您可以使用此账户管理您的目录。

> **Important**
>
> 请务必保存此密码。Amazon Directory Service 不会存储此密码并且它无法取回或重置。

- 为目录控制器创建安全组。安全组必须允许与 RDS for Db2 数据库实例进行通信。

在启动 Amazon Directory Service for Microsoft Active Directory 时，Amazon 创建一个组织单位（OU），其中包含目录的所有对象。此 OU（具有您在创建目录时输入的 NetBIOS 名称）位于域根目录中。此域根目录由 Amazon 拥有和管理。

使用 `Admin` 目录创建的 Amazon Managed Microsoft AD 账户有权为您的 OU 执行最常见的管理活动：

- 创建、更新或删除用户。
- 将资源添加到域（如文件或打印服务器），然后为 OU 中的用户分配这些资源的权限。
- 创建额外的 OU 和容器。
- 委托授权。
- 从 Active Directory 回收站还原删除的对象。
- 在 Amazon Directory Service 上对 Windows PowerShell 运行 Active Directory 和域名服务（DNS）模块。

`Admin` 账户还有权执行以下域范围的活动：

- 管理 DNS 配置（添加、删除或更新记录、区域和转发器）。
- 查看 DNS 事件日志。
- 查看安全事件日志。

使用 Amazon Managed Microsoft AD 创建目录

1. 登录 Amazon Web Services Management Console，然后打开 Amazon Directory Service 控制台，网址为：https://console.aws.amazon.com/directoryservicev2/。
2. 选择设置目录。
3. 选择 Amazon Managed Microsoft AD。Amazon Managed Microsoft AD 是当前唯一支持用于 Amazon RDS 的选项。
4. 选择下一步。
5. 在输入目录信息页面上，提供以下信息：
   - 版本 – 选择符合您要求的版本。
   - 目录 DNS 名称 – 目录的完全限定名称，例如 `corp.example.com`。
   - 目录 NetBIOS 名称 – 目录的可选短名称，例如 `CORP`。
   - 目录描述 – 目录的可选描述。
   - 管理员密码 – 目录管理员的密码。目录创建过程将使用用户名 `Admin` 和此密码创建一个管理员账户。

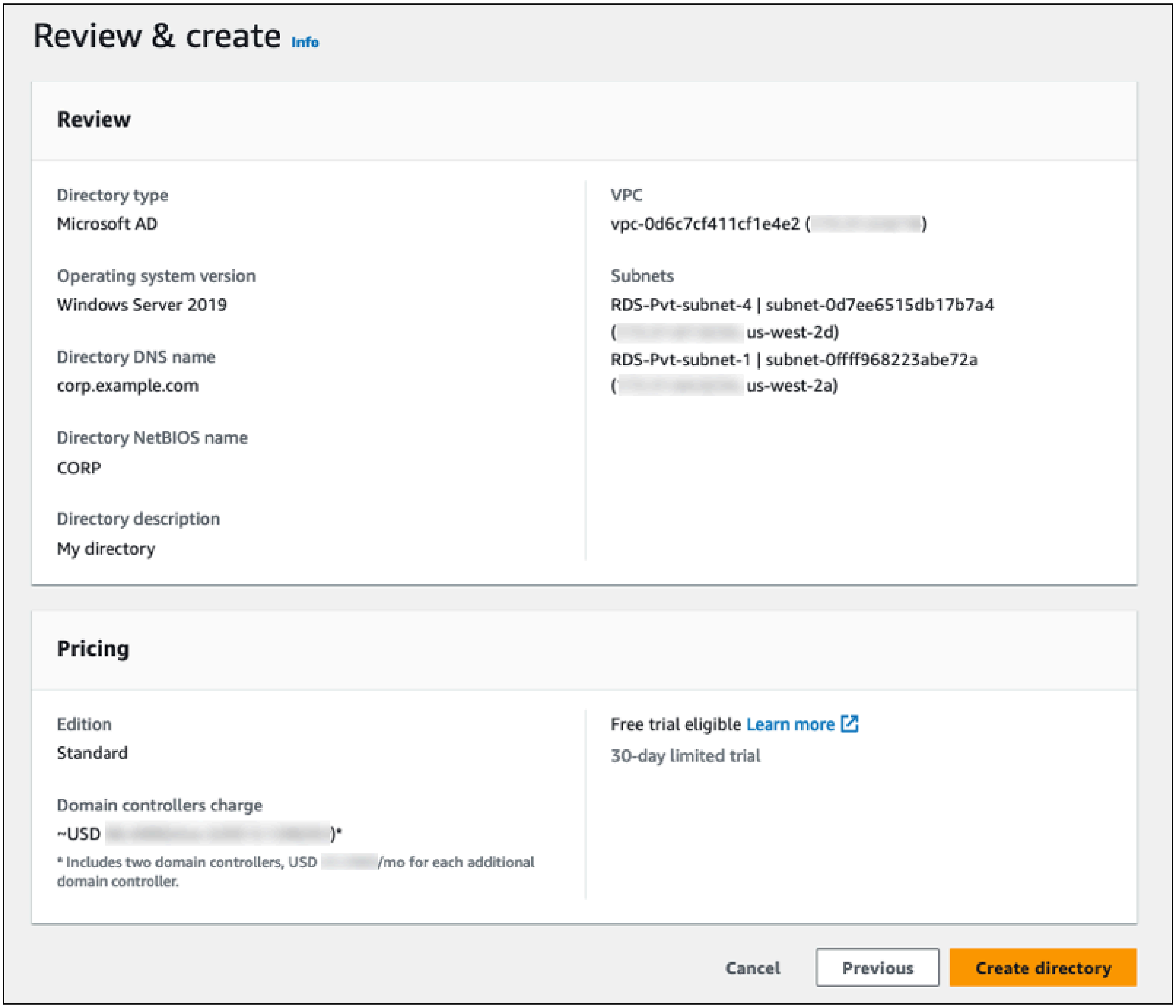
     目录管理员密码不能包含单词“admin”。此密码区分大小写，且长度必须介于 8 – 64 个字符之间。至少，它还必须包含下列四种类别中三种类别的一个字符：
     - 小写字母 (a–z)
     - 大写字母 (A–Z)
     - 数字 (0–9)
     - 非字母数字字符 (~!@#$%^&*_-+=`|\(){}[]:;"'<>,.?/)
     - 确认密码 – 重新键入管理员密码。

     > **Important**
     > 确保您保存该密码。Amazon Directory Service 不会存储该密码，也无法检索或重置该密码。
6. 选择下一步。
7. 在选择 VPC 和子网页面上，提供以下信息：
   - VPC – 为目录选择 VPC。您可以在该相同 VPC 或不同的 VPC 中创建 RDS for Db2 数据库实例。

- 子网 – 为目录服务器选择子网。两个子网必须位于不同的可用区。

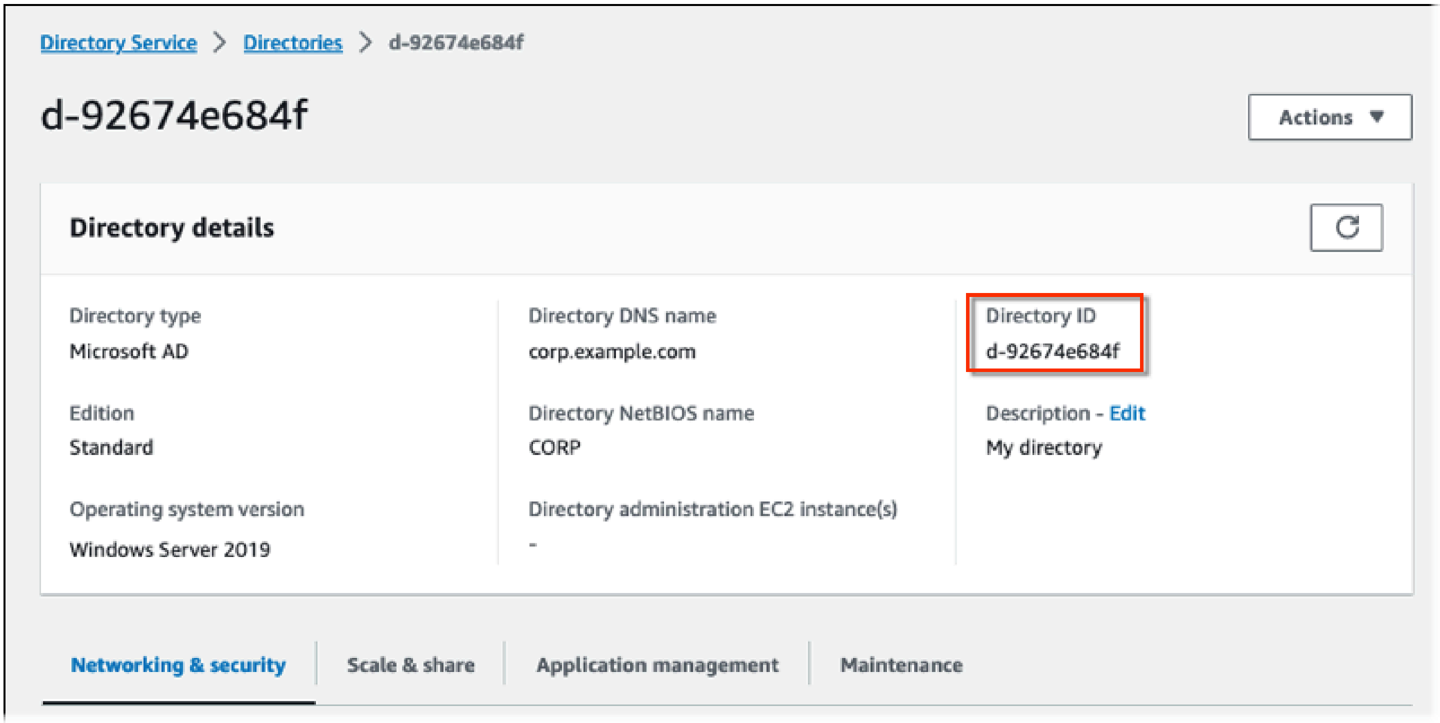
8. 选择下一步。

9. 检查目录信息。如果需要进行更改，请选择 Previous (上一步) 并作出更改。如果信息正确，请选择 Create directory (创建目录)。



目录创建需要几分钟时间。创建成功后，Status (状态) 值将更改为 Active (活动)。

要查看有关您的目录的信息，请在目录 ID 下选择目录 ID。记下目录 ID 值。创建或修改 RDS for Db2 数据库实例时需要此值。



步骤 2：为 Amazon RDS 创建 IAM 角色以访问 Amazon Directory Service

要使 Amazon RDS 为您调用 Amazon Directory Service，您的 Amazon Web Services 账户需要一个使用托管式 IAM policy `AmazonRDSDirectoryServiceAccess` 的 IAM 角色。该角色允许 Amazon RDS 调用 Amazon Directory Service。

当您使用 Amazon Web Services Management Console 创建数据库实例并且控制台用户账户具有 `iam:CreateRole` 权限时，控制台将自动创建所需的 IAM 角色。在这种情况下，角色名为 `rds-directoryservice-kerberos-access-role`。否则，您必须手动创建 IAM 角色。在创建该 IAM 角色时，请选择 `Directory Service`，然后将 Amazon 托管策略 `AmazonRDSDirectoryServiceAccess` 附加到该角色。

有关为服务创建 IAM 角色的更多信息，请参阅 IAM 用户指南中的[创建向Amazon服务委托权限的角色]。

> **Note**
>
> 用于 RDS for Microsoft SQL Server 的 Windows 身份验证的 IAM 角色不能用于 RDS for Db2。

作为使用 `AmazonRDSDirectoryServiceAccess` 托管式策略的替代方法，您可以创建具有所需权限的策略。在这种情况下，IAM 角色必须具有以下 IAM 信任策略：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "directoryservice.rds.amazonaws.com",
          "rds.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

该角色还必须具有以下 IAM 角色策略：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "ds:DescribeDirectories",
        "ds:AuthorizeApplication",
        "ds:UnauthorizeApplication",
        "ds:GetAuthorizedApplicationDetails"
      ],
    "Effect": "Allow",
    "Resource": "*"
    }
  ]
}
```

步骤 3：创建和配置用户

您可以使用 Active Directory Users and Computers 工具创建用户。这是 Active Directory Domain Services 和 Active Directory Lightweight Directory Services 工具之一。有关更多信息，请参阅

Microsoft 文档中的将用户和计算机添加到 Active Directory 域。在这种情况下，用户是个人或其它实体，例如属于域的计算机，其身份在目录中维护。

要在 Amazon Directory Service 目录中创建用户，您必须连接到属于 Amazon Directory Service 目录成员的基于 Windows 的 Amazon EC2 实例。同时，您必须以具有创建用户权限的用户身份登录。有关更多信息，请参阅 Amazon Directory Service 管理指南中的创建用户。

步骤 4：在 Amazon Managed Microsoft AD 中创建 RDS for Db2 管理员组

RDS for Db2 不支持对主用户或两个 Amazon RDS 保留用户 `rdsdb` 和 `rdsadmin` 进行 Kerberos 身份验证。相反，您需要在 Amazon Managed Microsoft AD 中创建一个名为 `masterdba` 的新组。有关更多信息，请参阅 Microsoft 文档中的在 Active Directory 中创建组账户。您添加到该组的任何用户都将拥有主用户权限。

启用 Kerberos 身份验证后，主用户将失去 `masterdba` 角色。因此，除非您禁用 Kerberos 身份验证，否则主用户将无法访问实例本地用户组成员资格。要继续使用带密码登录的主用户，请在 Amazon Managed Microsoft AD 上创建一个与主用户同名的用户。然后，将该用户添加到组 `masterdba` 中。

步骤 5：创建或修改 RDS for Db2 数据库实例

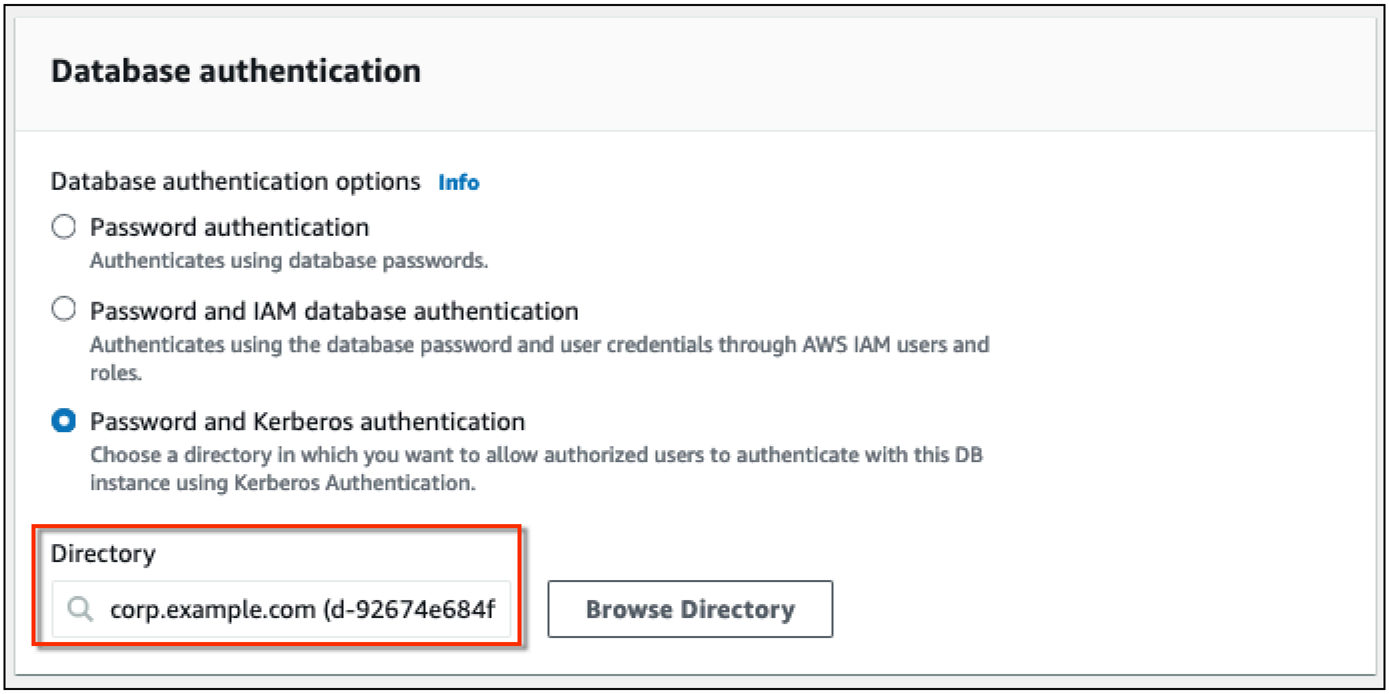
创建或修改 RDS for Db2 数据库实例，以便与您的目录一起使用。您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 将数据库实例与目录关联。您可以通过下列方式之一来执行该操作：

- 使用控制台、create-db-instance 命令或 CreateDBInstance API 操作创建新的 RDS for Db2 数据库实例。有关说明，请参阅 创建 Amazon RDS 数据库实例。
- 使用控制台、modify-db-instance 命令或 ModifyDBInstance API 操作修改现有 RDS for Db2 数据库实例。有关说明，请参阅 修改 Amazon RDS 数据库实例。
- 使用控制台、restore-db-instance-from-db-snapshot 命令或 RestoreDBInstanceFromDBSnapshot API 操作从数据库快照还原 RDS for Db2 数据库实例。有关说明，请参阅 还原到数据库实例。
- 使用控制台、restore-db-instance-to-point-in-time 命令或 RestoreDBInstanceToPointInTime API 操作将 RDS for Db2 数据库实例还原到某个时间点。有关说明，请参阅 将 Amazon RDS 的数据库实例还原到指定时间。

只有 VPC 中的 RDS for Db2 数据库实例才支持 Kerberos 身份验证。数据库实例可以与目录在同一 VPC 中或在不同 VPC 中。数据库实例必须使用允许在目录的 VPC 中传入和传出的安全组，以便数据库实例可与目录通信。

控制台

在使用控制台创建、修改或还原数据库实例时，请选择数据库身份验证部分中的密码和 Kerberos 身份验证。然后选择浏览目录。选择目录或选择创建目录以使用 Directory Service。



Amazon CLI

使用 Amazon CLI 时，数据库实例需要以下参数才能使用您创建的目录：

- 对于 `--domain` 参数，请使用创建目录时生成的域标识符（“d-*”标识符）。
- 对于 `--domain-iam-role-name` 参数，请使用您使用托管 IAM 策略 `AmazonRDSDirectoryServiceAccess` 创建的角色。

以下示例修改数据库实例以使用目录。将示例中的以下占位符替换为您自己的值：

- *`db_instance_name`* – RDS for Db2 数据库实例的名称。
- *`directory_id`* – 您创建的 Amazon Directory Service for Microsoft Active Directory 目录的 ID。
- *`role_name`* – 您创建的 IAM 角色的名称。

```
aws rds modify-db-instance --db-instance-identifier db_instance_name --domain
 d-directory_id --domain-iam-role-name role_name
```

> **Important**
>
> 如果您修改数据库实例以启用 Kerberos 身份验证，请在进行更改之后重启数据库实例。

步骤 6：配置 Db2 客户端

配置 Db2 客户端

1. 创建一个 /etc/krb5.conf 文件（或等效的文件）以指向域。

   > **Note**
   >
   > 对于 Windows 操作系统，请创建 C:\windows\krb5.ini 文件。

2. 验证流量是否可以在客户端主机和 Amazon Directory Service 之间流动。使用网络实用程序（如 Netcat）执行以下任务：

   a. 验证端口 53 上通过 DNS 的流量。

   b. 验证端口 53 上通过 TCP/UDP 的流量以及 Kerberos 的流量，包括用于 Amazon Directory Service 的端口 88 和 464。

3. 验证流量是否可以通过数据库端口在客户端主机和数据库实例之间流动。您可以使用命令 db2 连接和访问数据库。

以下示例是 Amazon Managed Microsoft AD 的 /etc/krb5.conf 文件内容：

```
[libdefaults]
default_realm = EXAMPLE.COM
[realms]
EXAMPLE.COM = {
kdc = example.com
admin_server = example.com
}
[domain_realm]
.example.com = EXAMPLE.COM
example.com = EXAMPLE.COM
```

通过 Kerberos 身份验证连接到 Amazon RDS for Db2

使用以下过程通过 Kerberos 身份验证连接到 Amazon RDS for Db2 数据库实例。

通过 Kerberos 身份验证连接到 RDS for Db2

1. 在命令提示符处，运行以下 命令。在以下示例中，将 *username* 替换为您的 Microsoft Active Directory 用户名。

```
kinit username
```

2. 如果 RDS for Db2 数据库实例使用可公开访问的 VPC，请将数据库实例端点的 IP 地址添加到 Amazon EC2 客户端上的 `/etc/hosts` 文件。以下示例获取 IP 地址，然后将其添加到 `/etc/hosts` 文件中。

```
% dig +short Db2-endpoint.AWS-Region.rds.amazonaws.com
;; Truncated, retrying in TCP mode.
ec2-34-210-197-118.AWS-Region.compute.amazonaws.com.
34.210.197.118

% echo "34.210.197.118  Db2-endpoint.AWS-Region.rds.amazonaws.com" >> /etc/hosts
```

3. 使用以下命令登录与 Active Directory 关联的 RDS for Db2 数据库实例。将 *database_name* 替换为 RDS for Db2 数据库的名称。

```
db2 connect to database_name
```

管理您的 Amazon RDS for Db2 数据库实例

本主题介绍您使用 Amazon RDS for Db2 数据库实例执行的常见管理任务。对于所有 Amazon RDS 数据库实例，有些任务是相同的。其它任务特定于 RDS for Db2。

以下任务是所有 RDS 数据库的共同任务。还有一些特定于 RDS for Db2 的任务，例如使用标准 SQL 客户端连接到 RDS for Db2 数据库。

任务区域	相关文档
实例类、存储和 PIOPS 如果您正在创建生产实例，请了解实例类、存储类型和预置 IOPS 在 Amazon RDS 中的工作原理。	数据库实例类 Amazon RDS 存储类型
多可用区部署 生产数据库实例应使用多可用区部署。多可用区部署为数据库实例提供了更高的可用性、数据持久性和容错能力。	配置和管理 Amazon RDS 的多可用区部署
Amazon VPC 如果您的 Amazon Web Services 账户具有默认虚拟私有云（VPC），则会自动在默认 VPC 中创建数据库实例。如果您的账户没有默认 VPC，并且您希望数据库实例位于 VPC 中，则必须在创建数据库实例之前创建 VPC 和子网组。	在 VPC 中使用数据库实例
安全组 默认情况下，数据库实例使用阻止访问的防火墙。确保使用正确的 IP 地址和网络配置创建安全组来访问数据库实例。	使用安全组控制访问权限
参数组 由于 RDS for Db2 数据库实例需要您添加 `rds.ibm_customer_id` 和 `rds.ibm_site_id` 参数，因此请在创建数据库实例前创建参数组。如果您的数据库实例需要其它特定的数据库参数，也请在创建数据库实例之前将它们添加到此参数组中。	将 IBM ID 添加到 RDS for Db2 数据库实例的参数组中 Amazon RDS 的参数组

任务区域	相关文档
选项组 如果数据库实例需要特定的数据库选项，则在创建数据库实例前创建选项组。	Amazon RDS for Db2 数据库实例选项
连接到您的数据库实例 在创建安全组并将其与数据库实例关联后，您可以使用任何标准 SQL 客户端应用程序（如 IBM Db2 CLP）连接到数据库实例。	连接到 Db2 数据库实例
备份和还原 您可以配置数据库实例以采用自动存储备份或采用手动存储快照，然后从备份或快照还原实例。	备份、还原和导出数据
监控 您可以使用 IBM Db2 Data Management Console 监控 RDS for Db2 数据库实例。 您可以使用 CloudWatch Amazon RDS 指标、事件和增强监控来监控 RDS for Db2 数据库实例。	使用 IBM Db2 Data Management Console 连接到 Amazon RDS for Db2 数据库实例 在 Amazon RDS 控制台中查看指标 查看 Amazon RDS 事件 使用增强监控来监控操作系统指标
日志文件 您可以访问 RDS for Db2 数据库实例的日志文件。	监控 Amazon RDS 日志文件

主题

- 执行 Amazon RDS for Db2 数据库实例的常见系统任务
- 执行 Amazon RDS for Db2 数据库实例的常见数据库任务

执行 Amazon RDS for Db2 数据库实例的常见系统任务

您可以在运行 Db2 的 Amazon RDS 数据库实例上执行某些与系统相关的常见数据库管理员任务。为了提供托管服务体验，Amazon RDS 不允许通过 shell 访问数据库实例，而仅限访问某些需要高级权限的系统过程和表。

有关对于 RDS for Db2 授予和撤销权限以及连接到远程数据库的信息，请参阅以下主题。

主题

- 授予和撤销对 RDS for Db2 的权限
- 连接到远程 RDS for Db2 数据库实例

创建自定义数据库端点

迁移到 Amazon RDS for Db2 时，您可以使用自定义数据库端点 URL 来最大限度地减少对应用程序的更改。例如，如果您使用 `db2.example.com` 作为当前 DNS 记录，则可以将其添加到 Amazon Route 53。在 Route 53 中，您可以使用私有托管区将当前 DNS 数据库端点映射到 RDS for Db2 数据库端点。要为 Amazon RDS 数据库端点添加自定义 `A` 或 `CNAME` 记录，请参阅《Amazon Route 53 开发人员指南》中的 Registering and managing domains using Amazon Route 53。

> **Note**
>
> 如果您无法将域转到 Route 53，则可以使用 DNS 提供商为 RDS for Db2 数据库端点 URL 创建 `CNAME` 记录。请查阅您的 DNS 提供商文档。

授予和撤销对 RDS for Db2 的权限

用户通过附加到数据库的组中的成员资格获得对数据库的访问权限。

使用以下过程授予和撤销用于控制数据库访问权限的权限。

这些过程使用在本地计算机上运行的 IBM Db2 CLP 连接到 RDS for Db2 数据库实例。请务必对 TCPIP 节点和数据库进行编目，以连接到在本地计算机上运行的 RDS for Db2 数据库实例。有关更多信息，请参阅 使用 IBM Db2 CLP 连接到 Amazon RDS for Db2 数据库实例。

主题

- 授予用户访问您的数据库的权限

- 更改用户的密码
- 向用户添加组
- 从用户中删除组
- 删除用户
- 列出用户
- 创建角色
- 授予角色
- 撤销角色
- 删除角色
- 授予数据库授权
- 撤销数据库授权

授予用户访问您的数据库的权限

授予用户访问您的数据库的权限

1. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到 `rdsadmin` 数据库。在以下示例中，用您自己的信息替换 *master_username* 和 *master_password*。

   ```
   db2 connect to rdsadmin user master_username using master_password
   ```

   此命令生成类似于以下示例的输出：

   ```
   Database Connection Information

   Database server        = DB2/LINUXX8664 11.5.8.0
   SQL authorization ID   = ADMIN
   Local database alias   = RDSADMIN
   ```

2. 通过调用 `rdsadmin.add_user` 将用户添加到您的授权列表。有关更多信息，请参阅 rdsadmin.add_user。

   ```
   db2 "call rdsadmin.add_user(
       'username',
       'password',
       'group_name,group_name')"
   ```

3. （可选）通过调用 `rdsadmin.add_groups` 向用户添加其它组。有关更多信息，请参阅 rdsadmin.add_groups。

```
db2 "call rdsadmin.add_groups(
    'username',
    'group_name,group_name')"
```

4. 确认用户可用的权限。在以下示例中，用您自己的信息替换 *rds_database_alias*、*master_user* 和 *master_password*。另外，将 *username* 替换为用户的用户名。

```
db2 terminate
db2 connect to rds_database_alias user master_user using master_password
db2 "SELECT SUBSTR(AUTHORITY,1,20) AUTHORITY, D_USER, D_GROUP, D_PUBLIC
       FROM TABLE (SYSPROC.AUTH_LIST_AUTHORITIES_FOR_AUTHID ('username', 'U') ) AS
 T
       ORDER BY AUTHORITY"
```

此命令生成类似于以下示例的输出：

```
AUTHORITY            D_USER D_GROUP D_PUBLIC
-------------------- ------ ------- --------
ACCESSCTRL           N      N       N
BINDADD              N      N       N
CONNECT              N      N       N
CREATETAB            N      N       N
CREATE_EXTERNAL_ROUT N      N       N
CREATE_NOT_FENCED_RO N      N       N
CREATE_SECURE_OBJECT N      N       N
DATAACCESS           N      N       N
DBADM                N      N       N
EXPLAIN              N      N       N
IMPLICIT_SCHEMA      N      N       N
LOAD                 N      N       N
QUIESCE_CONNECT      N      N       N
SECADM               N      N       N
SQLADM               N      N       N
SYSADM               *      N       *
SYSCTRL              *      N       *
SYSMAINT             *      N       *
SYSMON               *      N       *
```

```
WLMADM                    N       N        N
```

5. 将 RDS for Db2 角色 ROLE_NULLID_PACKAGES、ROLE_TABLESPACES 和 ROLE_PROCEDURES 授予您向其添加用户的组。有关更多信息，请参阅 Amazon RDS for Db2 默认角色。

> **Note**
>
> 我们在 RESTRICTIVE 模式下创建 RDS for Db2 数据库实例。因此，RDS for Db2 角色 ROLE_NULLID_PACKAGES、ROLE_TABLESPACES 和 ROLE_PROCEDURES 授予对 IBM Db2 CLP 和 Dynamic SQL 的 NULLID 软件包的执行权限。这些角色还授予用户对表空间的权限。

   a. 连接到 Db2 数据库。在以下示例中，用您自己的信息替换 *database_name*、*master_user* 和 *master_password*。

   ```
   db2 connect to database_name user master_user using master_password
   ```

   b. 将角色 ROLE_NULLED_PACKAGES 授予组。在以下示例中，将 *group_name* 替换为您要向其中添加该角色的组的名称。

   ```
   db2 "grant role ROLE_NULLID_PACKAGES to group group_name"
   ```

   c. 将角色 ROLE_TABLESPACES 授予同一个组。在以下示例中，将 *group_name* 替换为您要向其中添加该角色的组的名称。

   ```
   db2 "grant role ROLE_TABLESPACES to group group_name"
   ```

   d. 将角色 ROLE_PROCEDURES 授予同一个组。在以下示例中，将 *group_name* 替换为您要向其中添加该角色的组的名称。

   ```
   db2 "grant role ROLE_PROCEDURES to group group_name"
   ```

6. 向已将用户添加到的组授予 connect、bindadd、createtab 和 IMPLICIT_SCHEMA 权限。在以下示例中，将 *group_name* 替换为将用户添加到的第二个组的名称。

```
db2 "grant usage on workload SYSDEFAULTUSERWORKLOAD to public"
db2 "grant connect, bindadd, createtab, implicit_schema on database to
 group group_name"
```

7. 对将用户添加到的每个其它组重复步骤 4 到 6。
8. 通过以用户身份进行连接、创建表、向表中插入值以及从表中返回数据，测试用户的访问权限。在以下示例中，将 *rds_database_alias*、*username* 和 *password* 替换为数据库的名称以及用户的用户名和密码。

```
db2 connect to rds_database_alias user username using password
db2 "create table t1(c1 int not null)"
db2 "insert into t1 values (1),(2),(3),(4)"
db2 "select * from t1"
```

更改用户的密码

更改用户的密码

1. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到 `rdsadmin` 数据库。在以下示例中，用您自己的信息替换 *master_username* 和 *master_password*。

```
db2 connect to rdsadmin user master_username using master_password
```

2. 通过调用 `rdsadmin.change_password` 更改密码。有关更多信息，请参阅 rdsadmin.change_password。

```
db2 "call rdsadmin.change_password(
    'username',
    'new_password')"
```

向用户添加组

向用户添加组

1. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到 `rdsadmin` 数据库。在以下示例中，用您自己的信息替换 *master_username* 和 *master_password*。

```
db2 connect to rdsadmin user master_username using master_password
```

2. 通过调用 `rdsadmin.add_groups` 向用户添加组。有关更多信息，请参阅 rdsadmin.add_groups。

```
db2 "call rdsadmin.add_groups(
    'username',
    'group_name,group_name')"
```

从用户中删除组

从用户中删除组

1. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到 `rdsadmin` 数据库。在以下示例中，用您自己的信息替换 *master_username* 和 *master_password*。

```
db2 connect to rdsadmin user master_username using master_password
```

2. 通过调用 `rdsadmin.remove_groups` 删除组。有关更多信息，请参阅 rdsadmin.remove_groups。

```
db2 "call rdsadmin.remove_groups(
    'username',
    'group_name,group_name')"
```

删除用户

从授权列表中删除用户

1. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到 `rdsadmin` 数据库。在以下示例中，用您自己的信息替换 *master_username* 和 *master_password*。

```
db2 connect to rdsadmin user master_username using master_password
```

2. 通过调用 `rdsadmin.remove_user` 将用户从您的授权列表中删除。有关更多信息，请参阅 rdsadmin.remove_user。

```
db2 "call rdsadmin.remove_user('username')"
```

列出用户

要列出授权列表上的用户，请调用 `rdsadmin.list_users` 存储过程。有关更多信息，请参阅 rdsadmin.list_users。

```
db2 "call rdsadmin.list_users()"
```

创建角色

您可以使用 rdsadmin.create_role 存储过程创建角色。

创建角色

1. 连接到 `rdsadmin` 数据库。在以下示例中，用您的信息替换 *master_username* 和 *master_password*。

   ```
   db2 connect to rdsadmin user master_username using master_password
   ```

2. 将 Db2 设置为输出内容。

   ```
   db2 set serveroutput on
   ```

3. 创建角色。有关更多信息，请参阅 the section called “rdsadmin.create_role”。

   ```
   db2 "call rdsadmin.create_role(
       'database_name',
       'role_name')"
   ```

4. 将 Db2 设置为不输出内容。

   ```
   db2 set serveroutput off
   ```

授予角色

您可以使用 rdsadmin.grant_role 存储过程为角色、用户或组分配角色。

分配角色

1. 连接到 `rdsadmin` 数据库。在以下示例中，用您的信息替换 *master_username* 和 *master_password*。

```
db2 connect to rdsadmin user master_username using master_password
```

2. 将 Db2 设置为输出内容。

```
db2 set serveroutput on
```

3. 分配角色。有关更多信息，请参阅 the section called “rdsadmin.grant_role”。

```
db2 "call rdsadmin.grant_role(
    'database_name',
    'role_name',
    'grantee',
    'admin_option')"
```

4. 将 Db2 设置为不输出内容。

```
db2 set serveroutput off
```

撤销角色

您可以使用 rdsadmin.revoke_role 存储过程从角色、用户或组撤销角色。

撤销角色

1. 连接到 `rdsadmin` 数据库。在以下示例中，用您的信息替换 *master_username* 和 *master_password*。

```
db2 connect to rdsadmin user master_username using master_password
```

2. 撤销角色。有关更多信息，请参阅 the section called “rdsadmin.revoke_role”。

```
db2 "call rdsadmin.revoke_role(
    ?,
    'database_name',
    'role_name',
    'grantee')"
```

删除角色

可以使用 rdsadmin.drop_role 存储过程删除角色。

删除角色

1. 连接到 rdsadmin 数据库。在以下示例中，用您的信息替换 *master_username* 和 *master_password*。

```
db2 connect to rdsadmin user master_username using master_password
```

2. 删除角色。有关更多信息，请参阅 the section called “rdsadmin.drop_role”。

```
db2 "call rdsadmin.drop_role(
    ?,
    'database_name',
    'role_name')"
```

授予数据库授权

拥有 DBADM 授权的主用户可以向角色、用户或组授予 DBADM、ACCESSCTRL 或 DATAACCESS 授权。

授予数据库授权

1. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到 rdsadmin 数据库。在以下示例中，用您自己的信息替换 *master_username* 和 *master_password*。

```
db2 connect to rdsadmin user master_username using master_password
```

2. 通过调用 rdsadmin.dbadm_grant 授予用户访问权限。有关更多信息，请参阅 rdsadmin.dbadm_grant。

```
db2 "call rdsadmin.dbadm_grant(
    ?,
    'database_name,
    'authorization',
    'grantee')"
```

使用案例示例

以下过程将引导您完成创建角色、向该角色授予 DBADM 授权、将该角色分配给用户以及将该角色授予组。

1. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到 rdsadmin 数据库。在以下示例中，用您自己的信息替换 *master_username* 和 *master_password*。

```
db2 connect to rdsadmin user master_username using master_password
```

2. 为名为 TESTDB 的数据库创建一个名为 PROD_ROLE 的角色。有关更多信息，请参阅 rdsadmin.create_role。

```
db2 "call rdsadmin.create_role(
    'TESTDB',
    'PROD_ROLE')"
```

3. 将该角色分配给名为 PROD_USER 的用户。PROD_USER 已获得分配角色的管理员授权。有关更多信息，请参阅 rdsadmin.grant_role。

```
db2 "call rdsadmin.grant_role(
    ?,
    'TESTDB',
    'PROD_ROLE',
    'USER PROD_USER',
    'Y')"
```

4. （可选）提供其它授权或权限。以下示例针对名为 FUNDPROD 的数据库向名为 PROD_ROLE 的角色授予 DBADM 授权。有关更多信息，请参阅 rdsadmin.dbadm_grant。

```
db2 "call rdsadmin.dbadm_grant(
    ?,
    'FUNDPROD',
    'DBADM',
    'ROLE PROD_ROLE')"
```

5. 终止会话。

```
db2 terminate
```

6. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到 TESTDB 数据库。在以下示例中，用您自己的信息替换 *master_username* 和 *master_password*。

```
db2 connect to TESTDB user master_username using master_password
```

7. 向该角色添加更多授权。

```
db2 "grant connect, implicit_schema on database to role PROD_ROLE"
```

8. 将角色 PROD_ROLE 授予组。

```
db2 "grant role PROD_ROLE to group PRODGRP"
```

属于组 PRODGRP 的用户现在可以执行诸如连接到 TESTDB 数据库、创建表或创建架构之类的操作。

撤销数据库授权

拥有 DBADM 授权的主用户可以从角色、用户或组撤销 DBADM、ACCESSCTRL 或 DATAACCESS 授权。

撤销数据库授权

1. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到 rdsadmin 数据库。在以下示例中，用您自己的信息替换 *master_username* 和 *master_password*。

```
db2 connect to rdsadmin user master_username using master_password
```

2. 通过调用 rdsadmin.dbadm_revoke 撤销用户访问权限。有关更多信息，请参阅 rdsadmin.dbadm_revoke。

```
db2 "call rdsadmin.dbadm_revoke(
    ?,
    'database_name,
    'authorization',
    'grantee')"
```

连接到远程 RDS for Db2 数据库实例

使用以下步骤连接到远程 RDS for Db2 数据库实例，并运行 get snapshot 操作。

连接到远程 RDS for Db2 数据库实例

1. 运行 IBM Db2 CLP 客户端会话。有关对 RDS for Db2 数据库实例和数据库进行编目的信息，请参阅使用 IBM Db2 CLP 连接到 Amazon RDS for Db2 数据库实例。记下您的 RDS for Db2 数据库实例的主用户名和主密码。
2. 连接到 RDS for Db2 数据库实例。在以下示例中，将 *node_name*、*master_username* 和 *master_password* 替换为您编目的 TCPIP 节点名称以及 RDS for Db2 数据库实例的主用户名和主密码。

```
db2 attach to node_name user master_username using master_password
```

连接到远程 RDS for Db2 数据库实例后，您可以运行以下命令和其它 `get snapshot` 命令。有关更多信息，请参阅 IBM Db2 文档中的 GET SNAPSHOT 命令。

```
db2 list applications
db2 get snapshot for all databases
db2 get snapshot for database manager
db2 get snapshot for all applications
```

执行 Amazon RDS for Db2 数据库实例的常见数据库任务

您可以在 Amazon RDS for Db2 数据库实例上执行某些与数据库相关的常见 DBA 任务。为了提供托管服务体验，Amazon RDS 未提供数据库实例的 Shell 访问权限。此外，主用户无法运行需要 `SYSADM`、`SYSMAINT` 或 `SYSCTRL` 权限的命令或实用程序。

有关缓冲池、数据库和表空间的常见任务的信息，请参阅以下主题。

主题

- 缓冲池的常见任务
- 常见数据库任务
- 表空间的常见任务

缓冲池的常见任务

您可以针对 RDS for Db2 数据库创建、更改或删除缓冲池。创建、更改或删除缓冲池需要更高级别的 `SYSADM` 或 `SYSCTRL` 权限，而主用户无法获得相应的权限。因此，可改为使用 Amazon RDS 存储过程。

您也可以刷新缓冲池。

主题

- [创建缓冲池]
- [更改缓冲池]
- [删除缓冲池]
- [刷新缓冲池]

创建缓冲池

要创建 RDS for Db2 数据库的缓冲池，请调用 `rdsadmin.create_bufferpool` 存储过程。有关更多信息，请参阅 IBM Db2 文档中的 CREATE BUFFERPOOL 语句。

创建缓冲池

1. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到 `rdsadmin` 数据库。在以下示例中，用您自己的信息替换 *master_username* 和 *master_password*。

```
db2 "connect to rdsadmin user master_username using master_password"
```

2. 通过调用 `rdsadmin.create_bufferpool` 创建缓冲池。有关更多信息，请参阅 rdsadmin.create_bufferpool。

```
db2 "call rdsadmin.create_bufferpool(
    'database_name',
    'buffer_pool_name',
    buffer_pool_size,
    'immediate',
    'automatic',
    page_size,
    number_block_pages,
    block_size)"
```

更改缓冲池

要更改 RDS for Db2 数据库的缓冲池，请调用 `rdsadmin.alter_bufferpool` 存储过程。有关更多信息，请参阅 IBM Db2 文档中的 ALTER BUFFERPOOL 语句。

更改缓冲池

1. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到 `rdsadmin` 数据库。在以下示例中，用您自己的信息替换 *master_username* 和 *master_password*。

```
db2 "connect to rdsadmin user master_username using master_password"
```

2. 通过调用 `rdsadmin.alter_bufferpool` 更改缓冲池。有关更多信息，请参阅 [rdsadmin.alter_bufferpool]。

```
db2 "call rdsadmin.alter_bufferpool(
    'database_name',
    'buffer_pool_name',
    buffer_pool_size,
    'immediate',
    'automatic',
    change_number_blocks,
    number_block_pages,
    block_size)"
```

删除缓冲池

要删除 RDS for Db2 数据库的缓冲池，请调用 `rdsadmin.drop_bufferpool` 存储过程。有关更多信息，请参阅 IBM Db2 文档中的[删除缓冲池]。

> **Important**
> 确保没有为要删除的缓冲池分配任何表空间。

删除缓冲池

1. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到 `rdsadmin` 数据库。在以下示例中，用您自己的信息替换 *master_username* 和 *master_password*。

```
db2 "connect to rdsadmin user master_username using master_password"
```

2. 通过调用 `rdsadmin.drop_bufferpool` 删除缓冲池。有关更多信息，请参阅 [rdsadmin.drop_bufferpool]。

```
db2 "call rdsadmin.drop_bufferpool(
    'database_name',
    'buffer_pool_name')"
```

刷新缓冲池

您可以刷新缓冲池以强制实施检查点，以便 RDS for Db2 将页面从内存写入存储。

> **Note**
>
> 您不需要刷新缓冲池。Db2 在提交事务之前同步写入日志。脏页可能仍在缓冲池中，但是 Db2 会将它们异步写入存储。即使系统意外关闭，当您重启数据库时，Db2 也会自动执行崩溃恢复。在崩溃恢复期间，Db2 将已提交的更改写入数据库，或者回滚对未提交事务的更改。

刷新缓冲池

1. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到您的 Db2 数据库。在以下示例中，用您自己的信息替换 *rds_database_alias*、*master_username* 和 *master_password*。

   ```
   db2 connect to rds_database_alias user master_username using master_password
   ```

2. 刷新缓冲池。

   ```
   db2 flush bufferpools all
   ```

常见数据库任务

可以在 RDS for Db2 数据库实例上创建、删除或还原数据库。创建、删除或还原数据库需要更高级别的 SYSADM 权限，而主用户无法获得该权限。因此，可改为使用 Amazon RDS 存储过程。

还可以执行常见的管理任务，例如监控、维护和收集有关数据库的信息。

主题

- 创建数据库
- 为数据库配置设置

- 修改数据库参数
- 配置日志保留期
- 列出日志信息
- 停用数据库
- 激活数据库
- 重新激活数据库
- 删除数据库
- 还原数据库
- 列出数据库
- 收集有关数据库的信息
- 强制应用程序退出数据库
- 生成性能报告

创建数据库

要在 RDS for Db2 数据库实例上创建数据库，请调用 `rdsadmin.create_database` 存储过程。有关更多信息，请参阅 IBM Db2 文档中的 CREATE DATABASE 命令。

> **Note**
>
> 如果您打算修改 `db2_compatibility_vector` 参数，请在创建数据库之前进行修改。有关更多信息，请参阅 设置 db2_compatibility_vector 参数。

创建数据库

1. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到 `rdsadmin` 数据库。在以下示例中，用您自己的信息替换 *master_username* 和 *master_password*。

```
db2 "connect to rdsadmin user master_username using master_password"
```

2. 通过调用 `rdsadmin.create_database` 创建数据库。有关更多信息，请参阅 rdsadmin.create_database。

```
db2 "call rdsadmin.create_database('database_name')"
```

3. （可选）通过对您要创建的每个数据库调用 `rdsadmin.create_database` 可创建额外数据库。每个 Db2 数据库实例最多可包含 50 个数据库。有关更多信息，请参阅 rdsadmin.create_database。

```
db2 "call rdsadmin.create_database('database_name')"
```

4. （可选）使用以下方法之一确认数据库已创建：

   - 调用 `rdsadmin.list_databases`。有关更多信息，请参阅 rdsadmin.list_databases。
   - 运行以下 SQL 命令：

```
db2 "select varchar(r.task_type,25) as task_type, r.database_name,
    varchar(r.lifecycle,15) as lifecycle, r.created_at, r.database_name,
    varchar(bson_to_json(task_input_params),256) as input_params,
    varchar(r.task_output,1024) as task_output
    from table(rdsadmin.get_task_status(null,null,'create_database'))
    as r order by created_at desc"
```

为数据库配置设置

要为 RDS for Db2 数据库实例上的数据库配置设置，请调用 `rdsadmin.set_configuration` 存储过程。例如，您可以配置在还原操作期间要创建的缓冲区或缓冲区操纵器的数量。

为数据库配置设置

1. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到 `rdsadmin` 数据库。在以下示例中，用您自己的信息替换 *master_username* 和 *master_password*。

```
db2 "connect to rdsadmin user master_username using master_password"
```

2. （可选）通过调用 `rdsadmin.show_configuration` 检查您当前的配置设置。有关更多信息，请参阅 the section called “rdsadmin.show_configuration”。

```
db2 "call rdsadmin.show_configuration('name')"
```

3. 通过调用 `rdsadmin.set_configuration` 配置数据库的设置。有关更多信息，请参阅 the section called “rdsadmin.set_configuration”。

```
db2 "call rdsadmin.set_configuration(
    'name',
```

```
    'value')"
```

修改数据库参数

Amazon RDS for Db2 使用三种类型的参数：数据库管理器配置参数、注册表变量和数据库配置参数。可以通过参数组更新前两种类型，并通过 rdsadmin.update_db_param 存储过程更新最后一种类型。

> **Note**
>
> 只能修改现有参数的值。无法添加 RDS for Db2 不支持的新参数。

有关这些参数以及如何修改其值的更多信息，请参阅 the section called “Db2 参数”。

配置日志保留期

要配置 Amazon RDS 为 RDS for Db2 数据库保留日志文件的时间长度，请调用 `rdsadmin.set_archive_log_retention` 存储过程。

为数据库配置日志保留期

1. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到 `rdsadmin` 数据库。在以下示例中，用您自己的信息替换 *master_username* 和 *master_password*。

```
db2 "connect to rdsadmin user master_username using master_password"
```

2. （可选）通过调用 `rdsadmin.show_archive_log_retention` 检查您当前的日志保留期配置。有关更多信息，请参阅 the section called “rdsadmin.show_archive_log_retention”。

```
db2 "call rdsadmin.show_archive_log_retention(
    ?,
    'database_name')"
```

3. 通过调用 `rdsadmin.set_archive_log_retention` 配置数据库的日志保留期。有关更多信息，请参阅 the section called “rdsadmin.set_archive_log_retention”。

```
db2 "call rdsadmin.set_archive_log_retention(
    ?,
    'database_name',
    'archive_log_retention_hours')"
```

列出日志信息

要列出有关归档日志文件的详细信息，包括使用的总存储大小等详细信息，请调用 rdsadmin.list_archive_log_information 存储过程。

列出数据库的日志信息

1. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到 rdsadmin 数据库。在以下示例中，用您自己的信息替换 *master_username* 和 *master_password*。

```
db2 "connect to rdsadmin user master_username using master_password"
```

2. 通过调用 rdsadmin.list_archive_log_information 返回日志文件信息列表。有关更多信息，请参阅 the section called “rdsadmin.list_archive_log_information”。

```
db2 "call rdsadmin.list_archive_log_information(
    ?,
    'database_name')"
```

停用数据库

要在 RDS for Db2 数据库实例上停用数据库，请调用 rdsadmin.deactivate_database 存储过程。

默认情况下，当您在 RDS for Db2 数据库实例上创建数据库时，Amazon RDS 会激活数据库。您可以停用不常使用的数据库以节省内存资源。

停用数据库

1. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到 rdsadmin 数据库。在以下示例中，用您自己的信息替换 *master_username* 和 *master_password*。

```
db2 "connect to rdsadmin user master_username using master_password"
```

2. 通过调用 rdsadmin.deactivate_database 停用数据库。有关更多信息，请参阅 rdsadmin.deactivate_database。

```
db2 "call rdsadmin.deactivate_database(
    ?,
    'database_name')"
```

激活数据库

要在独立的 RDS for Db2 数据库实例上激活数据库，请调用 `rdsadmin.activate_database` 存储过程。

默认情况下，当您在 RDS for Db2 数据库实例上创建数据库时，Amazon RDS 会激活数据库。您可以停用不常使用的数据库以节省内存资源，在以后再激活已停用的数据库。

激活数据库

1. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到 `rdsadmin` 数据库。在以下示例中，用您自己的信息替换 *master_username* 和 *master_password*。

```
db2 "connect to rdsadmin user master_username using master_password"
```

2. 通过调用 `rdsadmin.activate_database` 激活数据库。有关更多信息，请参阅 rdsadmin.activate_database。

```
db2 "call rdsadmin.activate_database(
    ?,
    'database_name')"
```

重新激活数据库

要重新激活数据库，请调用 `rdsadmin.reactivate_database` 存储过程。更改数据库配置后，您可能需要在 RDS for Db2 数据库实例上重新激活数据库。要确定是否需要重新激活数据库，请连接到该数据库并运行 `db2 get db cfg show detail`。

也可以在更改数据库配置后调用此存储过程，以便在独立的 RDS for Db2 数据库实例上重新激活数据库。或者，可以通过先调用 `rdsadmin.deactivate_database` 存储过程，然后调用 `rdsadmin.activate_database` 存储过程，在独立的 RDS for Db2 数据库实例上重新激活数据库。有关更多信息，请参阅停用数据库和激活数据库。

重新激活数据库

1. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到 `rdsadmin` 数据库。在以下示例中，用您自己的信息替换 *master_username* 和 *master_password*。

```
db2 "connect to rdsadmin user master_username using master_password"
```

2. 通过调用 `rdsadmin.reactivate_database` 重新激活数据库。有关更多信息，请参阅 rdsadmin.reactivate_database。

```
db2 "call rdsadmin.reactivate_database(
    ?,
    'database_name')"
```

删除数据库

要从 RDS for Db2 数据库实例中删除数据库，请调用 `rdsadmin.drop_database` 存储过程。有关更多信息，请参阅 IBM Db2 文档中的 Dropping databases。

> **Note**
> 只有满足某些条件时，才能通过调用存储过程来删除数据库。有关更多信息，请参阅the section called “使用说明”适用于 `rdsadmin.drop_database` 的 。

删除数据库

1. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到 `rdsadmin` 数据库。在以下示例中，用您自己的信息替换 *master_username* 和 *master_password*。

```
db2 "connect to rdsadmin user master_username using master_password"
```

2. 通过调用 `rdsadmin.drop_database` 来删除数据库。有关更多信息，请参阅 rdsadmin.drop_database。

```
db2 "call rdsadmin.drop_database('database_name')"
```

还原数据库

要将数据库从 Amazon S3 存储桶移动到 RDS for Db2 数据库实例，请调用 `rdsadmin.restore_database` 存储过程。有关更多信息，请参阅 IBM Db2 文档中的 RESTORE DATABASE 命令。

还原数据库

1. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到 rdsadmin 数据库。在以下示例中，用您自己的信息替换 *master_username* 和 *master_password*。

```
db2 "connect to rdsadmin user master_username using master_password"
```

2. （可选）通过调用 rdsadmin.show_configuration 来检查当前的配置设置以优化还原操作。有关更多信息，请参阅 the section called “rdsadmin.show_configuration”。

```
db2 "call rdsadmin.show_configuration('name')"
```

3. 通过调用 rdsadmin.set_configuration 来配置设置以优化还原操作。显式设置这些值可以提高还原包含大量数据的数据库时的性能。有关更多信息，请参阅 the section called “rdsadmin.set_configuration”。

```
db2 "call rdsadmin.set_configuration(
    'name',
    'value')"
```

4. 通过调用 rdsadmin.restore_database 来还原数据库。有关更多信息，请参阅 the section called “rdsadmin.restore_database”。

```
db2 "call rdsadmin.restore_database(
    ?,
    'database_name',
    's3_bucket_name',
    's3_prefix',
    restore_timestamp,
    'backup_type')"
```

5. （可选）通过调用 rdsadmin.list_databases 并检查已还原的数据库是否已列出，来确认您的数据库已还原。有关更多信息，请参阅 rdsadmin.list_databases。

6. 通过调用 rdsadmin.rollforward_database 使数据库恢复联机并应用其它事务日志。有关更多信息，请参阅 the section called “rdsadmin.rollforward_database”。

```
db2 "call rdsadmin.rollforward_database(
    ?,
    'database_name',
    's3_bucket_name',
    s3_prefix,
```

```
    'rollforward_to_option',
    'complete_rollforward')"
```

7. 如果在上一步中将 `complete_rollforward` 设置为 `FALSE`，则必须通过调用 `rdsadmin.complete_rollforward` 来完成使数据库恢复联机的过程。有关更多信息，请参阅 the section called “rdsadmin.complete_rollforward”。

```
db2 "call rdsadmin.complete_rollforward(
    ?,
    'database_name')"
```

列出数据库

通过调用 `rdsadmin.list_databases` 用户定义的函数，您可以列出在 Amazon RDS for Db2 上运行的所有数据库。

列出您的数据库

1. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到 `rdsadmin` 数据库。在以下示例中，用您自己的信息替换 *master_username* 和 *master_password*。

```
db2 "connect to rdsadmin user master_username using master_password"
```

2. 通过调用 `rdsadmin.list_databases` 列出您的数据库。有关更多信息，请参阅 rdsadmin.list_databases。

```
db2 "select * from table(rdsadmin.list_databases())"
```

收集有关数据库的信息

要收集有关 RDS for Db2 数据库实例上数据库的信息，请调用 `rdsadmin.db2pd_command` 存储过程。这些信息可以帮助监控您的数据库或排查问题。

收集有关数据库的信息

1. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到 `rdsadmin` 数据库。在以下示例中，用您自己的信息替换 *master_username* 和 *master_password*。

```
db2 "connect to rdsadmin user master_username using master_password"
```

2. 通过调用 `rdsadmin.db2pd_command` 收集有关数据库的信息。有关更多信息，请参阅 rdsadmin.db2pd_command。

```
db2 "call rdsadmin.db2pd_command('db2pd_cmd')"
```

强制应用程序退出数据库

要强制应用程序退出 RDS for Db2 数据库实例上的数据库，请调用 `rdsadmin.force_application` 存储过程。在对数据库进行维护之前，请强制应用程序退出数据库。

强制应用程序退出数据库

1. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到 `rdsadmin` 数据库。在以下示例中，用您自己的信息替换 *master_username* 和 *master_password*。

```
db2 "connect to rdsadmin user master_username using master_password"
```

2. 通过调用 `rdsadmin.force_application` 来强制应用程序退出数据库 有关更多信息，请参阅 rdsadmin.force_application。

```
db2 "call rdsadmin.force_application(
    ?,
    'applications')"
```

生成性能报告

您可以使用过程或脚本生成性能报告。有关使用过程的信息，请参阅 IBM Db2 文档中的 DBSUMMARY procedure - Generate a summary report of system and application performance metrics。

Db2 在其 `~sqllib/sample/perf` 目录中包含一个 `db2mon.sh` 文件。运行该脚本会生成低成本、广泛的 SQL 指标报告。要下载 `db2mon.sh` 文件和相关的脚本文件，请参阅 IBM db2-samples GitHub 存储库中的 perf 目录。

使用脚本生成性能报告

1. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到您的 Db2 数据库。在以下示例中，用您自己的信息替换 *master_username* 和 *master_password*。

```
db2 connect to rdsadmin user master_username using master_password
```

2. 通过调用 `rdsadmin.create_bufferpool` 创建一个名为 db2monbp、页面大小为 4096 的缓冲池。有关更多信息，请参阅 rdsadmin.create_bufferpool。

```
db2 "call rdsadmin.create_bufferpool('database_name','db2monbp',4096)"
```

3. 通过调用 `rdsadmin.create_tablespace` 创建一个名为 `db2montmptbsp` 且使用 `db2monbp` 缓冲池的临时表空间。有关更多信息，请参阅 rdsadmin.create_tablespace。

```
db2 "call rdsadmin.create_tablespace('database_name',\
    'db2montmptbsp','db2monbp',4096,1000,100,'T')"
```

4. 打开 `db2mon.sh` 脚本，并修改有关连接到数据库的行。

   a. 删除以下行。

   ```
   db2 -v connect to $dbName
   ```

   b. 将上一步中的行替换为下一行。在以下示例中，将 *master_username* 和 *master_password* 替换为您的 RDS for Db2 数据库实例的主用户名和主密码。

   ```
   db2 -v connect to $dbName user master_username using master_password
   ```

   c. 移除以下各行。

   ```
   db2 -v create bufferpool db2monbp

   db2 -v create user temporary tablespace db2montmptbsp bufferpool db2monbp

   db2 -v drop tablespace db2montmptbsp

   db2 -v drop bufferpool db2monbp
   ```

5. 运行 `db2mon.sh` 脚本以按指定的间隔输出报告。在以下示例中，将 *absolute_path* 替换为指向脚本文件的完整路径，将 *rds_database_alias* 替换为数据库的名称，并将 *seconds* 替换为报告生成之间的秒数（0 到 3600）。

```
absolute_path/db2mon.sh rds_database_alias seconds | tee -a db2mon.out
```

示例

以下示例显示脚本文件位于 home 目录下的 perf 目录中。

```
/home/db2inst1/sqllib/samples/perf/db2mon.sh rds_database_alias seconds | tee -a
 db2mon.out
```

6. 删除为 db2mon.sh 文件创建的缓冲池和表空间。在以下示例中，将 *master_username* 和 *master_password* 替换为您的 RDS for Db2 数据库实例的主用户名和主密码。将 *database_name* 替换为数据库的名称。有关更多信息，请参阅rdsadmin.drop_tablespace和rdsadmin.drop_bufferpool。

```
db2 connect to rdsadmin user master_username using master_password

db2 "call rdsadmin.drop_tablespace('database_name','db2montmptbsp')"

db2 "call rdsadmin.drop_bufferpool('database_name','db2monbp')"
```

管理存储

Db2 使用自动存储来管理数据库对象（例如表、索引和临时文件）的物理存储。自动存储允许 Db2 系统根据需要创建和管理存储路径，而不是手动分配存储空间和跟踪正在使用的存储路径。这可以简化 Db2 数据库的管理并降低因人为错误而导致错误的可能性。有关更多信息，请参阅 IBM Db2 文档中的自动存储。

使用 RDS for Db2，您可以通过自动扩展逻辑卷和文件系统来动态增加存储大小。有关更多信息，请参阅 使用 Amazon RDS 数据库实例的存储。

表空间的常见任务

您可以为 RDS for Db2 数据库创建、更改、重命名或删除表空间。创建、更改、重命名或删除表空间需要更高级别的 SYSADM 权限，而主用户无法获得该权限。而应使用 Amazon RDS 存储过程。

主题

- 创建表空间
- 更改表空间
- 重命名表空间

- 删除表空间
- 检查表空间的状态
- 返回有关表空间的详细信息
- 列出表空间的状态和存储组
- 列出表的表空间
- 列出表空间容器

创建表空间

要为您的 RDS for Db2 数据库创建表空间，请调用 `rdsadmin.create_tablespace` 存储过程。有关更多信息，请参阅 IBM Db2 文档中的 CREATE TABLESPACE 语句。

> **Important**
>
> 要创建表空间，您必须具有相同页面大小的缓冲池才能与表空间相关联。有关更多信息，请参阅 缓冲池的常见任务。

创建表空间

1. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到 `rdsadmin` 数据库。在以下示例中，用您自己的信息替换 *master_username* 和 *master_password*。

   ```
   db2 "connect to rdsadmin user master_username using master_password"
   ```

2. 通过调用 `rdsadmin.create_tablespace` 创建表空间。有关更多信息，请参阅 rdsadmin.create_tablespace。

   ```
   db2 "call rdsadmin.create_tablespace(
       'database_name',
       'tablespace_name',
       'buffer_pool_name',
       tablespace_initial_size,
       tablespace_increase_size,
       'tablespace_type')"
   ```

更改表空间

要更改 RDS for Db2 数据库的表空间，请调用 rdsadmin.alter_tablespace 存储过程。您可以使用此存储过程来更改表空间的缓冲池、降低高水位或使表空间联机。有关更多信息，请参阅 IBM Db2 文档中的 ALTER TABLESPACE 语句。

更改表空间

1. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到 rdsadmin 数据库。在以下示例中，用您自己的信息替换 *master_username* 和 *master_password*。

```
db2 "connect to rdsadmin user master_username using master_password"
```

2. 通过调用 rdsadmin.alter_tablespace 更改表空间。有关更多信息，请参阅 rdsadmin.alter_tablespace。

```
db2 "call rdsadmin.alter_tablespace(
    'database_name',
    'tablespace_name',
    'buffer_pool_name',
    buffer_pool_size,
    tablespace_increase_size,
    'max_size', 'reduce_max',
    'reduce_stop',
    'reduce_value',
    'lower_high_water',
    'lower_high_water_stop',
    'switch_online')"
```

重命名表空间

要更改 RDS for Db2 数据库的表空间名称，请调用 rdsadmin.rename_tablespace 存储过程。有关更多信息，请参阅 IBM Db2 文档中的 RENAME TABLESPACE 语句。

重命名表空间

1. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到 rdsadmin 数据库。在以下示例中，用您自己的信息替换 *master_username* 和 *master_password*。

```
db2 "connect to rdsadmin user master_username using master_password"
```

2. 通过调用 `rdsadmin.rename_tablespace` 重命名表空间。有关更多信息，包括对表空间命名的限制，请参阅rdsadmin.rename_tablespace。

```
db2 "call rdsadmin.rename_tablespace(
    'database_name',
    'source_tablespace_name',
    'target_tablespace_name')"
```

删除表空间

要删除 RDS for Db2 数据库的表空间，请调用 `rdsadmin.drop_tablespace` 存储过程。在删除表空间之前，请先删除表空间中的所有对象，例如表、索引或大型对象（LOB）。有关更多信息，请参阅 IBM Db2 文档中的删除表空间。

删除表空间

1. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到 `rdsadmin` 数据库。在以下示例中，用您自己的信息替换 *master_username* 和 *master_password*。

```
db2 "connect to rdsadmin user master_username using master_password"
```

2. 通过调用 `rdsadmin.drop_tablespace` 删除表空间。有关更多信息，请参阅 rdsadmin.drop_tablespace。

```
db2 "call rdsadmin.drop_tablespace(
    'database_name',
    'tablespace_name')"
```

检查表空间的状态

可以通过使用 `cast` 函数检查表空间的状态。

检查表空间的状态

1. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到您的 Db2 数据库。在以下示例中，用您自己的信息替换 *rds_database_alias*、*master_username* 和 *master_password*。

```
db2 connect to rds_database_alias user master_username using master_password
```

2. 返回摘要输出。

   对于摘要输出：

```
db2 "select cast(tbsp_id as smallint) as tbsp_id,
cast(tbsp_name as varchar(35)) as tbsp_name,
cast(tbsp_type as varchar(3)) as tbsp_type,
cast(tbsp_state as varchar(10)) as state,
cast(tbsp_content_type as varchar(8)) as contents from
 table(mon_get_tablespace(null,-1)) order by tbsp_id"
```

返回有关表空间的详细信息

可以通过使用 `cast` 函数返回有关一个成员或所有成员的表空间的信息。

返回有关表空间的详细信息

1. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到您的 Db2 数据库。在以下示例中，用您自己的信息替换 *rds_database_alias*、*master_username* 和 *master_password*。

```
db2 connect to rds_database_alias user master_username using master_password
```

2. 返回数据库中面向一个成员或面向所有成员的所有表空间的详细信息。

   对于一个成员：

```
db2 "select cast(member as smallint) as member,
cast(tbsp_id as smallint) as tbsp_id,
cast(tbsp_name as varchar(35)) as tbsp_name,
cast(tbsp_type as varchar(3)) as tbsp_type,
cast(tbsp_state as varchar(10)) as state,
cast(tbsp_content_type as varchar(8)) as contents,
cast(tbsp_total_pages as integer) as total_pages,
cast(tbsp_used_pages as integer) as used_pages,
cast(tbsp_free_pages as integer) as free_pages,
cast(tbsp_page_top as integer) as page_hwm,
cast(tbsp_page_size as integer) as page_sz,
cast(tbsp_extent_size as smallint) as extent_sz,
cast(tbsp_prefetch_size as smallint) as prefetch_sz,
cast(tbsp_initial_size as integer) as initial_size,
cast(tbsp_increase_size_percent as smallint) as increase_pct,
```

```
cast(storage_group_name as varchar(12)) as stogroup from
 table(mon_get_tablespace(null,-1)) order by member, tbsp_id "
```

对于所有成员：

```
db2 "select cast(member as smallint) as member
cast(tbsp_id as smallint) as tbsp_id,
cast(tbsp_name as varchar(35)) as tbsp_name,
cast(tbsp_type as varchar(3)) as tbsp_type,
cast(tbsp_state as varchar(10)) as state,
cast(tbsp_content_type as varchar(8)) as contents,
cast(tbsp_total_pages as integer) as total_pages,
cast(tbsp_used_pages as integer) as used_pages,
cast(tbsp_free_pages as integer) as free_pages,
cast(tbsp_page_top as integer) as page_hwm,
cast(tbsp_page_size as integer) as page_sz,
cast(tbsp_extent_size as smallint) as extent_sz,
cast(tbsp_prefetch_size as smallint) as prefetch_sz,
cast(tbsp_initial_size as integer) as initial_size,
cast(tbsp_increase_size_percent as smallint) as increase_pct,
cast(storage_group_name as varchar(12)) as stogroup from
 table(mon_get_tablespace(null,-2)) order by member, tbsp_id "
```

列出表空间的状态和存储组

可以通过运行 SQL 语句来列出表空间的状态和存储组。

要列出表空间的状态和存储组，请运行以下 SQL 语句：

```
db2 "SELECT varchar(tbsp_name, 30) as tbsp_name,
                  varchar(TBSP_STATE, 30) state,
                  tbsp_type,
                  varchar(storage_group_name,30) storage_group
FROM TABLE(MON_GET_TABLESPACE('',-2)) AS t"
```

列出表的表空间

可以通过运行 SQL 语句来列出表的表空间。

要列出表的表空间，运行以下 SQL 语句。在以下示例中，将 *SCHEMA_NAME* 和 *TABLE_NAME* 替换为架构和表的名称：

```
db2 "SELECT
   VARCHAR(SD.TBSPACE,30) AS DATA_SPACE,
   VARCHAR(SL.TBSPACE,30) AS LONG_SPACE,
   VARCHAR(SI.TBSPACE,30) AS INDEX_SPACE
 FROM
   SYSCAT.DATAPARTITIONS P
   JOIN SYSCAT.TABLESPACES SD ON SD.TBSPACEID = P.TBSPACEID
   LEFT JOIN SYSCAT.TABLESPACES SL ON SL.TBSPACEID = P.LONG_TBSPACEID
   LEFT JOIN SYSCAT.TABLESPACES SI ON SI.TBSPACEID = P.INDEX_TBSPACEID
 WHERE
    TABSCHEMA = 'SCHEMA_NAME'
 AND TABNAME    = 'TABLE_NAME'"
```

列出表空间容器

可以通过使用 `cast` 命令列出所有表空间容器或特定的表空间容器。

列出表空间的表空间容器

1. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到您的 Db2 数据库。在以下示例中，用您自己的信息替换 *rds_database_alias*、*master_username* 和 *master_password*：

   ```
   db2 connect to rds_database_alias user master_username using master_password
   ```

2. 返回数据库中所有表空间容器或特定表空间容器的列表。

   对于所有表空间容器：

   ```
   db2 "select cast(member as smallint) as member,
   cast(tbsp_name as varchar(35)) as tbsp_name,
   cast(container_id as smallint) as id,
   cast(container_name as varchar(60)) as container_path, container_type as type from
    table(mon_get_container(null,-2)) order by member,tbsp_id,container_id"
   ```

   对于特定的表空间容器：

   ```
   db2 "select cast(member as smallint) as member,
   cast(tbsp_name as varchar(35)) as tbsp_name,
   cast(container_id as smallint) as id,
   cast(container_name as varchar(60)) as container_path, container_type as type from
    table(mon_get_container('TBSP_1',-2)) order by member, tbsp_id,container_id"
   ```

将 Amazon RDS for Db2 数据库实例与 Amazon S3 集成

您可以使用 Amazon RDS 存储过程在您的 Amazon RDS for Db2 数据库实例和 Amazon Simple Storage Service（Amazon S3）存储桶之间传输文件。有关更多信息，请参阅 Amazon RDS for Db2 存储过程参考。

> **Note**
>
> 数据库实例和 Amazon S3 存储桶必须位于同一 Amazon Web Services 区域。

要让 RDS for Db2 与 Amazon S3 集成，您的数据库实例必须可以访问 RDS for Db2 所在的 Amazon S3 存储桶。如果您当前没有 S3 存储桶，请创建存储桶。

主题

步骤 1：创建 IAM policy

在此步骤中，您将创建一个 Amazon Identity and Access Management（IAM）策略，该策略具有将文件从您的 Amazon S3 存储桶传输到您的 RDS 数据库实例所需的权限。此步骤假定您已创建了 S3 桶。有关更多信息，请参阅《Amazon S3 用户指南》中的创建存储桶。

创建策略前，请记下以下信息：

- 存储桶的 Amazon 资源名称（ARN）。
- 您的 Amazon Key Management Service（Amazon KMS）密钥的 ARN，如果您的存储桶使用 SSE-KMS 或 SSE-S3 加密。

创建包含以下权限的 IAM policy：

```
"kms:GenerateDataKey",
"kms:Decrypt",
"s3:PutObject",
"s3:GetObject",
"s3:AbortMultipartUpload",
```

```
"s3:ListBucket",
"s3:DeleteObject",
"s3:GetObjectVersion",
"s3:ListMultipartUploadParts"
```

您可以使用 Amazon Web Services Management Console或 Amazon Command Line Interface（Amazon CLI）创建 IAM policy。

控制台

创建 IAM 策略以允许 Amazon RDS 访问 Amazon S3 存储桶

1. 登录到 Amazon Web Services Management Console，然后通过以下网址打开 IAM 控制台：https://console.aws.amazon.com/iam/。
2. 在导航窗格中，选择策略。
3. 选择创建策略，然后选择 JSON。
4. 按服务添加操作。要将文件从 Amazon S3 存储桶传输到 Amazon RDS，必须选择存储桶权限和对象权限。
5. 展开 Resources（资源）。您必须指定您的存储桶和对象资源。
6. 选择下一步。
7. 对于策略名称，输入此策略的名称。
8. （可选）对于描述，输入此策略的描述。
9. 选择 创建策略。

Amazon CLI

创建 IAM 策略以允许 Amazon RDS 访问 Amazon S3 存储桶

1. 运行 create-policy 命令。在以下示例中，将 *iam_policy_name* 和 *s3_bucket_name* 替换为 IAM policy 的名称以及 RDS for Db2 数据库所在 Amazon S3 存储桶的名称。

   对于 Linux、macOS 或 Unix：

   ```
   aws iam create-policy \
       --policy-name iam_policy_name \
       --policy-document '{
         "Version": "2012-10-17",
         "Statement": [
   ```

```
        {
          "Effect": "Allow",
          "Action": [
            "kms:GenerateDataKey",
            "kms:Decrypt",
            "s3:PutObject",
            "s3:GetObject",
            "s3:AbortMultipartUpload",
            "s3:ListBucket",
            "s3:DeleteObject",
            "s3:GetObjectVersion",
            "s3:ListMultipartUploadParts"
          ],
          "Resource": [
            "arn:aws:s3:::s3_bucket_name/*",
            "arn:aws:s3:::s3_bucket_name"
          ]
        }
      ]
    }'
```

对于 Windows：

```
aws iam create-policy ^
    --policy-name iam_policy_name ^
    --policy-document '{
      "Version": "2012-10-17",
      "Statement": [
        {
          "Effect": "Allow",
            "Action": [
              "s3:PutObject",
              "s3:GetObject",
              "s3:AbortMultipartUpload",
              "s3:ListBucket",
              "s3:DeleteObject",
              "s3:GetObjectVersion",
              "s3:ListMultipartUploadParts"
            ],
            "Resource": [
              "arn:aws:s3:::s3_bucket_name/*",
              "arn:aws:s3:::s3_bucket_name"
            ]
```

```
            }
        ]
    }'
```

2. 创建策略后，请记下策略的 ARN。您需要步骤 2：创建 IAM 角色并附加您的 IAM policy的 ARN。

有关创建 IAM policy 的更多信息，请参阅《IAM 用户指南》中的创建 IAM policy。

步骤 2：创建 IAM 角色并附加您的 IAM policy

此步骤假定您已在步骤 1：创建 IAM policy中创建了 IAM 策略。在此步骤中，您将为 RDS for Db2 数据库实例创建一个 IAM 角色，然后将 IAM policy 附加到该角色。

您可以使用 Amazon Web Services Management Console或 Amazon CLI 为数据库实例创建 IAM 角色。

控制台

创建 IAM 角色并向其附加 IAM policy

1. 登录 Amazon Web Services Management Console，然后通过以下网址打开 IAM 控制台：https://console.aws.amazon.com/iam/。
2. 在导航窗格中，选择角色。
3. 选择 Create role（创建角色）。
4. 对于可信实体类型，请选择 Amazon Web Services 服务。
5. 对于服务或使用案例，选择 RDS，然后选择 RDS – 向数据库添加角色。
6. 选择下一步。
7. 对于权限策略，搜索并选择您创建的 IAM policy 的名称。
8. 选择下一步。
9. 对于 Role name（角色名称），输入一个角色名称。
10. （可选）在描述中，输入新角色的描述。
11. 选择 Create role（创建角色）。

Amazon CLI

创建 IAM 角色并向其附加 IAM policy

1. 运行 create-role 命令。在以下示例中，将 *iam_role_name* 替换为您的 IAM 角色的名称。

对于 Linux、macOS 或 Unix：

```
aws iam create-role \
    --role-name iam_role_name \
    --assume-role-policy-document '{
      "Version": "2012-10-17",
      "Statement": [
        {
          "Effect": "Allow",
          "Principal": {
            "Service": "rds.amazonaws.com"
          },
          "Action": "sts:AssumeRole"
        }
      ]
    }'
```

对于 Windows：

```
aws iam create-role ^
    --role-name iam_role_name ^
    --assume-role-policy-document '{
      "Version": "2012-10-17",
      "Statement": [
        {
          "Effect": "Allow",
          "Principal": {
            "Service": "rds.amazonaws.com"
          },
          "Action": "sts:AssumeRole"
        }
      ]
    }'
```

2. 创建角色之后，记下该角色的 ARN。您需要步骤 3：将您的 IAM 角色添加到 RDS for Db2 数据库实例的 ARN。
3. 运行 attach-role-policy 命令。在以下示例中，将 *iam_policy_arn* 替换为您在步骤 1：创建 IAM policy中创建的 IAM policy。将 *iam_role_name* 替换为刚创建的 IAM 角色的名称。

   对于 Linux、macOS 或 Unix：

```
aws iam attach-role-policy \
    --policy-arn iam_policy_arn \
    --role-name iam_role_name
```

对于 Windows：

```
aws iam attach-role-policy ^
    --policy-arn iam_policy_arn ^
    --role-name iam_role_name
```

有关更多信息，请参阅《IAM 用户指南》中的创建向 IAM 用户委派权限的角色。

步骤 3：将您的 IAM 角色添加到 RDS for Db2 数据库实例

在此步骤中，您将 IAM 角色添加到 RDS for Db2 数据库实例。请注意以下要求：

- 您必须具有访问 IAM 角色的权限，并向此角色附加所需的 Amazon S3 权限策略。
- 您一次只能将一个 IAM 角色与 RDS for Db2 数据库实例关联。
- RDS for Db2 数据库实例必须处于可用状态。

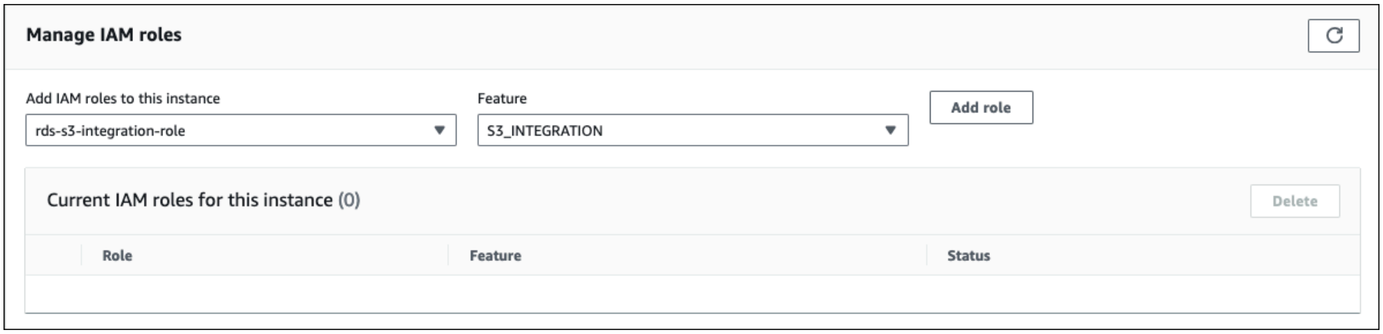
您可以使用 Amazon Web Services Management Console或 Amazon CLI 向数据库实例添加 IAM 角色。

控制台

将 IAM 角色添加到 RDS for Db2 数据库实例

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases（数据库）。
3. 选择 RDS for Db2 数据库实例名称。
4. 在 Connectivity & security（连接性和安全性）选项卡上，向下滚动到页面底部的 Manage IAM roles（管理 IAM 角色）部分。
5. 对于向此实例添加 IAM 角色，选择您在步骤 2：创建 IAM 角色并附加您的 IAM policy中创建的角色。
6. 对于 Feature (功能)，选择 S3_INTEGRATION。

7. 选择添加角色。



Amazon CLI

要将 IAM 角色添加到 RDS for Db2 数据库实例，请运行 add-role-to-db-instance 命令。在以下示例中，将 *db_instance_name* 和 *iam_role_arn* 替换为数据库实例的名称和您在[步骤 2：创建 IAM 角色并附加您的 IAM policy](步骤 2：创建 IAM 角色并附加您的 IAM policy)中创建的 IAM 角色的 ARN。

对于 Linux、macOS 或 Unix：

```
aws rds add-role-to-db-instance \
    --db-instance-identifier db_instance_name \
    --feature-name S3_INTEGRATION \
    --role-arn iam_role_arn \
```

对于 Windows：

```
aws rds add-role-to-db-instance ^
    --db-instance-identifier db_instance_name ^
    --feature-name S3_INTEGRATION ^
    --role-arn iam_role_arn ^
```

要确认该角色已成功添加到 RDS for Db2 数据库实例，请运行 describe-db-instances 命令。在以下示例中，将 *db_instance_name* 替换为数据库实例的名称。

对于 Linux、macOS 或 Unix：

```
aws rds describe-db-instances \
    --filters "Name=db-instance-id,Values=db_instance_name" \
    --query 'DBInstances[].AssociatedRoles'
```

对于 Windows：

```
aws rds describe-db-instances ^
    --filters "Name=db-instance-id,Values=db_instance_name" ^
    --query 'DBInstances[].AssociatedRoles'
```

此命令生成类似于以下示例的输出：

```
[
    [
        {
            "RoleArn": "arn:aws:iam::0123456789012:role/rds-db2-s3-role",
            "FeatureName": "S3_INTEGRATION",
            "Status": "ACTIVE"
        }
    ]
]
```

将数据迁移到 Amazon RDS for Db2

您可以使用 Amazon 或原生 Db2 工具将自行管理的 Db2 数据库迁移到 Amazon RDS for Db2。

有关使用 Amazon 服务从 Db2 数据库迁移到 Amazon RDS for Db2 的信息，请参阅使用 Amazon 服务将数据从 Db2 迁移到 Amazon RDS for Db2。

有关使用原生 Db2 工具从 Db2 数据库迁移到 Amazon RDS for Db2 的信息，请参阅使用原生 Db2 工具将数据从 Db2 迁移到 Amazon RDS for Db2。

使用 Amazon 服务将数据从 Db2 迁移到 Amazon RDS for Db2

在 Amazon RDS 中，可以通过多种方式将数据从 Db2 数据库迁移到 Amazon RDS for Db2。您可以将 Db2 数据库从 Linux、AIX 或 Windows 环境一次性迁移到 Amazon RDS for Db2。为了显著缩短停机时间，您可以执行停机时间接近零的迁移。可以通过将数据保存到 Amazon S3，然后一次将一个表加载到 Db2 数据库中，从而迁移数据。此外，您还可以通过复制执行同步迁移或使用 Amazon Database Migration Service。

要针对基于 Linux 的 Db2 数据库执行一次性迁移，Amazon RDS 仅支持离线和在线备份。Amazon RDS 不支持增量和 Delta 备份。要针对基于 Linux 的 Db2 数据库进行几乎零停机的迁移，Amazon RDS 需要在线备份。建议您使用在线备份实现几乎零停机的迁移，使用离线备份进行可以应对停机的迁移。

主题

针对 Amazon RDS for Db2 从 Linux 迁移到 Linux

通过这种迁移方法，您可以将自行管理的 Db2 数据库备份到 Amazon S3 存储桶。然后，使用 Amazon RDS 存储过程将 Db2 数据库还原到 Amazon RDS for Db2 数据库实例。有关使用 Amazon S3 的更多信息，请参阅将 Amazon RDS for Db2 数据库实例与 Amazon S3 集成。

RDS for Db2 的备份和还原遵循 IBM Db2 支持的升级路径和限制。有关更多信息，请参阅 IBM Db2 文档中的 Supported upgrade paths for Db2 servers 和 Upgrade restrictions for Db2 servers。

主题

使用原生还原的限制和建议

以下限制和建议适用于使用原生还原：

- Amazon RDS 仅支持迁移与支持的 RDS for Db2 版本相匹配的 Db2 本地版本。有关支持的版本的更多信息，请参阅 Amazon RDS 上支持的 Db2 次要版本。
- Amazon RDS 仅支持用于原生还原的离线和在线备份。Amazon RDS 不支持增量或 Delta 备份。
- 您无法从与 RDS for Db2 数据库实例所在区域不同的 Amazon Web Services 区域中的 Amazon S3 存储桶进行还原。
- Amazon S3 将上传到 Amazon S3 存储桶的文件大小限制为 5TB。如果数据库备份文件超过 5TB，则将备份文件拆分为较小的文件。
- Amazon RDS 不支持非防护外部例程、增量还原或 Delta 还原。
- 您无法从加密的源数据库中还原，但可以还原到加密的 Amazon RDS 数据库实例。

还原数据库时，将复制备份，然后在 RDS for Db2 数据库实例上提取。我们建议您为 RDS for Db2 数据库实例预调配存储空间，其大小等于或大于备份大小加上磁盘上原始数据库大小之和。

还原数据库的最大大小是支持的最大数据库大小减去备份的大小。例如，如果支持的最大数据库大小为 64TiB，备份的大小为 30TiB，则还原数据库的最大大小为 34TiB。

```
64 TiB - 30 TiB = 34 TiB
```

将数据库备份到 Amazon S3

要在 Amazon S3 上备份数据库，您需要以下 Amazon 组件：

- 用于存储备份文件的 Amazon S3 存储桶：上传您要迁移到 Amazon RDS 的所有备份文件。建议您使用离线备份进行可以应对停机的迁移。如果您已有一个 S3 存储桶，则可以使用该存储桶。如果没有 S3 存储桶，请参阅《Amazon S3 用户指南》中的创建存储桶。

> **Note**
>
> 如果数据库很大，需要很长时间才能传输到 S3 存储桶，则可以订购 Amazon Snow Family 设备并要求 Amazon 执行备份。将文件复制到设备并将其返还给 Snow 系列团队后，该团队会将备份的映像传输到您的 S3 存储桶。有关更多信息，请参阅 Amazon Snow Family 文档。

- 访问 S3 存储桶的 IAM 角色：如果已经有一个 IAM 角色，则可以使用该角色。如果您还没有角色，请参阅步骤 2：创建 IAM 角色并附加您的 IAM policy。
- 一个 IAM 策略，其中包含附加到 IAM 角色的信任关系和权限：有关更多信息，请参阅步骤 1：创建 IAM policy。
- 添加到 RDS for Db2 数据库实例的 IAM 角色：有关更多信息，请参阅步骤 3：将您的 IAM 角色添加到 RDS for Db2 数据库实例。

创建默认自动存储组

源数据库必须具有默认自动存储组。如果数据库没有默认自动存储组，则必须创建一个这类组。

创建默认自动存储组

1. 连接到您的源数据库。在以下示例中，将 *source_database* 替换为数据库的名称。

```
db2 connect to source_database
```

2. 创建自动存储组并将其设置为默认存储组。在以下示例中，将 *storage_path* 替换为存储组所在位置的绝对路径。

```
db2 "create stogroup IBMSTOGROUP ON storage_path set as default"
```

3. 终止后端进程。

```
db2 terminate
```

4. 停用数据库并停止所有数据库服务。在以下示例中，将 *source_database* 替换为您为其创建了存储组的数据库的名称。

```
db2 deactivate db source_database
```

5. 备份数据库。在以下示例中，将 *source_database* 替换为您为其创建了存储组的数据库的名称。将 *file_system_path* 替换为要在其中备份数据库的绝对路径。

```
db2 backup database source_database to file_system_path
```

还原 Db2 数据库

在 Amazon S3 上备份数据库并创建自动存储组后，就可以将 Db2 数据库还原到 RDS for Db2 数据库实例了。

将 Db2 数据库还原到 RDS for Db2 数据库实例

1. 连接到 RDS for Db2 数据库实例。有关更多信息，请参阅 连接到 Db2 数据库实例。
2. （可选）要确保为您的数据库配置了用于还原操作的最佳设置，您可以调用 the section called “rdsadmin.show_configuration” 来检查 `RESTORE_DATABASE_PARALLELISM` 和 `RESTORE_DATABASE_NUM_BUFFERS` 的值。根据需要，调用 the section called “rdsadmin.set_configuration” 来更改这些值。显式设置这些值可以提高还原包含大量数据的数据库时的性能。
3. 通过调用 `rdsadmin.restore_database` 还原您的数据库。有关更多信息，请参阅 rdsadmin.restore_database。

针对 Amazon RDS for Db2 从 Linux 迁移到 Linux，停机时间接近零

使用这种迁移方法，您可以将基于 Linux 的 Db2 数据库从自行管理的 Db2 数据库（源）迁移到 Amazon RDS for Db2。这种方法可使应用程序或用户最大限度地减少甚至没有中断或停机时间。这种方法可以备份您的数据库并通过日志重播进行还原，这有助于防止中断正在进行的操作并提供数据库的高可用性。

为了实现停机时间接近零的迁移，RDS for Db2 实施带日志重播的还原。此方法对自行管理的基于 Linux 的 Db2 数据库进行备份，然后在 RDS for Db2 服务器上还原它。使用 Amazon RDS 存储过程，您可以应用后续的事务日志来使数据库保持最新。

主题

- 将数据库备份到 Amazon S3
- 创建默认自动存储组
- 迁移 Db2 数据库

停机时间接近零的迁移的限制和建议

以下限制和建议适用于使用停机时间接近零的迁移。

- Amazon RDS 需要在线备份才能实现近乎零停机的迁移。这是因为在您上传存档的事务日志时，Amazon RDS 会使您的数据库处于前滚挂起状态。有关更多信息，请参阅 the section called “迁移 Db2 数据库”。
- 您无法从与 RDS for Db2 数据库实例所在区域不同的 Amazon Web Services 区域中的 Amazon S3 存储桶进行还原。
- Amazon S3 将上传到 S3 存储桶的文件大小限制为 5TB。如果数据库备份文件超过 5TB，则将备份文件拆分为较小的文件。
- Amazon RDS 不支持非防护外部例程、增量还原或 Delta 还原。
- 您无法从加密的源数据库中还原，但可以还原到加密的 Amazon RDS 数据库实例。

还原数据库时，Amazon RDS 将复制备份，然后在 RDS for Db2 数据库实例上提取该备份。建议您为 RDS for Db2 数据库实例预调配存储空间，其大小等于或大于备份大小加上磁盘上原始数据库大小之和。

还原数据库的最大大小是支持的最大数据库大小减去备份的大小。例如，如果支持的最大数据库大小为 64TiB，备份的大小为 30TiB，则还原数据库的最大大小为 34TiB。

```
64 TiB - 30 TiB = 34 TiB
```

将数据库备份到 Amazon S3

要在 Amazon S3 上备份数据库，您需要以下 Amazon 组件：

- 用于存储备份文件的 Amazon S3 存储桶：上传您要迁移到 Amazon RDS 的所有备份文件。Amazon RDS 需要在线备份才能实现近乎零停机的迁移。如果您已有一个 S3 存储桶，则可以使用该存储桶。如果没有 S3 存储桶，请参阅《Amazon S3 用户指南》中的创建存储桶。

> **Note**
>
> 如果数据库很大，需要很长时间才能传输到 S3 存储桶，则可以订购 Amazon Snow Family 设备并要求 Amazon 执行备份。将文件复制到设备并将其返还给 Snow 系列团队后，该团队会将备份的映像传输到您的 S3 存储桶。有关更多信息，请参阅 Amazon Snow Family 文档。

- 用于访问 S3 存储桶的 IAM 角色：如果您已经有一个 Amazon Identity and Access Management（IAM）角色，则可以使用该角色。如果您还没有角色，请参阅步骤 2：创建 IAM 角色并附加您的 IAM policy。
- 一个 IAM 策略，其中包含附加到 IAM 角色的信任关系和权限：有关更多信息，请参阅步骤 1：创建 IAM policy。
- 添加到 RDS for Db2 数据库实例的 IAM 角色：有关更多信息，请参阅步骤 3：将您的 IAM 角色添加到 RDS for Db2 数据库实例。

创建默认自动存储组

源数据库必须具有默认自动存储组。如果数据库没有默认自动存储组，则必须创建一个这类组。

创建默认自动存储组

1. 连接到您的源数据库。在以下示例中，将 *source_database* 替换为数据库的名称。

   ```
   db2 connect to source_database
   ```

2. 创建自动存储组并将其设置为默认存储组。在以下示例中，将 *storage_path* 替换为存储组所在位置的绝对路径。

   ```
   db2 "create stogroup IBMSTOGROUP ON storage_path set as default"
   ```

3. 终止后端进程。

   ```
   db2 terminate
   ```

迁移 Db2 数据库

在 Amazon S3 上备份数据库并创建自动存储组后，就可以将 Db2 数据库迁移到 RDS for Db2 数据库实例了。

执行停机时间接近零的迁移

1. 对源数据库进行联机备份。有关更多信息，请参阅 IBM Db2 文档中的 BACKUP DATABASE 命令。
2. 将数据库备份复制到 Amazon S3 存储桶。有关使用 Amazon S3 的更多信息，请参阅 Amazon Simple Storage Service 用户指南。
3. 使用 RDS for Db2 数据库实例的 *master_username* 和 *master_password* 连接到 `rdsadmin` 服务器。

```
db2 connect to rdsadmin user master_username using master_password
```

4. （可选）要确保为您的数据库配置了用于还原操作的最佳设置，您可以调用 the section called “rdsadmin.show_configuration” 来检查 `RESTORE_DATABASE_PARALLELISM` 和 `RESTORE_DATABASE_NUM_BUFFERS` 的值。根据需要，调用 the section called “rdsadmin.set_configuration” 来更改这些值。显式设置这些值可以提高还原包含大量数据的数据库时的性能。
5. 通过调用 `rdsadmin.restore_database`，在 RDS for Db2 服务器上还原备份。将 `backup_type` 设置为 `ONLINE`。有关更多信息，请参阅 rdsadmin.restore_database。
6. 将源服务器中的归档日志复制到 S3 存储桶。有关更多信息，请参阅 IBM Db2 文档中的归档日志记录。
7. 根据需要，通过调用 `rdsadmin.rollforward_database` 多次应用归档日志。将 `complete_rollforward` 设置为 `FALSE` 可使数据库保持 `ROLL-FORWARD PENDING` 状态。有关更多信息，请参阅 rdsadmin.rollforward_database。
8. 应用所有归档日志后，通过调用 `rdsadmin.complete_rollforward` 使数据库联机。有关更多信息，请参阅 rdsadmin.complete_rollforward。
9. 通过更新数据库的应用程序端点或更新 DNS 端点以将流量重定向到 RDS for Db2 服务器，将应用程序连接切换到 RDS for Db2 服务器。还可以使用 RDS for Db2 数据库端点在自行管理的 Db2 数据库上使用 Db2 自动客户端重新路由功能。有关更多信息，请参阅 IBM Db2 文档中的 Automatic client reroute description and setup。
10. （可选）关闭源数据库。

针对 Amazon RDS for Db2 从 Linux 同步迁移到 Linux

使用这种迁移方法，您可以在自行管理的 Db2 数据库和 Amazon RDS for Db2 数据库实例之间设置复制。对自行管理的数据库所做的更改几乎实时地复制到 RDS for Db2 数据库实例。这种方法可以提供持续可用性，并最大限度地减少迁移过程中的停机时间。

针对 Amazon RDS for Db2 从 AIX 或 Windows 迁移到 Linux

通过这种迁移方法，您可以使用原生 Db2 工具将自行管理的 Db2 数据库备份到 Amazon S3 存储桶。原生 Db2 工具包括 `export` 实用程序、`db2move` 系统命令或 `db2look` 系统命令。您的 Db2 数据库可以是自行管理的，也可以在 Amazon Elastic Compute Cloud（Amazon EC2）中管理。您可以将数据从 AIX 或 Windows 系统移到 Amazon S3 存储桶中。然后，使用 Db2 客户端将数据直接从 S3 存储桶加载到 Amazon RDS for Db2 数据库。停机时间因数据库的大小而异。有关使用 Amazon S3 的更多信息，请参阅[将 Amazon RDS for Db2 数据库实例与 Amazon S3 集成]。

将 Db2 数据库迁移到 RDS for Db2

1. 准备备份数据库。配置足够的存储量，以便在自行管理的 Db2 系统上保留备份。
2. 备份您的数据库。
   a. 运行 [db2look 系统命令]以提取所有对象的数据定义语言（DDL）文件。
   b. 运行 [Db2 导出实用程序]、[db2move 系统命令]或 [CREATE EXTERNAL TABLE 语句]，以将 Db2 表数据卸载到 Db2 系统上的存储中。
3. 将备份移动到 Amazon S3 存储桶。有关更多信息，请参阅 [将 Amazon RDS for Db2 数据库实例与 Amazon S3 集成]。

   > **Note**
   >
   > 如果数据库很大，需要很长时间才能传输到 S3 存储桶，则可以订购 Amazon Snow Family 设备并要求 Amazon 执行备份。将文件复制到设备并将其返还给 Snow 系列团队后，该团队会将备份的映像传输到您的 S3 存储桶。有关更多信息，请参阅 [Amazon Snow Family 文档]。

4. 使用 Db2 客户端将数据直接从 S3 存储桶加载到 RDS for Db2 数据库。有关更多信息，请参阅 [使用 Amazon S3 进行迁移]。

通过 Amazon S3 将 Db2 数据迁移到 Amazon RDS for Db2

通过这种迁移方法，您首先将单个表中的数据保存到您放入 Amazon S3 存储桶中的数据文件。然后，可以使用 LOAD command 将该数据文件中的数据加载到 Amazon RDS for Db2 数据库的表中。有关使用 Amazon S3 的更多信息，请参阅将 Amazon RDS for Db2 数据库实例与 Amazon S3 集成。

主题

将数据保存到 Amazon S3

要将单个表中的数据保存到 Amazon S3，请使用数据库实用程序来将数据从数据库管理系统（DBMS）提取到 CSV 文件中。然后，将数据文件上传到 Amazon S3。

要在 Amazon S3 上存储数据文件，您需要以下 Amazon 组件：

- 用于存储备份文件的 Amazon S3 存储桶：如果您已有一个 S3 存储桶，则可以使用该存储桶。如果没有 S3 存储桶，请参阅《Amazon S3 用户指南》中的创建存储桶。
- 访问 S3 存储桶的 IAM 角色：如果已经有一个 IAM 角色，则可以使用该角色。如果您还没有角色，请参阅步骤 2：创建 IAM 角色并附加您的 IAM policy。
- 一个 IAM 策略，其中包含附加到 IAM 角色的信任关系和权限：有关更多信息，请参阅步骤 1：创建 IAM policy。
- 添加到 RDS for Db2 数据库实例的 IAM 角色：有关更多信息，请参阅步骤 3：将您的 IAM 角色添加到 RDS for Db2 数据库实例。

将数据加载到 RDS for Db2 表中

在将数据文件保存到 Amazon S3 后，可以将这些文件中的数据加载到 RDS for Db2 数据库实例上的各个表中。

将 Db2 表数据加载到 RDS for Db2 数据库表中

1. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到 `rdsadmin` 数据库。在以下示例中，用您自己的信息替换 *master_username* 和 *master_password*。

   ```
   db2 connect to rdsadmin user master_username using master_password
   ```

2. 对存储访问权限别名进行编目，该别名指向存储已保存文件的 Amazon S3 存储桶。记下该别名的名称，以供在下一个步骤中使用。如果您计划从存储在同一 Amazon S3 存储桶中的数据文件加载多个表，则只需执行一次此步骤。

   以下示例对名为 *my_s3_alias* 的别名进行编目，该别名向名为 *jorge_souza* 的用户授予访问名为 *amzn-s3-demo-bucket* 的存储桶的权限。

```
db2 "call rdsadmin.catalog_storage_access(?, 'my_s3_alias', 'amzn-s3-demo-bucket',
 'USER', 'jorge_souza')"
```

   有关此存储过程的更多信息，请参阅the section called “rdsadmin.catalog_storage_access”。

3. 使用指向您的 Amazon S3 存储桶的存储访问权限别名运行 LOAD 命令。

   Note

   如果 LOAD 命令返回错误，则可能需要为 Amazon S3 创建 VPC 网关端点，并将出站规则添加到安全组。有关更多信息，请参阅 the section called “文件 I/O 错误”。

   以下示例将名为 *my_s3_datafile.csv* 的数据文件中的数据加载到名为 *my_db2_table* 的表中。该示例假设数据文件位于名为 *my_s3_alias* 的别名所指向的 Amazon S3 存储桶中。

```
db2 "load from db2remote://my_s3_alias//my_s3_datafile.csv of DEL insert
 into my_db2_table";
```

   以下示例将名为 *my_table1_export.ixf* 的数据文件中的 LOB 加载到名为 *my_db2_table* 的表中。该示例假设数据文件位于名为 *my_s3_alias* 的别名所指向的 Amazon S3 存储桶中。

```
db2 "call sysproc.admin_cmd('load from
 "db2remote://my_s3_alias//my_table1_export.ixf" of ixf
        lobs from "db2remote://my_s3_alias//" xml from "db2remote://my_s3_alias//"
        modified by lobsinfile implicitlyhiddeninclude identityoverride
 generatedoverride periodoverride transactionidoverride
        messages on server
        replace into "my_schema"."my_db2_table"
                              nonrecoverable
        indexing mode incremental allow no access')"
```

   对 Amazon S3 存储桶中要加载到 RDS for Db2 数据库实例的表中的每个数据文件重复此步骤。

有关 LOAD 命令的更多信息，请参阅 LOAD command。

使用 Amazon Database Migration Service（Amazon DMS）迁移到 Amazon RDS for Db2

您可以使用 Amazon DMS 进行一次性迁移，然后从 Linux、Unix（例如 AIX）和 Windows 上的 Db2 同步到 Amazon RDS for Db2。有关更多信息，请参阅什么是 Amazon Database Migration Service？。

使用原生 Db2 工具将数据从 Db2 迁移到 Amazon RDS for Db2

可以使用多种原生 Db2 工具、实用程序和命令，直接将数据从 Db2 数据库移到 Amazon RDS for Db2 数据库。要使用这些原生 Db2 工具，您必须能够将您的客户端计算机连接到 RDS for Db2 数据库实例。有关更多信息，请参阅 将客户端计算机连接到 Amazon RDS for Db2 数据库实例。

> **Note**
> 移动数据的另一种方法是先将其保存到 Amazon S3 存储桶，然后使用 LOAD 命令将该数据传输到 RDS for Db2 数据库的表中。由于 RDS for Db2 和 S3 之间的网络连接良好，因此在迁移大量数据时，此方法可提供最佳性能。有关更多信息，请参阅 the section called “使用 Amazon S3 进行迁移”。

工具名称	应用场景	限制
db2look	将元数据从自行管理的 Db2 数据库复制到 RDS for Db2 数据库。	• 必须修改用于创建缓冲池、创建表空间和创建角色的语法，使其与 RDS for Db2 存储过程 使用的语法相匹配。
IMPORT 命令	将小型表和包含大型对象（LOB）的表从客户端计算机迁移到 RDS for Db2 数据库实例。	• 由于需要执行 INSERT 和 DELETE 日志记录操作，速度比 LOAD 实用程序慢。 • 网络带宽有限，性能不佳。

工具名称	应用场景	限制
INGEST 实用程序	持续将客户端计算机上没有大型对象（LOB）的文件和管道中的数据流式传输到 RDS for Db2 数据库实例。支持 `INSERT` 和 `MERGE` 操作。	• 无法流式传输包含 LOB 的数据文件。改用 `IMPORT` 命令。 • 需要在自行管理的 Db2 数据库和 RDS for Db2 数据库之间建立连接。
INSERT 命令	将小型表中的数据从自行管理的 Db2 数据库复制到 RDS for Db2 数据库。	• 需要在自行管理的 Db2 数据库和 RDS for Db2 数据库之间建立连接。 • 网络带宽有限，性能不佳。
LOAD CLIENT 命令	将没有 大型对象（LOB）的小型表从客户端计算机迁移到 RDS for Db2 数据库实例。	• 无法迁移包含 LOB 的数据文件。改用 `IMPORT` 命令。 • 网络带宽有限，性能不佳。

将客户端计算机连接到 Amazon RDS for Db2 数据库实例

要使用任何原生 Db2 工具将数据从 Db2 数据库移动到 Amazon RDS for Db2 数据库，必须先将您的客户端计算机连接到 RDS for Db2 数据库实例。

客户端计算机可以是以下任一项：

- Linux、Windows 或 macOS 上的 Amazon Elastic Compute Cloud（Amazon EC2）实例 实例应与 RDS for Db2 数据库实例 Amazon Cloud9 或 Amazon CloudShell 位于同一虚拟私有云（VPC）中。
- Amazon EC2 实例中自行管理的 Db2 实例。实例应位于同一 VPC 中。
- Amazon EC2 实例中自行管理的 Db2 实例。如果您启用了 VPC 对等连接，这些实例可以位于不同的 VPC 中。有关更多信息，请参阅《Amazon Virtual Private Cloud VPC 对等连接指南》中的创建 VPC 对等连接。
- 在自行管理的环境中运行 Linux、Windows 或 macOS 的本地计算机。您必须拥有与 RDS for Db2 的公有连接，或者在自行管理的 Db2 实例和 Amazon 之间启用 VPN 连接。

要将您的客户端计算机连接到 RDS for Db2 数据库实例，请使用 IBM Db2 Data Management Console 登录您的客户端计算机。有关更多信息，请参阅创建 Amazon RDS 数据库实例 和IBM Db2 Data Management Console。

可以使用 Amazon Database Migration Service（Amazon DMS）对数据库运行查询、运行 SQL 执行计划和监控数据库。有关更多信息，请参阅《Amazon Database Migration Service 用户指南》中的什么是 Amazon Database Migration Service？

成功将客户端计算机连接到 RDS for Db2 数据库实例后，即可使用任何原生 Db2 工具来复制数据。有关更多信息，请参阅 使用原生 Db2 工具将数据从 Db2 迁移到 Amazon RDS for Db2。

使用 db2look 将数据库元数据从 Db2 复制到 Amazon RDS for Db2

db2look 是一款原生 Db2 工具，用于提取数据定义语言（DDL）文件、对象、授权、配置、WLM 和数据库布局。您可以使用 db2look 将数据库元数据从自行管理的 Db2 数据库复制到 Amazon RDS for Db2 数据库。有关更多信息，请参阅 IBM Db2 文档中的 Mimicking databases using db2look。

复制数据库元数据

1. 在自行管理的 Db2 系统上运行 db2look 工具以提取 DDL 文件。在以下示例中，将 *database_name* 替换为 Db2 数据库的名称。

   ```
   db2look -d database_name -e -l -a -f -wlm -cor -createdb -printdbcfg -o db2look.sql
   ```

2. 如果您的客户端计算机可以访问源（自行管理的 Db2）数据库和 RDS for Db2 数据库实例，则可以通过直接连接到远程实例在客户端计算机上创建 db2look.sql 文件。然后对远程自行管理的 Db2 实例进行编目。

   a. 对节点进行编目。在以下示例中，将 *dns_ip_address* 和 *port* 替换为自行管理的 Db2 数据库的 DNS 名称或 IP 地址和端口号。

   ```
   db2 catalog tcpip node srcnode REMOTE dns_ip_address server port
   ```

   b. 对数据库编目。在以下示例中，将 *source_database_name* 和 *source_database_alias* 替换为自行管理的 Db2 数据库的名称和要用于此数据库的别名。

   ```
   db2 catalog database source_database_name as source_database_alias at node
    srcnode \
       authentication server_encrypt
   ```

c. 连接到源数据库。在以下示例中，将 *source_database_alias*、*user_id* 和 *user_password* 替换为您在上一步中创建的别名以及自行管理的 Db2 数据库的用户 ID 和密码。

```
db2look -d source_database_alias -i user_id -w user_password -e -l -a -f -wlm \
    -cor -createdb -printdbcfg -o db2look.sql
```

3. 如果您无法从客户端计算机访问远程自行管理的 Db2 数据库，请将 db2look.sql 文件复制到客户端计算机上。然后对 RDS for Db2 数据库实例进行编目。

a. 对节点进行编目。在以下示例中，将 *dns_ip_address* 和 *port* 替换为 RDS for Db2 数据库实例的 DNS 名称或 IP 地址和端口号。

```
db2 catalog tcpip node remnode REMOTE dns_ip_address server port
```

b. 对数据库编目。在以下示例中，将 *rds_database_name* 和 *rds_database_alias* 替换为 RDS for Db2 数据库的名称和要用于此数据库的别名。

```
db2 catalog database rds_database_name as rds_database_alias at node remnode \
    authentication server_encrypt
```

c. 对管理 RDS for Db2 的管理员数据库进行编目。您不能使用此数据库存储任何数据。

```
db2 catalog database rdsadmin as rdsadmin at node remnode authentication
 server_encrypt
```

4. 创建缓冲池和表空间。管理员没有创建缓冲池或表空间的权限。但是，您可以使用 Amazon RDS 存储过程来创建它们。

a. 在 db2look.sql 文件中查找缓冲池和表空间的名称和定义。

b. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到 Amazon RDS。在以下示例中，用您自己的信息替换 *master_username* 和 *master_password*。

```
db2 connect to rdsadmin user master_username using master_password
```

c. 通过调用 rdsadmin.create_bufferpool 创建缓冲池。有关更多信息，请参阅 rdsadmin.create_bufferpool。

```
db2 "call rdsadmin.create_bufferpool(
    'database_name',
```

```
    'buffer_pool_name',
    buffer_pool_size,
    'immediate',
    'automatic',
    page_size,
    number_block_pages,
    block_size)"
```

d. 通过调用 rdsadmin.create_tablespace 创建表空间。有关更多信息，请参阅 rdsadmin.create_tablespace。

```
db2 "call rdsadmin.create_tablespace(
    'database_name',
    'tablespace_name',
    'buffer_pool_name',
    tablespace_initial_size,
    tablespace_increase_size,
    'tablespace_type')"
```

e. 对要添加的每个其它缓冲池或表空间重复步骤 c 或 d。

f. 终止连接。

```
db2 terminate
```

5. 创建表和对象。

a. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到 RDS for Db2 数据库。在以下示例中，用您自己的信息替换 *rds_database_name*、*master_username* 和 *master_password*。

```
db2 connect to rds_database_name user master_username using master_password
```

b. 运行 db2look.sql 文件。

```
db2 -tvf db2look.sql
```

c. 终止连接。

```
db2 terminate
```

使用 IMPORT 命令将数据从客户端计算机导入到 Amazon RDS for Db2

您可以从客户端计算机使用 `IMPORT` 命令将您的数据导入到 Amazon RDS for Db2 服务器中。

> **Important**
>
> `IMPORT` 命令方法对于迁移小型表和包含大型对象（LOB）的表很有用。由于 `INSERT` 和 `DELETE` 日志记录操作，`IMPORT` 命令比 `LOAD` 实用程序慢。如果客户端计算机与 RDS for Db2 之间的网络带宽有限，我们建议您使用不同的迁移方法。有关更多信息，请参阅 使用原生 Db2 工具将数据从 Db2 迁移到 Amazon RDS for Db2。

将数据导入 RDS for Db2 服务器

1. 使用 IBM Db2 Data Management Console 登录您的客户端计算机。有关更多信息，请参阅 使用 IBM Db2 Data Management Console 连接到 Amazon RDS for Db2 数据库实例。
2. 在客户端计算机上对 RDS for Db2 数据库进行编目。

   a. 对节点进行编目。在以下示例中，将 *dns_ip_address* 和 *port* 替换为自行管理的 Db2 数据库的 DNS 名称或 IP 地址和端口号。

   ```
   db2 catalog tcpip node srcnode REMOTE dns_ip_address server port
   ```

   b. 对数据库编目。在以下示例中，将 *source_database_name* 和 *source_database_alias* 替换为自行管理的 Db2 数据库的名称和要用于此数据库的别名。

   ```
   db2 catalog database source_database_name as source_database_alias at node
    srcnode \
      authentication server_encrypt
   ```

3. 连接到源数据库。在以下示例中，将 *source_database_alias*、*user_id* 和 *user_password* 替换为您在上一步中创建的别名以及自行管理的 Db2 数据库的用户 ID 和密码。

   ```
   db2look -d source_database_alias -i user_id -w user_password -e -l -a -f -wlm \
       -cor -createdb -printdbcfg -o db2look.sql
   ```

4. 在自行管理的 Db2 系统上使用 `EXPORT` 命令生成数据文件。在以下示例中，将 *directory* 替换为客户端计算机上存在数据文件的目录。将 *file_name* 和 *table_name* 替换为数据文件的名称和表的名称。

```
db2 "export to /directory/file_name.txt of del lobs to /directory/lobs/ \
    modified by coldel\| select * from table_name"
```

5. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到 RDS for Db2 数据库。在以下示例中，用您自己的信息替换 *rds_database_alias*、*master_username* 和 *master_password*。

```
db2 connect to rds_database_alias user master_username using master_password
```

6. 使用 `IMPORT` 命令将数据从客户端计算机上的文件导入到远程 RDS for Db2 数据库。有关更多信息，请参阅 IBM Db2 文档中的 IMPORT 命令。在以下示例中，将 *directory* 和 *file_name* 替换为客户端计算机上存在数据文件的目录和数据文件的名称。将 *SCHEMA_NAME* 和 *TABLE_NAME* 替换为数据库架构名称和表名称。

```
db2 "IMPORT from /directory/file_name.tbl OF DEL LOBS FROM /directory/lobs/ \
    modified by coldel\| replace into SCHEMA_NAME.TABLE_NAME"
```

7. 终止连接。

```
db2 terminate
```

使用 LOAD 命令将数据从客户端计算机导入到 Amazon RDS for Db2

可以使用 `LOAD CLIENT` 命令将数据从客户端计算机上的文件加载到 RDS for Db2 服务器。由于不存在与 RDS for Db2 服务器的 SSH 连接，因此，您可以在自行管理的 Db2 服务器或 Db2 客户端计算机上使用 `LOAD CLIENT` 命令。

Important

`LOAD CLIENT` 命令方法对于迁移小型表很有用。如果客户端与 RDS for Db2 之间的网络带宽有限，我们建议您使用不同的迁移方法。有关更多信息，请参阅 使用原生 Db2 工具将数据从 Db2 迁移到 Amazon RDS for Db2。

如果您的数据文件包含对大型对象文件名的引用，`LOAD` 命令将不起作用，因为大型对象（LOB）需要驻留在 Db2 服务器上。如果您尝试将 LOB 从客户端计算机加载到 RDS for Db2 服务器，则会收到 SQL3025N 错误。改用 IMPORT 命令。

将数据加载到 RDS for Db2 服务器

1. 使用 IBM Db2 Data Management Console 登录您的客户端计算机。有关更多信息，请参阅 使用 IBM Db2 Data Management Console 连接到 Amazon RDS for Db2 数据库实例。
2. 在客户端计算机上对 RDS for Db2 数据库进行编目。

   a. 对节点进行编目。在以下示例中，将 *dns_ip_address* 和 *port* 替换为自行管理的 Db2 数据库的 DNS 名称或 IP 地址和端口号。

   ```
   db2 catalog tcpip node srcnode REMOTE dns_ip_address server port
   ```

   b. 对数据库编目。在以下示例中，将 *source_database_name* 和 *source_database_alias* 替换为自行管理的 Db2 数据库的名称和要用于此数据库的别名。

   ```
   db2 catalog database source_database_name as source_database_alias at node
    srcnode \
       authentication server_encrypt
   ```

3. 连接到源数据库。在以下示例中，将 *source_database_alias*、*user_id* 和 *user_password* 替换为您在上一步中创建的别名以及自行管理的 Db2 数据库的用户 ID 和密码。

   ```
   db2look -d source_database_alias -i user_id -w user_password -e -l -a -f -wlm \
       -cor -createdb -printdbcfg -o db2look.sql
   ```

4. 在自行管理的 Db2 系统上使用 EXPORT 命令生成数据文件。在以下示例中，将 *directory* 替换为客户端计算机上存在数据文件的目录。将 *file_name* 和 *TABLE_NAME* 替换为数据文件的名称和表的名称。

   ```
   db2 "export to /directory/file_name.txt of del modified by coldel\| \
       select * from TPCH.TABLE_NAME"
   ```

5. 使用 RDS for Db2 数据库实例的主用户名和主密码连接到 RDS for Db2 数据库。在以下示例中，用您自己的信息替换 *rds_database_alias*、*master_username* 和 *master_password*。

   ```
   db2 connect to rds_database_alias user master_username using master_password
   ```

6. 使用 LOAD 命令将数据从客户端计算机上的文件加载到远程 RDS for Db2 数据库。有关更多信息，请参阅 IBM Db2 文档中的 LOAD 命令。在以下示例中，将 *directory* 替换为客户端计算机上存在数据文件的目录。将 *file_name* 和 *TABLE_NAME* 替换为数据文件的名称和表的名称。

```
db2 "LOAD CLIENT from /directory/file_name.txt \
    modified by coldel\| replace into TPCH.TABLE_NAME \
    nonrecoverable without prompting"
```

7. 终止连接。

```
db2 terminate
```

使用 INSERT 命令将数据从 Db2 导入到 Amazon RDS for Db2

您可以从自行管理的 Db2 服务器使用 INSERT 命令将您的数据插入到 Amazon RDS for Db2 数据库中。使用这种迁移方法，您可以为远程 RDS for Db2 数据库实例使用昵称。自行管理的 Db2 数据库（源）必须能够连接到 RDS for Db2 数据库（目标）。

> **Important**
>
> INSERT 命令方法对于迁移小型表很有用。如果自行管理的 Db2 数据库和 RDS for Db2 数据库之间的网络带宽有限，我们建议您使用不同的迁移方法。有关更多信息，请参阅 使用原生 Db2 工具将数据从 Db2 迁移到 Amazon RDS for Db2。

将数据从自行管理的 Db2 数据库复制到 RDS for Db2 数据库

1. 在自行管理的 Db2 实例上对 RDS for Db2 数据库实例进行编目。

    a. 对节点进行编目。在以下示例中，将 *dns_ip_address* 和 *port* 替换为自行管理的 Db2 数据库的 DNS 名称或 IP 地址和端口号。

    ```
    db2 catalog tcpip node remnode REMOTE dns_ip_address SERVER port
    ```

    b. 对数据库编目。在以下示例中，将 *rds_database_name* 替换为 RDS for Db2 数据库实例上的数据库名称。

    ```
    db2 catalog database rds_database_name as remdb at node remnode \
        authentication server_encrypt
    ```

2. 在自行管理的 Db2 实例上启用联合身份验证。在以下示例中，将 *source_database_name* 替换为自行管理的 Db2 实例上的数据库名称。

```
db2 update dbm cfg using FEDERATED YES source_database_name
```

3. 在 RDS for Db2 数据库实例创建表。

   a. 对节点进行编目。在以下示例中，将 *dns_ip_address* 和 *port* 替换为自行管理的 Db2 数据库的 DNS 名称或 IP 地址和端口号。

   ```
   db2 catalog tcpip node srcnode REMOTE dns_ip_address server port
   ```

   b. 对数据库编目。在以下示例中，将 *source_database_name* 和 *source_database_alias* 替换为自行管理的 Db2 数据库的名称和要用于此数据库的别名。

   ```
   db2 catalog database source_database_name as source_database_alias at node
    srcnode \
       authentication server_encrypt
   ```

4. 连接到源数据库。在以下示例中，将 *source_database_alias*、*user_id* 和 *user_password* 替换为您在上一步中创建的别名以及自行管理的 Db2 数据库的用户 ID 和密码。

```
db2look -d source_database_alias -i user_id -w user_password -e -l -a -f -wlm \
    -cor -createdb -printdbcfg -o db2look.sql
```

5. 设置联合身份验证，并在自行管理的 Db2 实例上为 RDS for Db2 数据库表创建昵称。

   a. 连接到本地数据库 在以下示例中，将 *source_database_name* 替换为自行管理的 Db2 实例上的数据库名称。

   ```
   db2 connect to source_database_name
   ```

   b. 创建用于访问 Db2 数据来源的包装器。

   ```
   db2 create wrapper drda
   ```

   c. 在联合数据库上定义数据来源。在以下示例中，将 *admin* 和 *admin_password* 替换为自行管理的 Db2 实例的凭证。将 *rds_database_name* 替换为 RDS for Db2 数据库实例上的数据库名称。

```
db2 "create server rdsdb2 type DB2/LUW version '11.5.9.0' \
    wrapper drda authorization "admin" password "admin_password" \
    options( dbname 'rds_database_name', node 'remnode')"
```

d. 映射两个数据库上的用户。在以下示例中，将 *master_username* 和 *master_password* 替换为 RDS for Db2 数据库实例的凭证。

```
db2 "create user mapping for user server rdsdb2 \
    options (REMOTE_AUTHID 'master_username', REMOTE_PASSWORD
 'master_password')"
```

e. 验证与 RDS for Db2 服务器的连接。

```
db2 set passthru rdsdb2
```

f. 在远程 RDS for Db2 数据库中为该表创建昵称。在以下示例中，将 *NICKNAME* 和 *TABLE_NAME* 替换为表的昵称和表的名称。

```
db2 create nickname REMOTE.NICKNAME for RDSDB2.TABLE_NAME.NICKNAME
```

6. 将数据插入远程 RDS for Db2 数据库的表中。在自行管理的 Db2 实例的本地表上的 `select` 语句中使用该昵称。在以下示例中，将 *NICKNAME* 和 *TABLE_NAME* 替换为表的昵称和表的名称。

```
db2 "INSERT into REMOTE.NICKNAME select * from RDS2DB2.TABLE_NAME.NICKNAME"
```

使用 INGEST 实用程序将数据从 Db2 导入到 Amazon RDS for Db2

您可以使用 `INGEST` 实用程序将数据从客户端计算机上的文件和管道持续流式传输到目标 Amazon RDS for Db2 数据库实例。`INGEST` 实用程序支持 `INSERT` 和 `MERGE` 操作。有关更多信息，请参阅 IBM Db2 文档中的 [Ingest 实用程序](https://www.ibm.com/docs/en/db2/11.5?topic=reference-ingest-utility)。

由于 `INGEST` 实用程序支持昵称，因此，您可以使用该实用程序将数据从您自行管理的 Db2 数据库传输到 RDS for Db2 数据库。只要两个数据库之间存在网络连接，这种方法就会起作用。

> **Important**
>
> `INGEST` 实用程序不支持大型对象（LOB）。改用 [`IMPORT` 命令](https://www.ibm.com/docs/en/db2/11.5?topic=commands-import)。

要使用 `INGEST` 实用程序的 `RESTARTABLE` 功能，请在 RDS for Db2 数据库上运行以下命令。

```
db2 "call sysproc.sysinstallobjects(‘INGEST’,‘C’,NULL,NULL)"
```

Amazon RDS for Db2 联合身份验证

可以将 Amazon RDS for Db2 数据库用作联合数据库。在为 RDS for Db2 设置联合身份验证后，您将能够从 RDS for Db2 数据库访问和查询多个数据库中的数据。联合身份验证使您无需将数据迁移到 RDS for Db2 数据库或将数据整合到单个数据库中。

通过将 RDS for Db2 数据库用作联合数据库，您可以继续访问 RDS for Db2 的所有功能，并可以利用各项 Amazon Web Services 服务，同时将数据保存在不同的数据库中。可以设置用于连接相同类型的不同数据库的同构联合身份验证，也可以设置用于连接不同类型的不同数据库的异构联合身份验证。

首先将 RDS for Db2 中的 Db2 数据库连接到远程数据库。然后，可以针对所有连接的数据库运行查询。例如，可以运行 SQL JOIN 语句，来将 RDS for Db2 数据库中的表与 z/OS 上的 Db2 远程数据库中的表相联接。

主题

同构联合身份验证

可以在 RDS for Db2 数据库和以下 Db2 系列产品之间设置同构联合身份验证：

- Db2 for Linux、UNIX、Windows（LUW）
- Db2 iSeries
- Db2 for z/OS

RDS for Db2 同构联合身份验证不支持以下操作：

- 运行 `CATALOG` 命令来在 RDS for Db2 主机数据库上设置节点目录和远程数据库
- 在与 z/OS 上的 Db2 进行联合身份验证时设置工作负载平衡（WLB）
- 配置 IBM 数据服务器驱动程序配置文件 (`db2dsdriver.cfg`)

RDS for Db2 同构联合身份验证具有以下要求：

- 您必须在 `UNFENCED` 模式下创建 DRDA 包装器。否则，联合身份验证在 RDS for Db2 中无法发挥作用。

- 必须支持从 RDS for Db2 主机数据库到远程主机数据库的传入和传出流量。有关更多信息，请参阅通过创建安全组提供对 VPC 中的数据库实例的访问。

主题

步骤 1：创建 DRDA 包装器和联合服务器

对于同构联合身份验证，请创建 DRDA 包装器和联合服务器。与远程主机的连接使用 `HOST`、`PORT` 和 `DBNAME`。

根据远程 Db2 数据库的类型，选择以下方法之一：

- Db2 for Linux、UNIX 和 Windows（LUX）数据库 – 运行以下 SQL 命令。在以下示例中，将 *server_name* 替换为要用于联合身份验证的服务器的名称。将 *db2_version* 替换为远程 Db2 数据库的版本。将 *username* 和 *password* 替换为要连接到的远程 Db2 数据库的凭证。将 *db_name*、*dns_name* 和 *port* 替换为要连接到的远程 Db2 数据库的相应值。

```
create wrapper drda options(DB2_FENCED 'N');
create server server_name type DB2/LUW wrapper drda version 'db2_version'
 authorization "master_username" password "master_password" options (add DBNAME
 'db_name',add HOST 'dns_name',add PORT 'port');
```

  示例

```
create wrapper drda options(DB2_FENCED 'N');
create server SERVER1 type DB2/LUW wrapper drda version '11.5' authorization
 "sysuser" password "******" options (add DBNAME 'TESTDB2',add HOST
 'ip-123-45-67-899.us-west-1.compute.internal',add PORT '25010');
```

- Db2 iSeries – 运行以下 SQL 命令。在以下示例中，将 *wrapper_name* 和 *library_name* 替换为 DRDA 包装器和 wrapper library file 的名称。将 *server_name* 替换为要用于联合身份验证的服务器的名称。将 *db2_version* 替换为远程 Db2 数据库的版本。将 *username* 和 *password* 替换为要连接到的远程 Db2 数据库的凭证。将 *dns_name*、*port* 和 *db_name* 替换为要连接到的远程 Db2 数据库的相应值。

```
create wrapper wrapper_name library 'library name' options(DB2_FENCED 'N');
create server server_name type db2/mvs version db2_version wrapper wrapper_name
 authorization "sername" password "password" options (HOST 'dns_name', PORT 'port',
 DBNAME 'db_name');
```

示例

```
create wrapper WRAPPER1 library 'libdb2drda.so' options(DB2_FENCED 'N');
create server SERVER1 type db2/mvs version 11 wrapper WRAPPER1 authorization
 "sysuser" password "******" options (HOST 'test1.123.com', PORT '446', DBNAME
 'STLEC1');
```

- Db2 for z/OS – 运行以下 SQL 命令。在以下示例中，将 *wrapper_name* 和 *library_name* 替换为 DRDA 包装器和 wrapper library file 的名称。将 *server_name* 替换为要用于联合身份验证的服务器的名称。将 *db2_version* 替换为远程 Db2 数据库的版本。将 *username* 和 *password* 替换为要连接到的远程 Db2 数据库的凭证。将 *dns_name*、*port* 和 *db_name* 替换为要连接到的远程 Db2 数据库的相应值。

```
create wrapper wrapper_name library 'library_name' options(DB2_FENCED 'N');
create server server_name type db2/mvs version db2_version wrapper wrapper_name
 authorization "username" password "password" options (HOST 'dns_name', PORT 'port',
 DBNAME 'db_name');
```

示例

```
create wrapper WRAPPER1 library 'libdb2drda.so' OPTIONS(DB2_FENCED 'N');
create server SERVER1 type db2/mvs version 11 wrapper WRAPPER1 authorization
 "sysuser" password "******" options (HOST 'test1.123.com', PORT '446', DBNAME
 'STLEC1');
```

步骤 2：创建用户映射

通过运行以下 SQL 命令创建用户映射，来将联合服务器与数据来源服务器相关联。在以下示例中，将 *server_name* 替换为要对其执行操作的远程服务器的名称。这是您在步骤 1 中创建的服务器。将 *username* 和 *password* 替换为该远程服务器的凭证。

```
create user mapping for user server server_name options (REMOTE_AUTHID 'username',
 REMOTE_PASSWORD 'password');
```

有关更多信息，请参阅 IBM Db2 文档中的 User mappings。

步骤 3：检查连接

通过检查连接，确认成功设置了联合身份验证。打开一个会话，通过使用 SET PASSTHRU 命令向远程数据来源发送原生 SQL 命令，然后在远程数据服务器上创建表。

1. 打开和关闭会话来向数据来源提交 SQL。在以下示例中，将 *server_name* 替换为在步骤 1 中为联合身份验证创建的服务器的名称。

```
set passthru server_name;
```

2. 创建一个新表。在以下示例中，将 *column_name*、*data_type* 和 *value* 替换为对应于表的项目。

```
create table table_name
 ( column_name data_type(value), column_name data_type(value);
```

   有关更多信息，请参阅 IBM Db2 文档中的 CREATE TABLE statement。

3. 创建索引，向表中插入行的值，然后重置连接。重置连接会断开连接，但会保留后端进程。在以下示例中，将 *index_name*、*table_name*、*column_name* 和 *columnx_value* 替换为您的信息。

```
create index index_name on table_name(column_name);
insert into table_name values(column1_value,column2_value,column3_value);
insert into table_name values(column1_value,column2_value,column3_value);
set passthru reset;

connect reset;
```

4. 连接到远程 Db2 数据库，为远程服务器创建昵称，然后执行操作。访问完远程 Db2 数据库中的数据后，重置并终止连接。在以下示例中，将 *database_name* 替换为远程 Db2 数据库的名称。将 *nickname* 替换为名称。将 *server_name* 和 *table_name* 替换为远程服务器的名称和该服务器上您要对其执行操作的表。将 *username* 替换为远程服务器的信息。将 *sql_command* 替换为要在远程服务器上执行的操作。

```
connect to database_name;
create nickname nickname for server_name."username"."table_name";
select sql_command from nickname;
connect reset;
```

```
terminate;
```

示例

以下示例创建了一个直通会话，来支持在联合服务器 testdb10 上进行操作。

接下来，它创建包含三列且各列的数据类型不同的表 t1。

然后，该示例对表 t1 中的三列创建索引 i1_t1。之后，它插入两行，其中包含这三列的值，然后断开连接。

最后，该示例连接到远程 Db2 数据库 testdb2，并在联合服务器 testdb10 中为表 t1 创建昵称。它使用用户名 TESTUSER 为该数据来源创建昵称。一条 SQL 命令输出表 t1 中的所有数据。该示例断开连接并结束会话。

```
set passthru testdbl0;

create table t1 ( c1 decimal(13,0), c2 char(200), c3 int);

create index i1_t1 on t1(c3);
insert into t1 values(1,'Test',1);
insert into t1 values(2,'Test 2',2);
connect reset;

connect to testdb2;
create nickname remote_t1 for testdbl0."TESTUSER"."T1";
select * from remote_t1;
connect reset;
terminate;
```

异构联合身份验证

可以在 RDS for Db2 数据库与其他数据来源（例如 Oracle 和 Microsoft SQL Server）之间设置异构联合身份验证。有关 Db2 LUW 支持的数据来源的完整列表，请参阅 IBM 支持网站上的 [Data Source Support Matrix of Federation Bundled in Db2 LUW V11.5]。

RDS for Db2 异构联合身份验证不支持以下各项：

- 其他数据来源的原生包装器
- 其他数据来源的 JDBC 包装器

- 与 Sybase、Informix 和 Teradata 数据来源的联合身份验证，因为这些数据来源要求在 RDS for Db2 上安装客户端软件

RDS for Db2 异构联合身份验证具有以下要求：

- RDS for Db2 仅支持 ODBC 包装器方法。
- 如果您创建了包装器的显式定义，则必须将选项 `DB2_FENCED` 设置为 `'N'`。有关 ODBC 的有效包装器选项列表，请参阅 IBM Db2 文档中的 ODBC options。
- 必须支持从 RDS for Db2 主机数据库到远程主机数据库的传入和传出流量。有关更多信息，请参阅通过创建安全组提供对 VPC 中的数据库实例的访问。

有关与 Oracle 的联合身份验证的信息，请参阅 IBM 支持网站上的 How to query Oracle by using Db2 Federation and the ODBC driver?

有关支持联合身份验证的数据来源的更多信息，请参阅 IBM 支持网站上的 Data Source Support Matrix of Federation Bundled in Db2 LUW V11.5。

主题

步骤 1：创建 ODBC 包装器

通过运行以下命令创建包装器：

```
db2 "create wrapper odbc options( module '/home/rdsdb/sqllib/federation/odbc/lib/
libodbc.so')"
```

步骤 2：创建联合服务器

通过运行以下命令创建联合服务器。在以下示例中，将 *server_name* 替换为要用于联合身份验证的服务器的名称。将 *wrapper_type* 替换为相应的包装器。将 *db_version* 替换为远程数据库的版本。将 *dns_name*、*port* 和 *service_name* 替换为要连接到的远程数据库的相应值。

```
db2 "create server server_name type wrapper_type version db_version options (HOST
 'dns_name', PORT 'port', SERVICE_NAME 'service_name')"
```

有关包装器类型的信息，请参阅 IBM 支持网站上的 Data Source Support Matrix of Federation Bundled in Db2 LUW V11.5。

示例

以下示例为远程 Oracle 数据库创建联合服务器。

```
db2 "create server server1 type oracle_odbc version 12.1 options (HOST
 'test1.amazon.com', PORT '1521', SERVICE_NAME 'pdborcl.amazon.com')"
```

步骤 3：创建用户映射

通过运行以下 SQL 命令创建用户映射，来将联合服务器与数据来源服务器相关联。在以下示例中，将 *server_name* 替换为要对其执行操作的远程服务器的名称。这是您在步骤 2 中创建的服务器。将 *username* 和 *password* 替换为该远程服务器的凭证。

```
create user mapping for user server server_name options (REMOTE_AUTHID 'username',
 REMOTE_PASSWORD 'password');
```

有关更多信息，请参阅 IBM Db2 文档中的 User mappings。

步骤 4：检查连接

通过检查连接，确认成功设置了联合身份验证。打开一个会话，通过使用 SET PASSTHRU 命令向远程数据来源发送原生 SQL 命令，然后在远程数据服务器上创建表。

1. 打开和关闭会话来向数据来源提交 SQL。在以下示例中，将 *server_name* 替换为在步骤 2 中为联合身份验证创建的服务器的名称。

   ```
   set passthru server_name;
   ```

2. 创建一个新表。在以下示例中，将 *column_name*、*data_type* 和 *value* 替换为对应于表的项目。

   ```
   create table table_name
    ( column_name data_type(value), column_name data_type(value);
   ```

有关更多信息，请参阅 IBM Db2 文档中的 CREATE TABLE statement。

3. 创建索引，向表中插入行的值，然后重置连接。重置连接会断开连接，但会保留后端进程。在以下示例中，将 *index_name*、*table_name*、*column_name* 和 *columnx_value* 替换为您的信息。

```
create index index_name on table_name(column_name);
insert into table_name values(column1_value,column2_value,column3_value);
insert into table_name values(column1_value,column2_value,column3_value);
set passthru reset;

connect reset;
```

4. 连接到远程 Db2 数据库，为远程服务器创建昵称，然后执行操作。访问完远程 Db2 数据库中的数据后，重置并终止连接。在以下示例中，将 *database_name* 替换为远程 Db2 数据库的名称。将 *nickname* 替换为名称。将 *server_name* 和 *table_name* 替换为远程服务器的名称和该服务器上您要对其执行操作的表。将 *username* 替换为远程服务器的信息。将 *sql_command* 替换为要在远程服务器上执行的操作。

```
connect to database_name;
create nickname nickname for server_name."username"."table_name";
select sql_command from nickname;
connect reset;
terminate;
```

示例

以下示例创建了一个直通会话，来支持在联合服务器 `testdb10` 上进行操作。

接下来，它创建包含三列且各列的数据类型不同的表 `t1`。

然后，该示例对表 `t1` 中的三列创建索引 `i1_t1`。之后，它插入两行，其中包含这三列的值，然后断开连接。

最后，该示例连接到远程 Db2 数据库 `testdb2`，并在联合服务器 `testdb10` 中为表 `t1` 创建昵称。它使用用户名 `TESTUSER` 为该数据来源创建昵称。一条 SQL 命令输出表 `t1` 中的所有数据。该示例断开连接并结束会话。

```
set passthru testdbl0;
```

```
create table t1 ( c1 decimal(13,0), c2 char(200), c3 int);

create index i1_t1 on t1(c3);
insert into t1 values(1,'Test',1);
insert into t1 values(2,'Test 2',2);
connect reset;

connect to testdb2;
create nickname remote_t1 for testdbl0."TESTUSER"."T1";
select * from remote_t1;
connect reset;
terminate;
```

使用适用于 Amazon RDS for Db2 的副本

要在 RDS for Db2 数据库实例之间配置复制，您可以创建副本数据库。有关 RDS for Db2 副本和其它数据库引擎之间的差异摘要，请参阅数据库引擎的只读副本之间的差异。

Db2 副本数据库是主数据库的物理副本。处于备用模式的 Db2 副本称为备用副本。Db2 不支持在副本中进行读或写操作，但您可以提升副本以使其变为可写。提升的副本拥有提出提升请求时的已复制数据。RDS for Db2 使用 Db2 高可用性灾难恢复（HADR）来进行复制。有关更多信息，请参阅 IBM Db2 文档中的 High availability disaster recovery (HADR)。

备用副本

创建或修改 Db2 副本时，可以将其置于备用模式。对于备用副本，Db2 环境变量 `DB2_HADR_ROS` 设置为 `OFF`，以便副本数据库不接受用户连接。备用副本的主要用途是跨区域灾难恢复。

备用副本无法为只读工作负载提供服务。备用副本没有任何归档日志。

您可以从一个源数据库实例创建最多三个副本。

数据库激活

Db2 HADR 是在数据库级别配置的。创建副本后，将为所有 Db2 数据库（包括 RDS 完全管理的 `rdsadmin`）设置 HADR。在创建 Db2 副本之前，必须显式激活所有数据库。否则，副本创建将失败，并且 Amazon RDS 发出一个事件。在数据库实例拥有一个或多个副本后，您无法使用 `rdsadmin.activate_database` 或 `rdsadmin.deactivate_database` 存储过程来激活或停用数据库实例上的任何数据库。但是，您可以调用 `rdsadmin.reactivate_database` 存储过程来重新激活数据库。有关更多信息，请参阅用于管理 RDS for Db2 数据库的存储过程。

HADR 配置

可以通过连接到数据库，然后运行 `db2 get db cfg` 来查看该数据库的所有 HADR 配置。

归档日志保留

满足以下条件后，Amazon RDS 会清除主数据库实例中的日志：

- 这些日志至少保留了两个小时。
- 归档日志保留时间的设置已过期。
- 归档日志已成功复制到所有副本数据库实例。此条件既适用于同一 Amazon Web Services 区域中的数据库实例，也适用于跨区域数据库实例。

有关如何设置存档日志保留时间的信息，请参阅rdsadmin.set_archive_log_retention。

Amazon RDS 会单独检查和清理每个数据库。如果数据库丢失 HADR 连接或者无法获得有关该连接的信息，Amazon RDS 将跳过该数据库并且不会清除归档日志。

Db2 复制过程中的中断

当您创建副本时，Amazon RDS 会制作源数据库实例的数据库快照，然后开始复制。数据库快照操作开始时，源数据库实例会经历非常短暂的 I/O 暂停。这种 I/O 暂停通常持续大约一秒时间。但是，如果源数据库实例为多可用区部署，则源数据库实例不会经历任何 I/O 暂停。这是因为在多可用区部署中，快照取自辅助数据库实例。

数据库快照变成 Db2 副本。Amazon RDS 为源数据库和副本设置必要的参数和权限，而不会经历任何服务中断。同样，如果删除副本，也不会发生中断。

RDS for Db2 副本的要求和注意事项

在创建 Db2 副本之前，请先熟悉以下要求和注意事项。

RDS for Db2 副本的版本和许可要求

在创建 RDS for Db2 副本之前，请查看以下信息：

- Db2 高级版（AE）和标准版（SE）都可以在备用模式下为自带许可（BYOL）模型和通过 Amazon Web Services Marketplace 的 Db2 许可证模型创建副本。
- 必须为只读副本指定自定义参数组。对于使用 BYOL 模型的副本，此自定义参数组必须包括您的 IBM Site ID 和 IBM Customer ID。您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 为只读副本指定此自定义参数组。有关更多信息，请参阅 Db2 自带许可的 IBM ID。
- 对于备用副本，无论数据库实例大小如何，vCPU 计数均为 2。以下各项描述了 RDS for Db2 许可证类型之间的区别：
  - 对于 BYOL 模型，Amazon License Manager 配置显示 RDS for Db2 数据库实例使用两个 vCPU。
  - 对于通过 Amazon Web Services Marketplace 的 Db2 许可证模型，账单会反映两个 vCPU 的许可证成本。
- 所有 Db2 11.5 版本都支持副本数据库实例。Db2 副本支持次要版本升级。

RDS for Db2 的备份和还原注意事项

在创建 RDS for Db2 副本之前，请考虑以下各项：

- 要创建 RDS for Db2 副本的快照或开启自动备份，请确保手动设置备份保留期。原定设置情况下，自动备份处于开启状态。
- 还原副本备份时，将还原到数据库时间，而不是进行备份的时间。数据库时间指备份中数据的最新应用的事务时间。差异很大，因为副本可能会落后于主数据库数分钟或数小时。当有多个数据库时，RDS for Db2 将使用最早的数据库时间。

  要查找差异，请运行 Amazon CLI describe-db-snapshots 命令或调用 RDS API DescribeDBSnapshots 操作。将 `SnapshotDatabaseTime` 值和 `OriginalSnapshotCreateTime` 值进行比较。`SnapshotDatabaseTime` 值是副本备份的数据库时间。`OriginalSnapshotCreateTime` 值是主数据库上最新应用的事务。

RDS for Db2 副本的复制注意事项

在创建 RDS for Db2 副本之前，请考虑以下各项：

- 复制对 RDS for Db2 数据库实例上的所有数据库使用 Db2 HADR。
- 复制不支持 `LOAD` 命令。如果从源数据库实例运行 `LOAD` 命令，将收到不一致的数据。
- RDS for Db2 不会复制以下各项：
  - 存储访问。注意依赖于存储访问的数据，例如外部表。
  - 非内联 LOB。
  - 外部存储过程的二进制文件（在 C 或 Java 中）。
- 对于备用副本，RDS for Db2 会复制以下各项：
  - 本地用户，主用户除外
  - 数据库配置参数

RDS for Db2 副本的其它注意事项

在创建 RDS for Db2 副本之前，请考虑以下各项：

- RDS for Db2 将数据库配置复制到副本中。当 RDS for Db2 提升副本时，它会停用并激活每个数据库。

- RDS for Db2 将本地用户（但不是主用户）复制到副本。可以修改副本上的主用户。有关更多信息，请参阅修改 Amazon RDS 数据库实例。
- 所有数据库必须处于活动状态。有关激活数据库的信息，请参阅用于管理 RDS for Db2 数据库的存储过程。
- 在创建副本之前，必须完成所有用于创建、删除、还原或前滚数据库的存储过程。有关这些存储过程的信息，请参阅用于管理 RDS for Db2 数据库的存储过程。
- 创建副本时，Amazon RDS 会将源数据库实例上所有数据库的数据库级参数 `blocknonlogged` 设置为 `YES`。当源副本再次变为独立实例时，Amazon RDS 将该值重新设置为 `NO`。有关更多信息，请参阅 IBM Db2 文档中的 blocknonlogged - Block creation of tables that allow non-logged activity configuration parameter。
- 创建副本时，Amazon RDS 会将源数据库实例上所有数据库的数据库级参数 `logindexbuild` 设置为 `YES`。当源副本再次变为独立实例时，Amazon RDS 将该值重新设置为 `NO`。有关更多信息，请参阅 IBM Db2 文档中的 logindexbuild - Log index pages created configuration parameter。

为创建 RDS for Db2 副本做准备

在开始使用副本之前，请执行以下任务：

任务

- 启用自动备份
- 规划计算和存储资源
- 准备数据库

启用自动备份

数据库实例可充当源数据库实例之前，您必须在源数据库实例上启用自动备份。要了解如何执行此过程，请参阅启用自动备份。

规划计算和存储资源

确保源数据库实例及其副本在计算和存储容量方面具有适当的大小，以适合其运行负载。如果副本达到计算、网络或存储资源容量，则副本会停止接收或应用来自其源实例的更改。RDS for Db2 不会通过干预来缓解源数据库实例与其副本之间的高副本滞后。您可以单独修改某个副本的存储和 CPU 资源，而不影响其源及其他副本。

准备数据库

在创建副本之前，请确保数据库实例包含您希望在数据库实例上存在的所有数据库。在创建复制副本之后，您无法在数据库实例上创建、删除数据库或本地还原数据库。对 rdsadmin.create_database、rdsadmin.drop_database 或 rdsadmin.restore_database 存储过程的任何调用都将失败。

此外，请确保数据库实例上的所有数据库都处于活动状态。否则，副本创建将失败。有关激活数据库的信息，请参阅用于管理 RDS for Db2 数据库的存储过程。

在备用模式下创建 RDS for Db2 副本

可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 在备用模式下创建副本。

控制台

从源 RDS for Db2 数据库实例创建备用副本

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。
3. 选择要用作备用副本源的 RDS for Db2 数据库实例。
4. 对于操作，选择创建副本。
5. 对于副本模式，选择备用。
6. 选择要使用的设置。对于数据库实例标识符，输入该只读副本的名称。根据需要，调整其他设置。
7. 对于区域，选择将在其中启动备用副本的 Amazon Web Services 区域。
8. 选择您的实例大小和存储类型。我们建议您为只读副本使用与源数据库实例相同的数据库实例类和存储类型。
9. 对于多可用区部署，选择创建备用实例，以便在另一个可用区中创建副本的备用来支持备用副本失效转移。
10. 选择要使用的其他设置。
11. 选择创建副本。

在数据库页面中，备用副本具有角色副本。

Amazon CLI

要在备用模式下创建 Db2 副本，请在 Amazon CLI 命令 create-db-instance-read-replica 中将 `--replica-mode` 设为 `mounted`。

Example

对于 Linux、macOS 或 Unix：

```
aws rds create-db-instance-read-replica \
    --db-instance-identifier my_read_replica \
    --source-db-instance-identifier my_db_instance \
    --replica-mode mounted
```

对于 Windows：

```
aws rds create-db-instance-read-replica ^
    --db-instance-identifier my_read_replica ^
    --source-db-instance-identifier my_db_instance ^
    --replica-mode mounted
```

RDS API

要在副本模式下创建 Db2 副本，请在 RDS API 操作 CreateDBInstanceReadReplica 中指定 `ReplicaMode=mounted`。

使用 RDS for Db2 副本备份

您可以创建和还原 RDS for Db2 副本的备份。RDS for Db2 同时支持自动备份和手动快照。有关更多信息，请参阅备份、还原和导出数据。以下各节描述了管理主数据库的备份与管理 RDS for Db2 副本的备份之间的主要区别。

主题

- 开启 RDS for Db2 副本备份
- 还原 RDS for Db2 副本备份

开启 RDS for Db2 副本备份

RDS for Db2 副本在默认情况下未开启自动备份。通过将备份保留期设置为非零正值来开启自动备份。

控制台

立即开启自动备份

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择数据库，然后选择要修改的数据库实例。
3. 选择修改。
4. 对于备份保留期，请选择非零正值，例如三天。
5. 选择 Continue (继续)。
6. 选择立即应用。
7. 选择修改数据库实例以保存更改并启用自动备份。

Amazon CLI

要开启自动备份，请使用 Amazon CLI modify-db-instance 命令。

包括以下参数：

- `--db-instance-identifier`
- `--backup-retention-period`
- `--apply-immediately` 或者 `--no-apply-immediately`

以下示例通过将备份保留期设置为三天来开启自动备份。更改将立即应用。

Example

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier my_db_instance  \
    --backup-retention-period 3 \
    --apply-immediately
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier my_db_instance  ^
```

```
    --backup-retention-period 3 ^
    --apply-immediately
```

RDS API

要开启自动备份，请使用 RDS API ModifyDBInstance 操作以及以下必需参数：

- `DBInstanceIdentifier`
- `BackupRetentionPeriod`

还原 RDS for Db2 副本备份

您可以按照还原主数据库备份的方式还原 RDS for Db2 副本备份。有关更多信息，请参阅以下主题：

- 还原到数据库实例
- 将 Amazon RDS 的数据库实例还原到指定时间

还原副本备份时的主要注意事项是确定要还原到的时间点。数据库时间指备份中数据的最新应用的事务时间。还原副本备份时，将还原到数据库时间，而不是备份完成的时间。差异很大，因为副本可能会落后于主数据库数分钟或数小时。因此，副本备份的数据库时间以及还原到的时间点可能比备份创建时间早得多。

要查找数据库时间和创建时间之间的差异，请运行 Amazon CLI describe-db-snapshots 命令或调用 RDS API DescribeDBSnapshots 操作。比较 `SnapshotDatabaseTime` 值和 `OriginalSnapshotCreateTime` 值。`SnapshotDatabaseTime` 值是副本备份的所有数据库中最早的数据库时间。`OriginalSnapshotCreateTime` 值是主数据库上最新应用的事务。请注意，多个数据库的复制滞后可能有所不同，数据库时间可能介于这两个时间之间。

以下 Amazon CLI 示例显示了两个时间之间的差异：

```
aws rds describe-db-snapshots \
    --db-instance-identifier my_db2_replica
    --db-snapshot-identifier my_replica_snapshot

{
    "DBSnapshots": [
        {
            "DBSnapshotIdentifier": "my_replica_snapshot",
            "DBInstanceIdentifier": "my_db2_replica",
```

```
            "SnapshotDatabaseTime": "2022-07-26T17:49:44Z",
            ...
            "OriginalSnapshotCreateTime": "2021-07-26T19:49:44Z"
        }
    ]
}
```

排查 RDS for Db2 复制问题

本内容描述了可能的 RDS for Db2 复制问题和解决方案。

主题

监控 Db2 复制滞后

要监控 Amazon CloudWatch 中的复制滞后，请查看 Amazon RDS `ReplicaLag` 指标。有关复制滞后时间的信息，请参阅监控只读复制和Amazon RDS 的 Amazon CloudWatch 指标。

对于备用副本，如果滞后时间过长，请查询 `MON_GET_HADR` 表以了解源数据库实例的状态。请勿查询副本数据库实例，因为副本数据库实例不接受用户连接。

Db2 复制错误

由于多种原因，Db2 复制可能处于错误状态。监控事件和数据库实例状态，以确保数据库实例正在复制。要确认数据库实例正在复制，请在 Amazon RDS 控制台中查看副本的诊断日志。在日志中，在 HADR 消息中查找错误。将日志序列号与主序列号进行比较。有关更多信息，请参阅 Amazon RDS for Db2 数据库日志文件。

Amazon RDS for Db2 数据库实例选项

下面显示了适用于运行 Db2 数据库引擎的 Amazon RDS 实例的选项或附加功能。要启用这些选项，您可以将它们添加到自定义选项组中，然后将选项组关联到数据库实例。有关使用选项组的详细信息，请参阅使用选项组。

Amazon RDS 支持以下 Db2 选项：

选项	选项 ID
Db2 审核日志记录	`DB2_AUDIT`

Db2 审核日志记录

借助 Db2 审核日志记录，Amazon RDS 会记录数据库活动，包括登录到数据库的用户和对数据库运行的查询。RDS 使用您提供的 Amazon Identity and Access Management（IAM）角色将完成的审核日志上传到 Amazon S3 存储桶。

主题

- 设置 Db2 审核日志记录
- 管理 Db2 审核日志
- 查看审核日志
- 排查 Db2 审核日志记录问题

设置 Db2 审核日志记录

要为 Amazon RDS for Db2 数据库启用审核日志记录，您需要在 RDS for Db2 数据库实例上启用 `DB2_AUDIT` 选项。然后，配置审核策略以为特定数据库启用该功能。要在 RDS for Db2 数据库实例上启用该选项，您需要为 `DB2_AUDIT` 选项配置选项设置。为此，您可以为 Amazon S3 存储桶提供 Amazon 资源名称（ARN），并向 IAM 角色提供访问存储桶的权限。

要为 RDS for Db2 数据库设置 Db2 审核日志记录，请完成以下步骤。

主题

- 步骤 1：创建 Amazon S3 存储桶
- 步骤 2：创建 IAM 策略
- 步骤 3：创建 IAM 角色并附加您的 IAM 策略
- 步骤 4：为 Db2 审核日志记录配置选项组
- 步骤 5：配置审核策略
- 步骤 6：检查审核配置

步骤 1：创建 Amazon S3 存储桶

如果您尚未完成此操作，请创建一个 Amazon S3 存储桶，以便 Amazon RDS 可以在其中上传您的 RDS for Db2 数据库的审核日志文件。以下限制适用于您用作审核文件目标的 S3 存储桶。

- 它必须与您的 RDS for Db2 数据库实例位于相同的 Amazon Web Services 区域。
- 它不得对公众开放。

- 存储桶拥有者也必须是 IAM 角色拥有者。

要了解如何创建 Amazon S3 存储桶，请参阅《Amazon S3 用户指南》中的创建存储桶。

启用审核日志记录后，Amazon RDS 会自动将日志从您的数据库实例发送到以下位置：

- 数据库实例级别日志 – `bucket_name/db2-audit-logs/dbi_resource_id/date_time_utc/`
- 数据库级别日志 - `bucket_name/db2-audit-logs/dbi_resource_id/date_time_utc/db_name/`

请记下存储桶的 Amazon 资源名称（ARN）。这些信息是完成后续步骤所必需的。

步骤 2：创建 IAM 策略

创建一个 IAM 策略，该策略具有将审核日志文件从数据库实例传输到 Amazon S3 存储桶所需的权限。此步骤假定您已创建了 S3 存储桶。

创建策略前，请收集以下信息：

- 存储桶的 ARN。
- 您的 Amazon Key Management Service（Amazon KMS）密钥的 ARN，如果您的存储桶使用 SSE-KMS 加密。

创建包含以下权限的 IAM 策略：

```
"s3:ListBucket",
 "s3:GetBucketAcl",
 "s3:GetBucketLocation",
 "s3:PutObject",
 "s3:ListMultipartUploadParts",
 "s3:AbortMultipartUpload",
 "s3:ListAllMyBuckets"
```

> **Note**
>
> Amazon RDS 需要在内部执行 `s3:ListAllMyBuckets` 操作，以确认同一 Amazon Web Services 账户是否拥有 S3 存储桶和 RDS for Db2 数据库实例。

如果存储桶使用 SSE-KMS 加密，则还应包括 IAM 角色和 Amazon KMS 密钥的以下权限。

在 IAM 角色的策略中包括以下权限。

```
"kms:GenerateDataKey",
 "kms:Decrypt"
```

在 Amazon KMS 密钥的密钥策略中包含以下权限。将 *111122223333* 替换为您的账号，并将 *AR0A123456789EXAMPLE* 替换为 IAM 角色名称。

```
{
  "Sid": "Allow RDS role to use the key",
  "Effect": "Allow",
  "Principal": {
    "AWS": [
      "arn:aws:sts::111122223333:assumed-role/AR0A123456789EXAMPLE/RDS-Db2Audit",
      "arn:aws:iam::111122223333:role/AR0A123456789EXAMPLE"
    ]
  },
  "Action": [
    "kms:GenerateDataKey",
    "kms:Decrypt"
  ],
  "Resource": "*"
}
```

您可以使用 Amazon Web Services Management Console或 Amazon Command Line Interface（Amazon CLI）创建 IAM 策略。

控制台

创建 IAM 策略以允许 Amazon RDS 访问 Amazon S3 存储桶

1. 登录到 Amazon Web Services Management Console，然后通过以下网址打开 IAM 控制台：https://console.aws.amazon.com/iam/。
2. 在导航窗格中，选择策略。
3. 选择创建策略，然后选择 JSON。
4. 在添加操作中，按 S3 进行筛选。添加 ListBucket、GetBucketAcl 和 GetBucketLocation 访问权限。
5. 在添加资源中，选择添加。在资源类型中，选择存储桶，然后输入存储桶的名称。选择添加资源。

6. 选择添加新语句。
7. 在添加操作中，按 S3 进行筛选。添加 PutObject、ListMultipartUploadParts 和 AbortMultipartUpload 访问权限。
8. 在添加资源中，选择添加。在资源类型中，选择对象，然后输入*#######/**。选择添加资源。
9. 选择添加新语句。
10. 在添加操作中，按 S3 进行筛选。添加 ListAllMyBuckets 访问权限。
11. 在添加资源中，选择添加。在资源类型中，选择所有资源。选择添加资源。
12. 如果您使用自己的 KMS 密钥来加密数据：
    1. 选择添加新语句。
    2. 在添加操作中，按 KMS 进行筛选。添加 GenerateDataKey 和 Decrypt 访问权限。
    3. 在添加资源中，选择添加。在资源类型中，选择所有资源。选择添加资源。
13. 选择下一步。
14. 对于策略名称，输入此策略的名称。
15. （可选）对于描述，输入此策略的描述。
16. 选择创建策略。

Amazon CLI

创建 IAM 策略以允许 Amazon RDS 访问 Amazon S3 存储桶

1. 运行 create-policy 命令。在以下示例中，将 *iam_policy_name* 和 *amzn-s3-demo-bucket* 替换为 IAM 策略的名称以及目标 Amazon S3 存储桶的名称。

   对于 Linux、macOS 或 Unix：

```
aws iam create-policy \
    --policy-name iam_policy_name \
    --policy-document '{
        "Version": "2012-10-17",
        "Statement": [
            {
                "Sid": "Statement1",
                "Effect": "Allow",
                "Action": [
                    "s3:ListBucket",
                    "s3:GetBucketAcl",
```

```
                    "s3:GetBucketLocation"
                ],
                "Resource": [
                    "arn:aws:s3:::amzn-s3-demo-bucket"
                ]
            },
            {
                "Sid": "Statement2",
                "Effect": "Allow",
                "Action": [
                    "s3:PutObject",
                    "s3:ListMultipartUploadParts",
                    "s3:AbortMultipartUpload"
                ],
                "Resource": [
                    "arn:aws:s3:::amzn-s3-demo-bucket/*"
                ]
            },
            {
                "Sid": "Statement3",
                "Effect": "Allow",
                "Action": [
                    "s3:ListAllMyBuckets"
                ],
                "Resource": [
                    "*"
                ]
            },
            {
                "Sid": "Statement4",
                "Effect": "Allow",
                "Action": [
                    "kms:GenerateDataKey",
                    "kms:Decrypt"
                ],
                "Resource": [
                    "*"
                ]
            }
        ]
}'
```

对于 Windows：

```
aws iam create-policy ^
    --policy-name iam_policy_name ^
    --policy-document '{
        "Version": "2012-10-17",
        "Statement": [
            {
                "Sid": "Statement1",
                "Effect": "Allow",
                "Action": [
                    "s3:ListBucket",
                    "s3:GetBucketAcl",
                    "s3:GetBucketLocation"
                ],
                "Resource": [
                    "arn:aws:s3:::amzn-s3-demo-bucket"
                ]
            },
            {
                "Sid": "Statement2",
                "Effect": "Allow",
                "Action": [
                    "s3:PutObject",
                    "s3:ListMultipartUploadParts",
                    "s3:AbortMultipartUpload"
                ],
                "Resource": [
                    "arn:aws:s3:::amzn-s3-demo-bucket/*"
                ]
            },
            {
                "Sid": "Statement3",
                "Effect": "Allow",
                "Action": [
                    "s3:ListAllMyBuckets"
                ],
                "Resource": [
                    "*"
                ]
            },
            {
                "Sid": "Statement4",
                "Effect": "Allow",
                "Action": [
```

```
                    "kms:GenerateDataKey",
                    "kms:Decrypt"
                ],
                "Resource": [
                    "*"
                ]
            }
        ]
    }'
```

2. 创建策略后，请记下策略的 ARN。您需要[步骤 3：创建 IAM 角色并附加您的 IAM 策略](#)的 ARN。

有关创建 IAM policy 的更多信息，请参阅《IAM 用户指南》中的[创建 IAM policy](#)。

步骤 3：创建 IAM 角色并附加您的 IAM 策略

此步骤假定您已在[步骤 2：创建 IAM 策略](#)中创建了 IAM 策略。在此步骤中，您将为 RDS for Db2 数据库实例创建一个 IAM 角色，然后将 IAM 策略附加到该角色。

您可以使用控制台或 Amazon CLI 为数据库实例创建 IAM 角色。

控制台

创建 IAM 角色并向其附加 IAM policy

1. 登录 Amazon Web Services Management Console，然后通过以下网址打开 IAM 控制台：https://console.aws.amazon.com/iam/。
2. 在导航窗格中，选择角色。
3. 选择创建角色。
4. 对于可信实体类型，请选择 Amazon Web Services 服务。
5. 对于服务或使用案例，请选择 RDS，然后选择 RDS – 向数据库添加角色。
6. 选择下一步。
7. 对于权限策略，搜索并选择您创建的 IAM policy 的名称。
8. 选择下一步。
9. 对于 Role name（角色名称），输入一个角色名称。
10. （可选）在描述中，输入新角色的描述。
11. 选择创建角色。

Amazon CLI

创建 IAM 角色并向其附加 IAM policy

1. 运行 create-role 命令。在以下示例中，将 *iam_role_name* 替换为您的 IAM 角色的名称。

   对于 Linux、macOS 或 Unix：

```
aws iam create-role \
    --role-name iam_role_name \
    --assume-role-policy-document '{
      "Version": "2012-10-17",
      "Statement": [
        {
          "Effect": "Allow",
          "Principal": {
            "Service": "rds.amazonaws.com"
          },
          "Action": "sts:AssumeRole"
        }
      ]
    }'
```

   对于 Windows：

```
aws iam create-role ^
    --role-name iam_role_name ^
    --assume-role-policy-document '{
      "Version": "2012-10-17",
      "Statement": [
        {
          "Effect": "Allow",
          "Principal": {
            "Service": "rds.amazonaws.com"
          },
          "Action": "sts:AssumeRole"
        }
      ]
    }'
```

2. 创建角色之后，记下该角色的 ARN。在下一步骤步骤 4：为 Db2 审核日志记录配置选项组中，您需要用到此 ARN。

3. 运行 attach-role-policy 命令。在以下示例中，将 *iam_policy_arn* 替换为您在步骤 2：创建 IAM 策略中创建的 IAM policy。将 *iam_role_name* 替换为刚创建的 IAM 角色的名称。

   对于 Linux、macOS 或 Unix：

```
aws iam attach-role-policy \
   --policy-arn iam_policy_arn \
   --role-name iam_role_name
```

   对于 Windows：

```
aws iam attach-role-policy ^
   --policy-arn iam_policy_arn ^
   --role-name iam_role_name
```

有关更多信息，请参阅《IAM 用户指南》中的创建向 IAM 用户委派权限的角色。

步骤 4：为 Db2 审核日志记录配置选项组

将 Db2 审核日志记录选项添加到 RDS for Db2 数据库实例的过程如下：

1. 创建新的选项组，或者复制或修改现有选项组。
2. 添加和配置所有必需的选项。
3. 将选项组与数据库实例相关联。

在添加 Db2 审核日志记录选项后，您无需重新启动数据库实例。只要选项组处于活动状态，您就可以创建审核并在 S3 存储桶中存储审核日志。

在数据库实例的选项组中添加和配置 Db2 审核日志记录

1. 选择以下选项之一：
   - 使用现有选项组。
   - 创建自定义数据库选项组并使用该选项组。有关更多信息，请参阅 创建选项组。
2. 将 DB2_AUDIT 选项添加到选项组，然后配置选项设置。有关添加选项的更多信息，请参阅 将选项添加到选项组。

- 在 IAM_ROLE_ARN 中，输入您在the section called “创建 IAM 角色并附加您的 IAM 策略”中创建的 IAM 角色的 ARN。
- 在 S3_BUCKET_ARN 中，输入要用于 Db2 审核日志的 S3 存储桶的 ARN。存储桶必须与您的 RDS for Db2 数据库实例位于同一区域。与您输入的 IAM 角色关联的策略必须允许对该资源执行所需的操作。

3. 将选项组应用到新的或现有的数据库实例。选择以下选项之一：

   - 如果您正在创建新的数据库实例，请在启动实例时应用选项组。
   - 在现有数据库实例上，请通过修改实例并附加新的选项组来应用选项组。有关更多信息，请参阅修改 Amazon RDS 数据库实例。

步骤 5：配置审核策略

要为 RDS for Db2 数据库配置审核策略，请使用 RDS for Db2 数据库实例的主用户名和主密码连接到 `rdsadmin` 数据库。然后，使用数据库的数据库名称和适用的参数值调用 `rdsadmin.configure_db_audit` 存储过程。

以下示例连接到数据库并为类别为 AUDIT、CHECKING、OBJMAINT、SECMAINT、SYSADMIN 和 VALIDATE 的 `testdb` 配置审核策略。状态值 `BOTH` 记录成功和失败，`ERROR TYPE` 默认情况下为 `NORMAL`。有关如何使用该存储过程的更多信息，请参阅the section called “rdsadmin.configure_db_audit”。

```
db2 "connect to rdsadmin user master_user using master_password"
db2 "call rdsadmin.configure_db_audit('testdb', 'ALL', 'BOTH', ?)"
```

步骤 6：检查审核配置

要确保您的审核策略设置正确，请检查审核配置的状态。

要检查配置，请使用 RDS for Db2 数据库实例的主用户名和主密码连接到 `rdsadmin` 数据库。然后，使用数据库的数据库名称运行以下 SQL 语句。在以下示例中，数据库名称为 *`testdb`*。

```
db2 "select task_id, task_type, database_name, lifecycle,
    varchar(bson_to_json(task_input_params), 500) as task_params,
    cast(task_output as varchar(500)) as task_output
    from table(rdsadmin.get_task_status(null,'testdb','CONFIGURE_DB_AUDIT'))"

Sample Output

```

```
TASK_ID              TASK_TYPE            DATABASE_NAME       LIFECYCLE
-------------------- -------------------- --------------- -------------
                   2  CONFIGURE_DB_AUDIT            DB2DB       SUCCESS

... continued ...
TASK_PARAMS
--------------------------------------------------------
{ "AUDIT_CATEGORY" : "ALL", "CATEGORY_SETTING" : "BOTH" }

... continued ...
                                        TASK_OUTPUT
---------------------------------------------------
2023-12-22T20:27:03.029Z Task execution has started.

2023-12-22T20:27:04.285Z Task execution has completed successfully.
```

管理 Db2 审核日志

设置 Db2 审核日志记录后，您可以修改特定数据库的审核策略，或者禁用数据库级别或整个数据库实例的审核日志记录。您也可以更改将日志文件上传到其中的 Amazon S3 存储桶。

主题

- 修改 Db2 审核策略
- 修改日志文件的位置
- 禁用 Db2 审核日志

修改 Db2 审核策略

要修改特定 RDS for Db2 数据库的审核策略，请运行 `rdsadmin.configure_db_audit` 存储过程。使用此存储过程，您可以更改审核策略的类别、类别设置和错误类型配置。有关更多信息，请参阅 the section called “rdsadmin.configure_db_audit”。

修改日志文件的位置

要更改将您的日志文件上传到其中的 Amazon S3 存储桶，请执行以下操作之一：

- 修改附加到 RDS for Db2 数据库实例的当前选项组 – 更新 `DB2_AUDIT` 选项的 `S3_BUCKET_ARN` 设置以指向新存储桶。此外，请务必更新附加到附加选项组中的 `IAM_ROLE_ARN` 设置指定的 IAM 角色的 IAM 策略。此 IAM 策略必须为新存储桶提供所需的访问权限。有关 IAM 策略中所需权限的信息，请参阅创建 IAM 策略。

- 将 RDS for Db2 数据库实例附加到其他选项组 - 修改数据库实例以更改附加到该实例的选项组。确保为新选项组配置了正确的 S3_BUCKET_ARN 和 IAM_ROLE_ARN 设置。有关如何为 DB2_AUDIT 选项配置这些设置的信息，请参阅配置选项组。

修改选项组后，请确保立即应用更改。有关更多信息，请参阅 the section called “修改数据库实例”。

禁用 Db2 审核日志

要禁用 Db2 审核日志记录，请执行以下操作之一：

- 禁用 RDS for Db2 数据库实例的审核日志记录 - 修改您的数据库实例并从中删除带有 DB2_AUDIT 选项的选项组。有关更多信息，请参阅 the section called “修改数据库实例”。
- 禁用特定数据库的审核日志记录 - 使用数据库的数据库名称调用 `rdsadmin.disable_db_audit`，停止审核日志记录并删除审核策略。有关更多信息，请参阅 the section called “rdsadmin.disable_db_audit”。

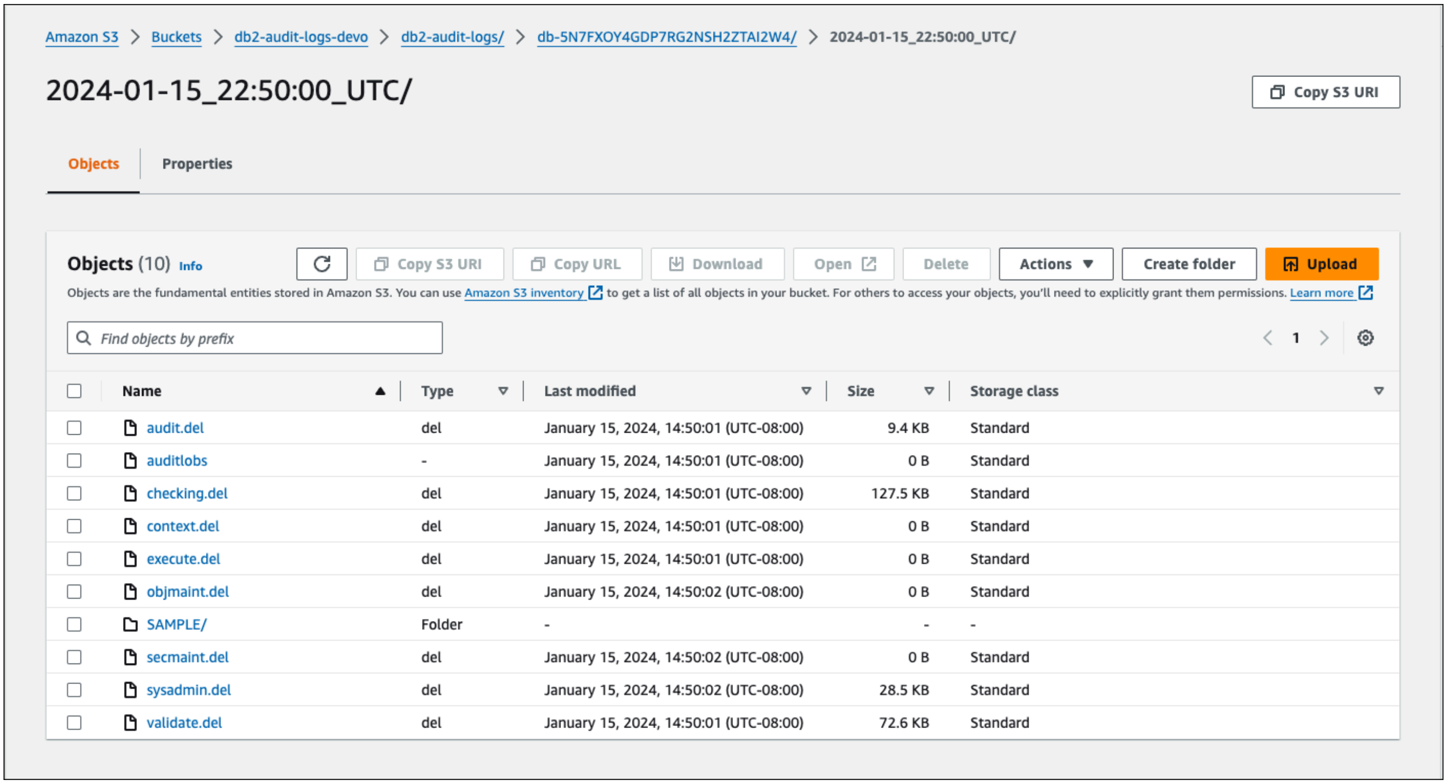
```
db2 "call rdsadmin.disable_db_audit(
    'db_name',
    ?)"
```

查看审核日志

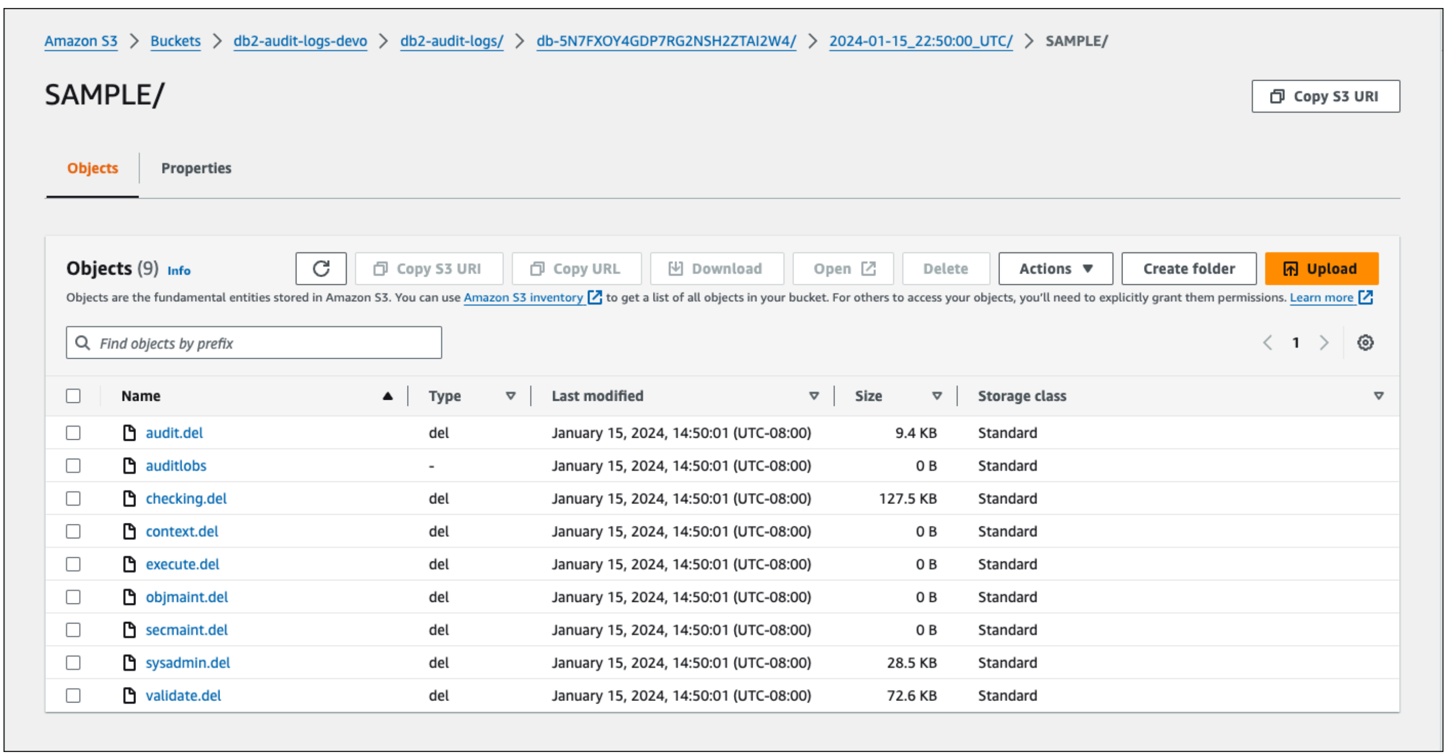
启用 Db2 审核日志记录后，请至少等待一小时，然后再查看 Amazon S3 存储桶中的审核数据。Amazon RDS 会自动将日志从您的 RDS for Db2 数据库实例发送到以下位置：

- 数据库实例级别日志 – `bucket_name/db2-audit-logs/dbi_resource_id/date_time_utc/`
- 数据库级别日志 - `bucket_name/db2-audit-logs/dbi_resource_id/date_time_utc/db_name/`

以下 Amazon S3 控制台屏幕截图示例显示了 RDS for Db2 数据库实例级别日志文件的文件夹列表。



以下 Amazon S3 控制台屏幕截图示例显示了 RDS for Db2 数据库实例的数据库级别日志文件。



排查 Db2 审核日志记录问题

可使用以下信息排查 Db2 审核日志记录的常见问题。

无法配置审核策略

如果调用存储过程 `rdsadmin.configure_db_audit` 返回错误，则可能是包含 `DB2_AUDIT` 选项的选项组未与 RDS for Db2 数据库实例关联。修改数据库实例以添加该选项组，然后尝试再次调用存储过程。有关更多信息，请参阅 修改 Amazon RDS 数据库实例。

Amazon S3 存储桶中没有数据

如果 Amazon S3 存储桶中缺少日志记录数据，请检查以下内容：

- Amazon S3 存储桶与您的 RDs for Db2 数据库实例位于同一区域。
- 您在 `IAM_ROLE_ARN` 选项设置中指定的角色已配置为具有将日志上传到Amazon S3 存储桶所需的权限。有关更多信息，请参阅 创建 IAM 策略。
- 在与您的 RDS for Db2 数据库实例关联的选项组中，`IAM_ROLE_ARN` 和 `S3_BUCKET_ARN` 选项设置的 ARN 正确。有关更多信息，请参阅 配置选项组。

您可以通过连接到数据库并运行 SQL 语句来检查审核日志记录配置的任务状态。有关更多信息，请参阅 检查审核配置。

您还可以查看事件，详细了解日志可能丢失的原因。有关如何查看事件的信息，请参阅the section called “在 Amazon RDS 控制台中查看日志、事件和流”。

Amazon RDS for Db2 的外部存储过程

您可以创建外部例程，并将其作为外部存储过程注册到您的 Amazon RDS for Db2 数据库。目前，RDS for Db2 只支持对外部存储过程使用基于 Java 的例程。

基于 Java 的外部存储过程

基于 Java 的外部存储过程是作为外部存储过程注册到 RDS for Db2 数据库的外部 Java 例程。

主题

基于 Java 的外部存储过程的限制

在开发外部例程之前，请考虑以下限制和局限性。

要创建外部例程，请务必使用 Db2 提供的 Java 开发工具包（JDK）。有关更多信息，请参阅 Java software support for Db2 database products。

您的 Java 程序只能在 `/tmp` 目录中创建文件，而 Amazon RDS 不支持对这些文件启用可执行文件或设置用户 ID（SUID）权限。您的 Java 程序也无法使用套接字系统调用或以下系统调用：

- _sysctl
- acct
- afs_syscall
- bpf
- capset
- chown
- chroot
- create_module
- delete_module
- fanotify_init
- fanotify_mark
- finit_module

- fsconfig
- fsopen
- fspick
- get_kernel_syms
- getpmsg
- init_module
- mount
- move_mount
- nfsservctl
- open_by_handle_at
- open_tree
- pivot_root
- putpmsg
- query_module
- quotactl
- reboot
- security
- setdomainname
- setfsuid
- sethostname
- sysfs
- tuxcall
- umount2
- uselib
- ustat
- vhangup
- vserver

有关 Db2 的外部例程的其它限制，请参阅 IBM Db2 文档中的 Restrictions on external routines 。

配置基于 Java 的外部存储过程

要配置外部存储过程，请使用外部例程创建一个 .jar 文件，将其安装在 RDS for Db2 数据库上，然后将它注册为外部存储过程。

主题

步骤 1：启用外部存储过程

要启用外部存储过程，请在与数据库实例关联的自定义参数组中，将参数 `db2_alternate_authz_behaviour` 设置为以下值之一：

- `EXTERNAL_ROUTINE_DBADM` – 隐式向任何具有 `DBADM` 权限的用户、组或角色授予 `CREATE_EXTERNAL_ROUTINE` 权限。
- `EXTERNAL_ROUTINE_DBAUTH` – 允许具有 `DBADM` 权限的用户向任何用户、组或角色授予 `CREATE_EXTERNAL_ROUTINE` 权限。在这种情况下，任何用户、组或角色都不会被隐式授予此权限，即使是具有 `DBADM` 权限的用户也一样。

有关此设置的更多信息，请参阅 IBM Db2 文档中的 GRANT (database authorities) statement。

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 Amazon RDS API 创建和修改自定义参数组。

控制台

在自定义参数组中配置 db2_alternate_authz_behaviour 参数

1. 如果您想要使用的自定义数据库参数组不同于数据库实例正在使用的自定义数据库参数组，就必须创建一个新的数据库参数组。如果您使用自带许可（BYOL）模式，请确保新的自定义参数组包含 IBM ID。有关这些 ID 的信息，请参阅 the section called “Db2 自带许可的 IBM ID”。有关创建数据库参数组的详细信息，请参阅 在 Amazon RDS 中创建数据库参数组。
2. 在自定义参数组中设置 `db2_alternate_authz_behaviour` 参数的值。有关修改参数组的更多信息，请参阅在 Amazon RDS 中修改数据库参数组中的参数。

Amazon CLI

在自定义参数组中配置 db2_alternate_authz_behaviour 参数

1. 如果您想要使用的自定义数据库参数组不同于数据库实例正在使用的自定义数据库参数组，请运行 create-db-parameter-group 命令创建一个自定义参数组。如果您使用自带许可（BYOL）模式，请确保新的自定义参数组包含 IBM ID。有关这些 ID 的信息，请参阅 the section called “Db2 自带许可的 IBM ID”。

   包括以下必需选项：

   - `--db-parameter-group-name` - 您正在创建的参数组的名称。
   - `--db-parameter-group-family` – Db2 引擎版本和主要版本。有效值为 `db2-se-11.5` 和 `db2-ae-11.5`。
   - `--description` – 此参数组的描述。

   有关创建数据库参数组的详细信息，请参阅 在 Amazon RDS 中创建数据库参数组。

   以下示例显示如何为参数组系列 `db2-se-11.5` 创建名为 `MY_EXT_SP_PARAM_GROUP` 的自定义参数组。

   对于 Linux、macOS 或 Unix：

```
aws rds create-db-parameter-group \
--region us-east-1 \
--db-parameter-group-name MY_EXT_SP_PARAM_GROUP \
--db-parameter-group-family db2-se-11.5 \
--description "test db2 external routines"
```

   对于 Windows：

```
aws rds create-db-parameter-group ^
--region us-east-1 ^
--db-parameter-group-name MY_EXT_SP_PARAM_GROUP ^
--db-parameter-group-family db2-se-11.5 ^
--description "test db2 external routines"
```

2. 通过运行 modify-db-parameter-group 命令修改您的自定义参数组中的 `db2_alternate_authz_behaviour` 参数。

包括以下必需选项：

- `--db-parameter-group-name` – 您创建的参数组的名称。
- `--parameters` – 参数名称、值和参数更新应用方法的数组。

有关修改参数组的更多信息，请参阅在 Amazon RDS 中修改数据库参数组中的参数。

以下示例说明如何通过将 `db2_alternate_authz_behaviour` 的值设置为 `EXTERNAL_ROUTINE_DBADM` 来修改参数组 `MY_EXT_SP_PARAM_GROUP`。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-parameter-group \
    --db-parameter-group-name MY_EXT_SP_PARAM_GROUP \
    --parameters
 "ParameterName='db2_alternate_authz_behaviour',ParameterValue='EXTERNAL_ROUTINE_DBADM',App
```

对于 Windows：

```
aws rds modify-db-parameter-group ^
    --db-parameter-group-name MY_EXT_SP_PARAM_GROUP ^
    --parameters
 "ParameterName='db2_alternate_authz_behaviour',ParameterValue='EXTERNAL_ROUTINE_DBADM',App
```

RDS API

在自定义参数组中配置 db2_alternate_authz_behaviour 参数

1. 如果您想要使用的自定义数据库参数组不同于数据库实例正在使用的自定义数据库参数组，请使用 Amazon RDS API CreateDBParameterGroup 操作创建一个新的数据库参数组。如果您使用自带许可（BYOL）模式，请确保新的自定义参数组包含 IBM Db2 ID。有关这些 ID 的信息，请参阅 the section called “Db2 自带许可的 IBM ID”。

   包括以下必需参数：

   - `DBParameterGroupName`
   - `DBParameterGroupFamily`
   - `Description`

有关创建数据库参数组的详细信息，请参阅 在 Amazon RDS 中创建数据库参数组。

2. 修改您使用 RDS API ModifyDBParameterGroup 操作创建的自定义参数组中的 `db2_alternate_authz_behaviour` 参数。

   包括以下必需参数：

   - `DBParameterGroupName`
   - `Parameters`

   有关修改参数组的更多信息，请参阅在 Amazon RDS 中修改数据库参数组中的参数。

步骤 2：使用外部例程安装 .jar 文件

创建 Java 例程后，创建 .jar 文件，然后运行 `db2 "call sqlj.install_jar('file:`*`file_path`*`',`*`jar_ID`*`)"` 来将它安装到 RDS for Db2 数据库上。

以下示例展示了如何创建 Java 例程并将其安装到 RDS for Db2 数据库上。该示例包括可用于测试该过程的简单例程的示例代码。本示例作出以下假设：

- Java 代码是在安装了 Db2 的服务器上编译的。这是最佳实践，因为不使用 IBM 提供的 JDK 进行编译可能会导致无法解释的错误。
- 服务器对于 RDS for Db2 数据库进行了本地编目。

如果您想使用以下示例代码尝试该过程，请复制它并将它保存到名为 `MYJAVASP.java` 的文件中。

```
import java.sql.*;
public class MYJAVASP
{
public static void my_JAVASP (String inparam) throws SQLException, Exception
{
try
{
// Obtain the calling context's connection details.
Connection myConn = DriverManager.getConnection("jdbc:default:connection");
String myQuery = "INSERT INTO TEST.TEST_TABLE VALUES (?, CURRENT DATE)";
PreparedStatement myStmt = myConn.prepareStatement(myQuery);
myStmt.setString(1, inparam);
myStmt.executeUpdate();
```

```
}
catch (SQLException sql_ex)
{
throw sql_ex;
}
catch (Exception ex)
{
throw ex;
}
}
```

以下命令编译 Java 例程。

```
~/sqllib/java/jdk64/bin/javac MYJAVASP.java
```

以下命令创建 .jar 文件。

```
~/sqllib/java/jdk64/bin/jar cvf MYJAVASP.jar MYJAVASP.class
```

以下命令连接到名为 MY_DB2_DATABASE 的数据库并安装 .jar 文件。

```
db2 "connect to MY_DB2_DATABASE user master_username using master_password"

db2 "call sqlj.install_jar('file:/tmp/MYJAVASP.jar','MYJAVASP')"
db2 "call sqlj.refresh_classes()"
```

步骤 3：注册外部存储过程

在 RDS for Db2 数据库上安装 .jar 文件后，通过运行 db2 CREATE PROCEDURE 或 db2 REPLACE PROCEDURE 命令将其注册为存储过程。

以下示例说明如何连接到数据库，并将在上一步中创建的 Java 例程注册为存储过程。

```
db2 "connect to MY_DB2_DATABASE user master_username using master_password"

create procedure TESTSP.MYJAVASP (in input char(6))
specific myjavasp
dynamic result sets 0
deterministic
language java
parameter style java
```

```
no dbinfo
fenced
threadsafe
modifies sql data
program type sub
external name 'MYJAVASP!my_JAVASP';
```

步骤 4：验证外部存储过程

使用以下步骤来测试在上一步中注册的示例外部存储过程。

验证外部存储过程

1. 在下面的示例中创建一个类似于 TEST.TEST_TABLE 的表。

   ```
   db2 "create table TEST.TEST_TABLE(C1 char(6), C2 date)"
   ```

2. 调用新的外部存储过程。该调用返回的状态为 0。

   ```
   db2 "call TESTSP.MYJAVASP('test')"
   Return Status = 0
   ```

3. 查询您在步骤 1 中创建的表，来验证存储过程调用的结果。

   ```
   db2 "SELECT * from TEST.TEST_TABLE"
   ```

   此查询将生成类似于下面示例的输出：

   ```
   C1     C2
   ------ ----------
   test   02/05/2024
   ```

Amazon RDS for Db2 的已知问题和限制

下面各项是有关使用 Amazon RDS for Db2 的已知问题和限制：

主题

身份验证限制

Amazon RDS 将 db2auth 默认设置为 JCC_ENFORCE_SECMEC。但是，如果您不想通过网络强制使用用户 ID 和密码加密，可以通过将参数组中的 CLEAR_TEXT 参数更改为 db2auth 来覆盖此设置。有关更多信息，请参阅 在 Amazon RDS 中修改数据库参数组中的参数。

非防护例程

RDS for Db2 不支持创建非防护例程，也不支持通过备份和还原数据来迁移这些例程。要检查您的数据库是否包含任何非防护例程，请运行以下 SQL 命令：

```
SELECT 'COUNT:' || count(*) FROM SYSCAT.ROUTINES where fenced='N' and routineschema not
 in ('SQLJ','SYSCAT','SYSFUN','SYSIBM','SYSIBMADM','SYSPROC','SYSTOOLS')
```

迁移期间的非自动存储表空间

RDS for Db2 不支持创建新的非自动存储表空间。当您使用原生还原对数据库进行一次性迁移时，RDS for Db2 会自动将您的非自动存储表空间转换为自动存储表空间，然后将您的数据库还原到 RDS for Db2。有关一次性迁移的信息，请参阅针对 Amazon RDS for Db2 从 Linux 迁移到 Linux和针对 Amazon RDS for Db2 从 AIX 或 Windows 迁移到 Linux。

设置 db2_compatibility_vector 参数

使用 Amazon RDS，您可以在创建数据库实例时创建初始数据库，然后修改关联参数组中的参数。但是，对于 Db2，如果您要在参数组中设置 db2_compatibility_vector 参数，

则必须先在自定义参数组中修改此参数，在没有数据库的情况下创建数据库实例，然后使用 rdsadmin.create_database 存储过程创建数据库。

设置 **db2_compatibility_vector** 参数

1. 创建自定义参数组。（您无法修改默认参数组中的参数。）
2. 修改参数。
3. 创建数据库实例。
4. 使用 rdsadmin.create_database 存储过程创建数据库。
5. 将参数组与包含数据库的数据库实例相关联。

迁移包含无效包的数据库

如果您使用 RESTORE 命令，将包含 INVALID 包的 Db2 数据库迁移到 RDS for Db2，则在开始使用数据库时可能会遇到问题。由于数据库实例用户 rdsdb 的授权设置以及删除了从 PUBLIC 授权，INVALID 包可能会导致问题。INVALID 包会导致以下命令失败：

- db2updv115
- db2 "call SYSPROC.ADMIN_REVALIDATE_DB_OBJECTS()"

在使用 RESTORE 命令迁移数据库之前，请运行以下命令来确保数据库中不包含 INVALID 包：

```
db2 "select count(*) from sysibm.sysroutines where valid = 'N'"
```

如果命令返回的计数大于零，则调用以下命令：

```
db2 "call SYSPROC.ADMIN_REVALIDATE_DB_OBJECTS()"
```

然后，调用前面的命令以确认您的数据库中不再包含 INVALID 包。

```
db2 "select count(*) from sysibm.sysroutines where valid = 'N'"
```

现在，您已做好准备，可以备份数据库并将其还原到 RDS for Db2 数据库实例。

Amazon RDS for Db2 存储过程参考

可以通过调用内置存储过程来管理运行 Db2 引擎的 Amazon RDS for Db2 数据库实例。

存储过程	类别	描述
the section called “rdsadmin.activate_database”	数据库	使用 `rdsadmin.activate_database` 存储过程可激活独立 RDS for Db2 数据库实例上的数据库。
the section called “rdsadmin.add_groups”	授予和撤销权限	使用 `rdsadmin.add_groups` 存储过程可为 RDS for Db2 数据库实例上数据库的用户添加一个或多个组。
the section called “rdsadmin.add_user”	授予和撤销权限	使用 `rdsadmin.add_user` 存储过程可将用户添加到 RDS for Db2 数据库实例上数据库的授权列表中。
the section called “rdsadmin.alter_bufferpool”	缓冲池	使用 `rdsadmin.alter_bufferpool` 存储过程可修改 RDS for Db2 数据库实例上数据库的缓冲池。
the section called “rdsadmin.alter_tablespace”	表空间	使用 `rdsadmin.alter_tablespace` 存储过程可修改 RDS for Db2 数据库实例上数据库的表空间。
the section called “rdsadmin.catalog_storage_access”	存储访问	使用 `rdsadmin.catalog_storage_access` 存储过程可对存储别名进行编目，以访问 RDS for Db2 数据库实例上数据库的带有 Db2 数据文件的 Amazon S3 存储桶。
the section called “rdsadmin.change_password”	授予和撤销权限	使用 `rdsadmin.change_password` 存储过程可更改 RDS for Db2 数据库实例上数据库的用户密码。
the section called “rdsadmin.complete_rollforward”	数据库	使用 `rdsadmin.complete_rollforward` 存储过程可使 RDS for Db2 数据库实例上的数据库从 `ROLL-FORWARD PENDING` 状态联机。当您调用 the section called “rdsadmin

存储过程	类别	描述
		.rollforward_database" 但将 complete_rollforward 参数设置为 FALSE 时，会进入 ROLL-FORWARD PENDING 状态。
the section called "rdsadmin.configure_db_audit"	审计策略	使用 rdsadmin.configure_db_audit 存储过程可修改 RDS for Db2 数据库实例上数据库的审计策略。如果审计策略不存在，则运行此存储过程将创建审计策略。
the section called "rdsadmin.create_bufferpool"	缓冲池	使用 rdsadmin.create_bufferpool 存储过程可为 RDS for Db2 数据库实例上的数据库创建缓冲池。
the section called "rdsadmin.create_database"	数据库	使用 rdsadmin.create_database 存储过程可在 RDS for Db2 数据库实例上创建数据库。
the section called "rdsadmin.create_role"	授予和撤销权限	使用 rdsadmin.create_role 存储过程可创建要附加到 RDS for Db2 数据库实例上数据库的角色。
the section called "rdsadmin.create_tablespace"	表空间	使用 rdsadmin.create_tablespace 存储过程可为 RDS for Db2 数据库实例上的数据库创建表空间。
the section called "rdsadmin.db2pd_command"	数据库	使用 rdsadmin.db2pd_command 存储过程可收集有关 RDS for Db2 数据库实例上数据库的信息。借助此信息可监控 RDS for Db2 中的数据库并排查其问题。
the section called "rdsadmin.dbadm_grant"	授予和撤销权限	使用 rdsadmin.dbadm_grant 存储过程可向 RDS for Db2 数据库实例上数据库的一个或多个角色、用户或组授予一种或多种授权类型（DBADM、ACCESSCTRL 或 DATAACCESS）。

存储过程	类别	描述
the section called “rdsadmin.dbadm_revoke”	授予和撤销权限	使用 `rdsadmin.dbadm_revoke` 存储过程可从 RDS for Db2 数据库实例上数据库的一个或多个角色、用户或组撤销一种或多种授权类型（`DBADM`、`ACCESSCTRL` 或 `DATAACCESS`）。
the section called “rdsadmin.deactivate_database”	数据库	使用 `rdsadmin.deactivate_database` 存储过程可停用 RDS for Db2 数据库实例上的数据库。您可以停用数据库以节省内存资源。
the section called “rdsadmin.disable_db_audit”	审计策略	使用 `rdsadmin.disable_db_audit` 存储过程可停止审计日志记录，并从 RDS for Db2 数据库实例上的数据库中移除审计策略。
the section called “rdsadmin.drop_bufferpool”	缓冲池	使用 `rdsadmin.drop_bufferpool` 存储过程可从 RDS for Db2 数据库实例上的数据库中删除缓冲池。
the section called “rdsadmin.drop_database”	数据库	使用 `rdsadmin.drop_database` 存储过程可从 RDS for Db2 数据库实例中删除数据库。
the section called “rdsadmin.drop_role”	授予和撤销权限	使用 `rdsadmin.drop_role` 存储过程可从 RDS for Db2 数据库实例上的数据库中删除角色。
the section called “rdsadmin.drop_tablespace”	表空间	使用 `rdsadmin.drop_tablespace` 存储过程可从 RDS for Db2 数据库实例上的数据库中删除表空间。
the section called “rdsadmin.force_application”	数据库	使用 `rdsadmin.force_application` 存储过程可强制应用程序退出 RDS for Db2 数据库实例上的数据库以执行维护。

存储过程	类别	描述
the section called “rdsadmin.grant_role”	授予和撤销权限	使用 `rdsadmin.grant_role` 存储过程可为 RDS for Db2 数据库实例上数据库中的被授予者角色、用户或组分配角色。也可以使用此存储过程向被授予者角色 DBADM 授予分配角色的权限。
the section called “rdsadmin.list_archive_log_information”	数据库	使用 `rdsadmin.list_archive_log_information` 存储过程可返回有关 RDS for Db2 数据库实例上数据库的归档日志的信息。这些信息包括诸如各个日志文件的大小和创建日期，以及归档日志文件使用的总存储空间之类的详细信息。
the section called “rdsadmin.list_users”	授予和撤销权限	使用 `rdsadmin.list_users` 存储过程可针对 RDS for Db2 数据库实例上的数据库返回授权列表中的用户列表。
the section called “rdsadmin.reactivate_database”	数据库	使用 `rdsadmin.reactivate_database` 存储过程，可在更改数据库配置后重新激活 RDS for Db2 数据库实例上的数据库。对于独立数据库实例上的数据库，您可以使用此存储过程或 rdsadmin.activate_database 存储过程。
the section called “rdsadmin.remove_groups”	授予和撤销权限	使用 `rdsadmin.remove_groups` 存储过程可从 RDS for Db2 数据库实例上数据库的用户中移除一个或多个组。
the section called “rdsadmin.remove_user”	授予和撤销权限	使用 `rdsadmin.remove_user` 存储过程可从 RDS for Db2 数据库实例上数据库的授权列表中移除用户。
the section called “rdsadmin.rename_tablespace”	表空间	使用 `rdsadmin.rename_tablespace` 存储过程可重命名 RDS for Db2 数据库实例上数据库的表空间。

存储过程	类别	描述
the section called “rdsadmin.restore_database”	数据库	使用 `rdsadmin.restore_database` 存储过程可从 Amazon S3 存储桶中还原 RDS for Db2 数据库实例上的数据库。
the section called “rdsadmin.revoke_role”	授予和撤销权限	使用 `rdsadmin.revoke_role` 存储过程可从 RDS for Db2 数据库实例上数据库中的被授予者角色、用户或组撤销角色。
the section called “rdsadmin.rollforward_database”	数据库	使用 `rdsadmin.rollforward_database` 存储过程可使 RDS for Db2 数据库实例上的数据库联机，并在通过调用 the section called “rdsadmin.restore_database” 还原 RDS for Db2 数据库实例上的数据库后应用事务日志。
the section called “rdsadmin.set_archive_log_retention”	数据库	使用 `rdsadmin.set_archive_log_retention` 存储过程可为 RDS for Db2 数据库实例上的数据库配置保留归档日志文件的时间。还可以使用此存储过程禁用归档日志保留功能。
the section called “rdsadmin.set_configuration”	数据库	使用 `rdsadmin.set_configuration` 存储过程可为 RDS for Db2 数据库实例上的数据库配置某些设置。
the section called “rdsadmin.show_archive_log_retention”	数据库	使用 `rdsadmin.show_archive_log_retention` 存储过程可返回 RDS for Db2 数据库实例上数据库的当前归档日志保留设置。
the section called “rdsadmin.show_configuration”	数据库	使用 `rdsadmin.show_configuration` 存储过程可返回 RDS for Db2 数据库实例上数据库的一个或多个可修改的设置。
the section called “rdsadmin.uncatalog_storage_access”	存储访问	使用 `rdsadmin.uncatalog_storage_access` 存储过程可删除用于访问包含 Db2 数据文件的 Amazon S3 存储桶的存储别名。

存储过程	类别	描述
the section called “rdsadmin.update_db_param”	数据库	使用 `rdsadmin.update_db_param` 存储过程可为 RDS for Db2 数据库实例上的数据库更新数据库参数。

主题

Amazon RDS for Db2 存储过程注意事项

在对运行 Db2 引擎的 RDS for Db2 数据库实例使用 Amazon RDS 系统存储过程之前，应了解以下信息：

- 运行存储过程之前，必须首先以 RDS for Db2 数据库实例的主用户身份连接到 `rdsadmin` 数据库。在以下示例中，用您自己的信息替换 *`master_username`* 和 *`master_password`*。

```
db2 "connect to rdsadmin user master_username using master_password"
```

- 存储过程返回 `ERR_MESSAGE` 参数，该参数指示存储过程是否成功运行以及未成功运行的原因。

  示例

  以下示例指示存储过程已成功运行。

```
Parameter Name : ERR_MESSAGE
Parameter Value : -
Return Status = 0
```

  以下示例表明，由于存储过程中使用的 Amazon S3 存储桶名称无效，存储过程未成功运行。

```
Parameter Name : ERR_MESSAGE
Parameter Value : Invalid S3 bucket name
Return Status = -1006
```

有关调用存储过程时返回的错误消息，请参阅the section called “存储过程错误”。

有关如何检查存储过程状态的信息，请参阅rdsadmin.get_task_status。

用于对 RDS for Db2 授予和撤销权限的存储过程

本主题中介绍的内置存储过程可管理 Amazon RDS for Db2 数据库的用户、角色、组和授权。要运行这些过程，主用户必须先连接到 `rdsadmin` 数据库。

有关使用这些存储过程的任务，请参阅the section called “授予和撤销权限”。

请参阅以下内置存储过程，了解其语法、参数、用法说明和示例。

存储过程

- rdsadmin.create_role
- rdsadmin.grant_role
- rdsadmin.revoke_role
- rdsadmin.drop_role
- rdsadmin.add_user
- rdsadmin.change_password
- rdsadmin.list_users
- rdsadmin.remove_user
- rdsadmin.add_groups
- rdsadmin.remove_groups
- rdsadmin.dbadm_grant
- rdsadmin.dbadm_revoke

rdsadmin.create_role

创建角色。

语法

```
db2 "call rdsadmin.create_role(
    'database_name',
    'role_name')"
```

参数

以下参数为必需参数：

database_name

命令将在其上运行的数据库的名称。数据类型为 varchar。

role_name

您要创建的角色的名称。数据类型为 varchar。

使用说明

有关检查创建角色的状态的信息，请参阅rdsadmin.get_task_status。

示例

以下示例为数据库 DB2DB 创建一个名为 MY_ROLE 的角色。

```
db2 "call rdsadmin.create_role(
    'DB2DB',
    'MY_ROLE')"
```

rdsadmin.grant_role

将角色分配给角色、用户或组。

语法

```
db2 "call rdsadmin.grant_role(
    ?,
    'database_name',
    'role_name',
    'grantee',
    'admin_option')"
```

参数

以下输出参数是必需参数：

?

一个参数标记，用于输出任务的唯一标识符。此参数仅接受 ?。

以下输入参数是必需参数：

database_name

命令将在其上运行的数据库的名称。数据类型为 varchar。

role_name

您要创建的角色的名称。数据类型为 varchar。

grantee

要接收授权的角色、用户或组。数据类型为 varchar。有效值：ROLE、USER、GROUP、PUBLIC。

格式必须是值后跟名称。用英文逗号分隔多个值和名称。示例：“USER *user1*, *user2*, GROUP *group1*, *group2*”。将名称替换为您自己的信息。

以下输入参数为可选参数：

admin_option

指定被授权者 ROLE 是否具有分配角色的 DBADM 授权。数据类型为 char。默认为 N。

使用说明

有关检查分配角色的状态的信息，请参阅[rdsadmin.get_task_status]。

示例

示例 1：为角色、用户和组分配角色并授予权限

下面的示例将对于数据库 TESTDB 的名为 ROLE_TEST 的角色分配给名为 role1 的角色、名为 user1 的用户和名为 group1 的组。ROLE_TEST 获得分配角色的管理员授权。

```
db2 "call rdsadmin.grant_role(
    ?,
    'TESTDB',
    'ROLE_TEST',
    'ROLE role1, USER user1, GROUP group1',
    'Y')"
```

示例 2：为角色分配 **PUBLIC** 但不授予权限

下面的示例将针对数据库 TESTDB 的名为 ROLE_TEST 的角色分配给 PUBLIC。ROLE_TEST 未获得分配角色的管理员授权。

```
db2 "call rdsadmin.grant_role(
    ?,
    'TESTDB',
    'ROLE_TEST',
    'PUBLIC')"
```

rdsadmin.revoke_role

从角色、用户或组撤销角色。

语法

```
db2 "call rdsadmin.revoke_role(
    ?,
    'database_name',
    'role_name',
    'grantee')"
```

参数

以下输出参数是必需参数：

?

一个参数标记，用于输出任务的唯一标识符。此参数仅接受 ?。

以下输入参数是必需参数：

database_name

命令将在其上运行的数据库的名称。数据类型为 varchar。

role_name

您要撤销的角色的名称。数据类型为 varchar。

grantee

要失去授权的角色、用户或组。数据类型为 varchar。有效值：ROLE、USER、GROUP、PUBLIC。

格式必须是值后跟名称。用英文逗号分隔多个值和名称。示例："USER *user1*, *user2*, GROUP *group1*, *group2*"。将名称替换为您自己的信息。

使用说明

有关如何检查角色撤销状态的信息，请参阅rdsadmin.get_task_status。

示例

示例 1：从角色、用户和组中撤消角色

以下示例从名为 role1 的角色、名为 user1 的用户和名为 group1 的组中撤销数据库 TESTDB 的名为 ROLE_TEST 的角色。

```
db2 "call rdsadmin.revoke_role(
    ?,
    'TESTDB',
    'ROLE_TEST',
    'ROLE role1, USER user1, GROUP group1')"
```

示例 2：从 **PUBLIC** 撤消角色

以下示例从 PUBLIC 撤销数据库 TESTDB 的名为 ROLE_TEST 的角色。

```
db2 "call rdsadmin.revoke_role(
    ?,
    'TESTDB',
    'ROLE_TEST',
    'PUBLIC')"
```

rdsadmin.drop_role

删除角色。

语法

```
db2 "call rdsadmin.drop_role(
    ?,
    'database_name',
    'role_name')"
```

参数

以下输出参数是必需参数：

?

一个参数标记，用于输出任务的唯一标识符。此参数仅接受 ?。

以下输入参数是必需参数：

database_name

命令将在其上运行的数据库的名称。数据类型为 varchar。

role_name

您要删除的角色的名称。数据类型为 varchar。

使用说明

有关如何检查角色删除状态的信息，请参阅[rdsadmin.get_task_status](#)。

示例

以下示例删除数据库 TESTDB 中名为 ROLE_TEST 的角色。

```
db2 "call rdsadmin.drop_role(
    ?,
    'TESTDB',
    'ROLE_TEST')"
```

rdsadmin.add_user

将用户添加到授权列表。

语法

```
db2 "call rdsadmin.add_user(
    'username',
    'password',
    'group_name,group_name')"
```

参数

以下参数为必需参数：

username

用户的用户名。数据类型为 varchar。

##

用户的密码。数据类型为 varchar。

以下参数为可选参数：

group_name

要将用户添加到的组的名称。数据类型为 varchar。默认值是空字符串或 Null。

使用说明

您可以将用户添加到一个组，也可以通过用逗号分隔组名称添加到多个组。

您可以在创建新用户时创建组，也可以在向现有用户添加组时创建组。无法单独创建组。

> **Note**
>
> 您可以通过调用 rdsadmin.add_user 添加的最大用户数为 5000。

有关检查添加用户的状态的信息，请参阅rdsadmin.get_task_status。

示例

以下示例创建了一个名为 jorge_souza 的用户，并将该用户分配给名为 sales 和 inside_sales 的组。

```
db2 "call rdsadmin.add_user(
    'jorge_souza',
    '*******',
    'sales,inside_sales')"
```

rdsadmin.change_password

更改用户的密码。

语法

```
db2 "call rdsadmin.change_password(
    'username',
    'new_password')"
```

参数

以下参数为必需参数：

username

用户的用户名。数据类型为 varchar。

new_password

用户的新密码。数据类型为 varchar。

使用说明

有关检查更改密码的状态的信息，请参阅rdsadmin.get_task_status。

示例

以下示例更改 jorge_souza 的密码。

```
db2 "call rdsadmin.change_password(
    'jorge_souza',
    '*******')"
```

rdsadmin.list_users

列出授权列表中的用户。

语法

```
db2 "call rdsadmin.list_users()"
```

使用说明

有关检查列出用户的状态的信息，请参阅rdsadmin.get_task_status。

rdsadmin.remove_user

将用户从授权列表中移除。

语法

```
db2 "call rdsadmin.remove_user('username')"
```

参数

以下参数是必需参数：

username

用户的用户名。数据类型为 varchar。

使用说明

有关检查移除用户的状态的信息，请参阅rdsadmin.get_task_status。

示例

以下示例移除 jorge_souza 访问 RDS for Db2 数据库实例中的数据库的权限。

```
db2 "call rdsadmin.remove_user('jorge_souza')"
```

rdsadmin.add_groups

向用户添加组。

语法

```
db2 "call rdsadmin.add_groups(
    'username',
    'group_name,group_name')"
```

参数

以下参数为必需参数：

username

用户的用户名。数据类型为 varchar。

group_name

要将用户添加到的组的名称。数据类型为 varchar。默认值是空字符串。

使用说明

您可以向用户添加一个组，也通过用逗号分隔组名称来添加多个组。有关检查添加组的状态的信息，请参阅rdsadmin.get_task_status。

示例

以下示例将 direct_sales 和 b2b_sales 组添加到用户 jorge_souza。

```
db2 "call rdsadmin.add_groups(
    'jorge_souza',
    'direct_sales,b2b_sales')"
```

rdsadmin.remove_groups

从用户中移除组。

语法

```
db2 "call rdsadmin.remove_groups(
    'username',
    'group_name,group_name')"
```

参数

以下参数为必需参数：

username

用户的用户名。数据类型为 varchar。

group_name

要从中移除用户的组的名称。数据类型为 varchar。

使用说明

可以从用户中移除一个组，也可以通过用逗号分隔组名称来移除多个组。

有关检查移除组的状态的信息，请参阅rdsadmin.get_task_status。

示例

以下示例从用户 jorge_souza 中移除 direct_sales 和 b2b_sales 组。

```
db2 "call rdsadmin.remove_groups(
    'jorge_souza',
    'direct_sales,b2b_sales')"
```

rdsadmin.dbadm_grant

向角色、用户或组授予 DBADM、ACCESSCTRL 或 DATAACCESS 授权。

语法

```
db2 "call rdsadmin.dbadm_grant(
    ?,
    'database_name',
    'authorization',
    'grantee')"
```

参数

以下输出参数是必需参数：

?

一个参数标记，用于输出任务的唯一标识符。此参数仅接受 ?。

以下输入参数是必需参数：

database_name

命令将在其上运行的数据库的名称。数据类型为 varchar。

##

要授予的授权类型。数据类型为 varchar。有效值：DBADM、ACCESSCTRL、DATAACCESS。

使用逗号分隔多个类型。

grantee

要接收授权的角色、用户或组。数据类型为 varchar。有效值：ROLE、USER、GROUP。

格式必须是值后跟名称。用英文逗号分隔多个值和名称。示例：“USER *user1*, *user2*, GROUP *group1*, *group2*”。将名称替换为您自己的信息。

使用说明

接收访问权限的角色必须存在。

有关检查授予数据库管理员访问权限的状态的信息，请参阅rdsadmin.get_task_status。

示例

示例 1：向角色授予数据库管理员访问权限

以下示例向名为 TESTDB 的数据库中的 ROLE_DBA 角色授予数据库管理员访问权限。

```
db2 "call rdsadmin.dbadm_grant(
    ?,
    'TESTDB',
    'DBADM',
    'ROLE ROLE_DBA')"
```

示例 2：向用户和组授予数据库管理员访问权限

以下示例向名为 TESTDB 的数据库中的 user1 和 group1 授予数据库管理员访问权限。

```
db2 "call rdsadmin.dbadm_grant(
    ?,
    'TESTDB',
    'DBADM',
    'USER user1, GROUP group1')"
```

示例 3：向多个用户和组授予数据库管理员访问权限

以下示例向名为 TESTDB 的数据库中的 user1、user2、group1 和 group2 授予数据库管理员访问权限。

```
db2 "call rdsadmin.dbadm_grant(
```

```
    ?,
    'TESTDB',
    'DBADM',
    'USER user1, user2, GROUP group1, group2')"
```

rdsadmin.dbadm_revoke

从角色、用户或组撤销 DBADM、ACCESSCTRL 或 DATAACCESS 授权。

语法

```
db2 "call rdsadmin.dbadm_revoke(
    ?,
    'database_name',
    'authorization',
    'grantee')"
```

参数

以下输出参数是必需参数：

?

任务的唯一标识符。此参数仅接受 ?。

以下输入参数是必需参数：

database_name

命令将在其上运行的数据库的名称。数据类型为 varchar。

##

要撤销的授权类型。数据类型为 varchar。有效值：DBADM、ACCESSCTRL、DATAACCESS。

使用逗号分隔多个类型。

grantee

要从中撤销授权的角色、用户或组。数据类型为 varchar。有效值：ROLE、USER、GROUP。

格式必须是值后跟名称。用英文逗号分隔多个值和名称。示例：“USER *user1*, *user2*, GROUP *group1*, *group2*”。将名称替换为您自己的信息。

使用说明

有关检查撤销数据库管理员访问权限的状态的信息，请参阅rdsadmin.get_task_status。

示例

示例 1：撤消角色的数据库管理员访问权限

以下示例对名为 TESTDB 的数据库中的 ROLE_DBA 角色撤销数据库管理员访问权限。

```
db2 "call rdsadmin.dbadm_revoke(
    ?,
    'TESTDB',
    'DBADM',
    'ROLE ROLE_DBA')"
```

示例 2：撤消用户和群组的数据库管理员访问权限

以下示例对名为 TESTDB 的数据库中的 user1 和 group1 撤销数据库管理员访问权限。

```
db2 "call rdsadmin.dbadm_revoke(
    ?,
    'TESTDB',
    'DBADM',
    'USER user1, GROUP group1')"
```

示例 3：撤消多个用户和组的数据库管理员访问权限

以下示例对名为 TESTDB 的数据库中的 user1、user2、group1 和 group2 撤销数据库管理员访问权限。

```
db2 "call rdsadmin.dbadm_revoke(
    ?,
    'TESTDB',
    'DBADM',
    'USER user1, user2, GROUP group1, group2')"
```

用于 RDS for Db2 的审计策略的存储过程

本主题中介绍的内置存储过程可管理使用审计日志记录的 Amazon RDS for Db2 数据库的审计策略。有关更多信息，请参阅 the section called “Db2 审核日志记录”。要运行这些过程，主用户必须先连接到 `rdsadmin` 数据库。

请参阅以下内置存储过程，了解其语法、参数、用法说明和示例。

存储过程

- rdsadmin.configure_db_audit
- rdsadmin.disable_db_audit

rdsadmin.configure_db_audit

为 *db_name* 指定的 RDS for Db2 数据库配置审核策略。如果您正在配置的策略不存在，则调用此存储过程即可创建该策略。如果该策略确实存在，则调用此存储过程会使用您提供的参数值对其进行修改。

语法

```
db2 "call rdsadmin.configure_db_audit(
    'db_name',
    'category',
    'category_setting',
    '?')"
```

参数

以下参数为必需参数。

db_name

要为其配置审核策略的 RDS for Db2 数据库的数据库名称。数据类型为 `varchar`。

category

要为其配置此审核策略的类别的名称。数据类型为 `varchar`。此参数的有效值如下所示：

- `ALL` - 选择 `ALL` 时，Amazon RDS 将不包含 `CONTEXT`、`EXECUTE` 或 `ERROR` 类别。
- `AUDIT`
- `CHECKING`
- `CONTEXT`

- ERROR
- EXECUTE - 您可以为此类别配置数据，也可以不配置数据。配置数据意味着还要记录为任何主机变量和参数标记提供的输入数据值。默认为不配置数据。有关更多信息，请参阅 *category_setting* 参数描述和the section called “示例”。
- OBJMAINT
- SECMAINT
- SYSADMIN
- VALIDATE

有关这些类别的更多信息，请参阅 IBM Db2 文档。

category_setting

指定审核类别的设置。数据类型为 varchar。

下表显示了每个类别的有效类别设置值。

类别	有效类别设置
ALL AUDIT CHECKING CONTEXT OBJMAINT SECMAINT SYSADMIN VALIDATE	BOTH|FAILURE|SUCCESS|NONE
ERROR	AUDIT|NORMAL 。默认值为 NORMAL。
EXECUTE	BOTH,WITH|BOTH,WITHOUT|FAILURE,WITH|FAILURE,WITHOUT|SUCCESS,WITH|SUCCESS,WITHOUT|NONE

使用说明

在调用 rdsadmin.configure_db_audit 之前，请确保包含您正在为其配置审核策略的数据库的 RDS for Db2 数据库实例与包含 DB2_AUDIT 选项的选项组相关联。有关更多信息，请参阅 the section called “设置 Db2 审核日志记录”。

配置审核策略后，您可以按照检查审核配置中的步骤检查数据库的审核配置状态。

为 category 参数指定 ALL 时不包括 CONTEXT、EXECUTE 或 ERROR 类别。要将这些类别添加到审核策略中，请对您要添加的每个类别分别调用 rdsadmin.configure_db_audit。有关更多信息，请参阅 the section called “示例”。

示例

以下示例为名为 TESTDB 的数据库创建或修改审核策略。在示例 1 至示例 5 中，如果之前未配置 ERROR 类别，则该类别将设置为 NORMAL（默认）。要将该设置更改为 AUDIT，请按照Example 6: Specifying the ERROR category操作。

示例 1：指定 **ALL** 类别

```
db2 "call rdsadmin.configure_db_audit('TESTDB', 'ALL', 'BOTH', ?)"
```

在此示例中，调用在审核策略中配置了 AUDIT、CHECKING、OBJMAINT、SECMAINT、SYSADMIN 和 VALIDATE 类别。指定 BOTH 意味着将对其中每个类别的成功和失败事件进行审核。

示例 2：使用数据指定 **EXECUTE** 类别

```
db2 "call rdsadmin.configure_db_audit('TESTDB', 'EXECUTE', 'SUCCESS,WITH', ?)"
```

在此示例中，调用在审核策略中配置了 EXECUTE 类别。指定 SUCCESS,WITH 意味着此类别的日志将仅包含成功事件，并将包括为主机变量和参数标记提供的输入数据值。

示例 3：在没有数据的情况下指定 **EXECUTE** 类别

```
db2 "call rdsadmin.configure_db_audit('TESTDB', 'EXECUTE', 'FAILURE,WITHOUT', ?)"
```

在此示例中，调用在审核策略中配置了 EXECUTE 类别。指定 FAILURE,WITHOUT 意味着此类别的日志将仅包含失败事件，不包括为主机变量和参数标记提供的输入数据值。

示例 4：指定不带状态事件的 **EXECUTE** 类别

```
db2 "call rdsadmin.configure_db_audit('TESTDB', 'EXECUTE', 'NONE', ?)"
```

在此示例中，调用在审核策略中配置了 EXECUTE 类别。指定 NONE 意味着将不对此类别中的任何事件进行审核。

示例 5：指定 **OBJMAINT** 类别

```
db2 "call rdsadmin.configure_db_audit('TESTDB', 'OBJMAINT', 'NONE', ?)"
```

在此示例中，调用在审核策略中配置了 OBJMAINT 类别。指定 NONE 意味着将不对此类别中的任何事件进行审核。

示例 6：指定 **ERROR** 类别

```
db2 "call rdsadmin.configure_db_audit('TESTDB', 'ERROR', 'AUDIT', ?)"
```

在此示例中，调用在审核策略中配置了 ERROR 类别。指定 AUDIT 意味着将在日志中捕获所有错误（包括审核日志记录本身中发生的错误）。默认错误类型为 NORMAL。使用 NORMAL，审核生成的错误将被忽略，并且仅捕获与正在执行的操作相关的错误的 SQLCODE。

rdsadmin.disable_db_audit

停止对 *db_name* 指定的 RDS for Db2 数据库进行审核日志记录，并删除为其配置的审核策略。

> **Note**
>
> 此存储过程仅删除通过调用 the section called “rdsadmin.configure_db_audit” 配置的审核策略。

语法

```
db2 "call rdsadmin.disable_db_audit('db_name', ?)"
```

参数

以下参数为必需参数。

db_name

要对其禁用审核日志记录的 RDS for Db2 数据库的数据库名称。数据类型为 `varchar`。

使用说明

调用 `rdsadmin.disable_db_audit` 不会对 RDS for Db2 数据库实例禁用审核日志记录。要在数据库实例级别禁用审核日志记录，请从数据库实例中删除选项组。有关更多信息，请参阅 禁用 Db2 审核日志。

示例

以下示例针对名为 `TESTDB` 的数据库禁用审核日志记录。

```
db2 "call rdsadmin.disable_db_audit('TESTDB', ?)"
```

用于管理 RDS for Db2 缓冲池的存储过程

本主题中介绍的内置存储过程可管理 Amazon RDS for Db2 数据库的缓冲池。要运行这些过程，主用户必须先连接到 `rdsadmin` 数据库。

这些存储过程可用于执行各种任务。此列表并不详尽。

- 缓冲池的常见任务
- 生成性能报告
- 使用 db2look 复制数据库元数据
- 为 IBM Db2 Data Management Console 创建存储库数据库

请参阅以下内置存储过程，了解其语法、参数、用法说明和示例。

存储过程

- rdsadmin.create_bufferpool
- rdsadmin.alter_bufferpool
- rdsadmin.drop_bufferpool

rdsadmin.create_bufferpool

创建缓冲池。

语法

```
db2 "call rdsadmin.create_bufferpool(
    'database_name',
    'buffer_pool_name',
    buffer_pool_size,
    'immediate',
    'automatic',
    page_size,
    number_block_pages,
    block_size)"
```

参数

以下参数为必需参数：

database_name

要在其上运行命令的数据库的名称。数据类型为 varchar。

buffer_pool_name

要创建的缓冲池的名称。数据类型为 varchar。

以下参数可选：

buffer_pool_size

缓冲池的大小，以页数表示。数据类型为 integer。默认为 -1。

immediate

指定命令是否立即运行。数据类型为 char。默认为 Y。

automatic

指定是否将缓冲池设置为自动。数据类型为 char。默认为 Y。

page_size

缓冲池的页面大小。数据类型为 integer。有效值：4096、8192、16384、32768。默认为 8192。

number_block_pages

缓冲池中的块页面数。数据类型为 integer。默认为 0。

block_size

块页面的块大小。数据类型为 integer。有效值：2 到 256。默认为 32。

使用说明

有关检查创建缓冲池的状态的信息，请参阅rdsadmin.get_task_status。

示例

示例 1：使用默认参数创建缓冲池

以下示例使用默认参数为名为 TESTDB 的数据库创建名为 BP8 的缓冲池，因此该缓冲池使用 8KB 的页面大小。

```
db2 "call rdsadmin.create_bufferpool(
    'TESTDB',
    'BP8')"
```

示例 2：创建缓冲池以便通过自动分配立即运行

以下示例为名为 TESTDB 的数据库创建一个名为 BP16 的缓冲池，该缓冲池使用 16KB 的页面大小，初始页面计数为 1000，并且设置为自动。Db2 会立即运行该命令。如果您使用的初始页面计数为 -1，则 Db2 将使用自动分配页面。

```
db2 "call rdsadmin.create_bufferpool(
    'TESTDB',
    'BP16',
    1000,
    'Y',
    'Y',
    16384)"
```

示例 3：使用块页面创建缓冲池以立即运行

以下示例为名为 TESTDB 的数据库创建名为 BP16 的缓冲池。此缓冲池的页面大小为 16KB，初始页面计数为 10000。Db2 使用 500 个块页面立即运行该命令，块大小为 512。

```
db2 "call rdsadmin.create_bufferpool(
    'TESTDB',
    'BP16',
    10000,
    'Y',
    'Y',
    16384,
    500,
    512)"
```

rdsadmin.alter_bufferpool

更改缓冲池。

语法

```
db2 "call rdsadmin.alter_bufferpool(
```

```
    'database_name',
    'buffer_pool_name',
    buffer_pool_size,
    'immediate',
    'automatic',
    change_number_blocks,
    number_block_pages,
    block_size)"
```

参数

以下参数为必需参数：

database_name

要在其上运行命令的数据库的名称。数据类型为 varchar。

buffer_pool_name

要更改的缓冲池的名称。数据类型为 varchar。

buffer_pool_size

缓冲池的大小，以页数表示。数据类型为 integer。

以下参数可选：

immediate

指定命令是否立即运行。数据类型为 char。默认为 Y。

automatic

指定是否将缓冲池设置为自动。数据类型为 char。默认为 N。

change_number_blocks

指定缓冲池中的块页面数是否发生更改。数据类型为 char。默认为 N。

number_block_pages

缓冲池中的块页面数。数据类型为 integer。默认为 0。

block_size

块页面的块大小。数据类型为 integer。有效值：2 到 256。默认为 32。

使用说明

有关检查更改缓冲池的状态的信息，请参阅rdsadmin.get_task_status。

示例

以下示例将名为 TESTDB 的数据库的缓冲池（名为 BP16）更改为非自动，并将大小更改为 10000 页。Db2 会立即运行此命令。

```
db2 "call rdsadmin.alter_bufferpool(
    'TESTDB',
    'BP16',
    10000,
    'Y',
    'N')"
```

rdsadmin.drop_bufferpool

删除缓冲池。

语法

```
db2 "call rdsadmin.drop_bufferpool(
    'database_name',
    'buffer_pool_name'"
```

参数

以下参数为必需参数：

database_name

缓冲池所属的数据库的名称。数据类型为 varchar。

buffer_pool_name

要删除的缓冲池的名称。数据类型为 varchar。

使用说明

有关检查删除缓冲池的状态的信息，请参阅rdsadmin.get_task_status。

示例

以下示例删除名为 TESTDB 的数据库的名为 BP16 的缓冲池。

```
db2 "call rdsadmin.drop_bufferpool(
    'TESTDB',
    'BP16')"
```

用于管理 RDS for Db2 数据库的存储过程

本主题中介绍的内置存储过程可管理 Amazon RDS for Db2 数据库。要运行这些过程，主用户必须先连接到 `rdsadmin` 数据库。

这些存储过程可用于执行各种任务。此列表并不详尽。

- 常见数据库任务
- 使用 EBCDIC 排序规则创建数据库
- 收集有关数据库的信息
- 修改数据库配置参数
- 从 Linux 迁移到 Linux
- 从 Linux 迁移到 Linux，停机时间接近零

请参阅以下内置存储过程，了解其语法、参数、用法说明和示例。

存储过程

- rdsadmin.create_database
- rdsadmin.deactivate_database
- rdsadmin.activate_database
- rdsadmin.reactivate_database
- rdsadmin.drop_database
- rdsadmin.update_db_param
- rdsadmin.set_configuration
- rdsadmin.show_configuration
- rdsadmin.restore_database
- rdsadmin.rollforward_database
- rdsadmin.complete_rollforward
- rdsadmin.db2pd_command
- rdsadmin.force_application
- rdsadmin.set_archive_log_retention
- rdsadmin.show_archive_log_retention

- rdsadmin.list_archive_log_information

rdsadmin.create_database

创建一个数据库。

语法

```
db2 "call rdsadmin.create_database('database_name')"
```

参数

> **Note**
>
> 此存储过程不验证所需参数的组合。当您调用 rdsadmin.get_task_status 时，由于 `database_codeset`、`database_territory` 和 `database_collation` 的组合无效，用户定义函数可能会返回错误。有关更多信息，请参阅 IBM Db2 文档中的 Choosing the code page, territory, and collation for your database。

以下参数是必需参数：

`database_name`

要创建的数据库的名称。数据类型为 `varchar`。

以下参数可选：

`database_page_size`

数据库的默认页面大小。有效值：`4096`、`8192`、`16384`、`32768`。数据类型为 `integer`。默认值为 `8192`。

> **Important**
>
> Amazon RDS 支持 4KiB、8KiB 和 16KiB 页面的写入原子性。相比之下，32KiB 页面可能面临写入数据受到损坏 或部分数据写入桌面的风险。如果使用 32KiB 页面，我们建议启用时间点故障恢复和自动备份。否则，您将面临无法从损坏的页面中恢复的风险。有关更多信息，请参阅the section called “备份简介”和the section called “时间点故障恢复”。

database_code_set

数据库的代码集。数据类型为 varchar。默认值为 UTF-8。

database_territory

数据库的两个字母的国家/地区代码。数据类型为 varchar。默认值为 US。

database_collation

排序规则序列，用于确定如何对存储在数据库中的字符串进行排序和比较。数据类型为 varchar。

有效值：

- COMPATIBILITY – IBM Db2 版本 2 排序规则序列。
- EBCDIC_819_037 – ISO 拉丁语代码页，排序规则；CCSID 037（EBCDIC 美国英语）。
- EBCDIC_819_500 – ISO 拉丁语代码页，排序规则；CCSID 500（EBCDIC 国际）。
- EBCDIC_850_037 – ASCII 拉丁语代码页，排序规则；CCSID 037（EBCDIC 美国英语）。
- EBCDIC_850_500 – ASCII 拉丁语代码页，排序规则；CCSID 500（EBCDIC 国际）。
- EBCDIC_932_5026 – ASCII 日语代码页，排序规则；CCSID 037（EBCDIC 美国英语）。
- EBCDIC_932_5035 – ASCII 日语代码页，排序规则；CCSID 500（EBCDIC 国际）。
- EBCDIC_1252_037 – Windows 拉丁语代码页，排序规则；CCSID 037（EBCDIC 美国英语）。
- EBCDIC_1252_500 – Windows 拉丁语代码页，排序规则；CCSID 500（EBCDIC 国际）。
- IDENTITY – 默认排序规则。对字符串进行逐字节比较。
- IDENTITY_16BIT – UTF-16 的兼容性编码方案：8 位（CESU-8）排序规则序列。有关更多信息，请参阅 Unicode Consortium 网站上的 [Unicode Technical Report #26]。
- NLSCHAR – 仅适用于泰语代码页（CP874）。
- SYSTEM – 如果您使用 SYSTEM，则数据库会自动将排序规则序列用于 database_codeset 和 database_territory。

默认值为 IDENTITY。

此外，RDS for Db2 还支持以下排序规则组：language-aware-collation 和 locale-sensitive-collation。有关更多信息，请参阅 IBM Db2 文档中的 [Choosing a collation for a Unicode database]。

database_autoconfigure_str

AUTOCONFIGURE 命令语法，例如 'AUTOCONFIGURE APPLY DB'。数据类型为 varchar。默认值是空字符串或 Null。

有关更多信息，请参阅 IBM Db2 文档中的 AUTOCONFIGURE 命令。

使用说明

如果您打算修改 db2_compatibility_vector 参数，请在创建数据库之前进行修改。有关更多信息，请参阅 设置 db2_compatibility_vector 参数。

特殊注意事项：

- 发送到 Db2 实例的 CREATE DATABASE 命令使用 RESTRICTIVE 选项。
- RDS for Db2 仅使用 AUTOMATIC STORAGE 表空间。
- RDS for Db2 使用 NUMSEGS 和 DFT_EXTENT_SZ 的默认值。
- RDS for Db2 使用存储加密，但不支持数据库加密。

有关这些注意事项的更多信息，请参阅 IBM Db2 文档中的 CREATE DATABASE command。

在调用 rdsadmin.create_database 之前，必须连接到 rdsadmin 数据库。在以下示例中，将 *master_username* 和 *master_password* 替换为 RDS for Db2 数据库实例信息：

```
db2 connect to rdsadmin user master_username using master_password
```

有关检查创建数据库的状态的信息，请参阅rdsadmin.get_task_status。

要查看调用 rdsadmin.create_database 时返回的错误消息，请参阅the section called “存储过程错误”。

示例

以下示例创建一个名为 TESTJP 的数据库，该数据库具有适用于日本的 *database_code_set*、*database_territory* 和 *database_collation* 参数的正确组合：

```
db2 "call rdsadmin.create_database('TESTJP', 4096, 'IBM-437', 'JP', 'SYSTEM')"
```

rdsadmin.deactivate_database

停用数据库。

语法

```
db2 "call rdsadmin.deactivate_database(
```

```
    ?,
    'database_name')"
```

参数

以下输出参数是必需参数：

?

输出错误消息的参数标记。此参数仅接受 ?。

以下输入参数是必需参数：

database_name

要停用的数据库的名称。数据类型为 varchar。

使用说明

您可以停用数据库以节省内存资源或进行其它数据库配置更改。要使已停用的数据库恢复联机，请调用 rdsadmin.activate_database 存储过程。

有关如何检查数据库停用状态的信息，请参阅rdsadmin.get_task_status。

要查看调用 rdsadmin.deactivate_database 时返回的错误消息，请参阅the section called “存储过程错误”。

示例

以下示例停用名为 TESTDB 的数据库。

```
db2 "call rdsadmin.deactivate_database(?, 'TESTDB')"
```

rdsadmin.activate_database

激活数据库。

有关 rdsadmin.reactivate_database 与 rdsadmin.activate_database 的差异的更多信息，请参阅 使用说明。

语法

```
db2 "call rdsadmin.activate_database(
    ?,
    'database_name')"
```

参数

以下输出参数是必需参数：

?

输出错误消息的参数标记。此参数仅接受 ?。

以下输入参数是必需参数：

database_name

要激活的数据库的名称。数据类型为 varchar。

使用说明

默认情况下，所有数据库在创建后都处于激活状态。如果停用独立数据库实例上的数据库以节省内存资源或进行其它数据库配置更改，请调用 rdsadmin.activate_database 存储过程以再次激活数据库。

有关如何检查数据库激活状态的信息，请参阅rdsadmin.get_task_status。

要查看调用 rdsadmin.activate_database 时返回的错误消息，请参阅the section called “存储过程错误”。

示例

以下示例激活名为 TESTDB 的数据库。

```
db2 "call rdsadmin.activate_database(?, 'TESTDB')"
```

rdsadmin.reactivate_database

重新激活数据库。

有关 rdsadmin.activate_database 与 `rdsadmin.reactivate_database` 之间差异的更多信息，请参阅使用说明。

语法

```
db2 "call rdsadmin.reactivate_database(
    ?,
    'database_name')"
```

参数

以下输出参数是必需参数：

?

输出错误消息的参数标记。此参数仅接受 `?`。

以下输入参数是必需参数：

database_name

要重新激活的数据库的名称。数据类型为 `varchar`。

使用说明

调用 `rdsadmin.reactivate_database` 存储过程时，该存储过程首先通过调用 rdsadmin.deactivate_database 存储过程来停用数据库，然后通过调用 rdsadmin.activate_database 存储过程来激活数据库。

更改数据库配置后，您可能需要在 RDS for Db2 数据库实例上重新激活数据库。要确定是否需要重新激活数据库，请连接到该数据库并运行 `db2 get db cfg show detail`。

对于独立数据库实例上的数据库，您可以使用 `rdsadmin.reactivate_database` 存储过程来重新激活该数据库。或者，如果您已经调用了 rdsadmin.deactivate_database 存储过程，则可以改为调用 rdsadmin.activate_database 存储过程。

有关检查重新激活数据库的状态的信息，请参阅 rdsadmin.get_task_status。

要查看调用 `rdsadmin.reactivate_database` 时返回的错误消息，请参阅the section called “存储过程错误”。

示例

以下示例重新激活名为 TESTDB 的数据库。

```
db2 "call rdsadmin.reactivate_database(?, 'TESTDB')"
```

rdsadmin.drop_database

删除数据库。

语法

```
db2 "call rdsadmin.drop_database('database_name')"
```

参数

以下参数是必需参数：

database_name

要删除的数据库的名称。数据类型为 varchar。

使用说明

仅当满足以下条件时，才可以通过调用 rdsadmin.drop_database 来删除数据库：

- 您在使用 Amazon RDS 控制台或 Amazon CLI 创建 RDS for Db2 数据库实例时未指定数据库的名称。有关更多信息，请参阅 创建数据库实例。
- 您通过调用 the section called “rdsadmin.create_database” 存储过程创建了数据库。
- 您通过调用 the section called “rdsadmin.restore_database” 存储过程从脱机或备份的映像中还原了数据库。

在调用 rdsadmin.drop_database 之前，必须连接到 rdsadmin 数据库。在以下示例中，将 *master_username* 和 *master_password* 替换为 RDS for Db2 数据库实例信息：

```
db2 connect to rdsadmin user master_username using master_password
```

有关如何检查数据库删除状态的信息，请参阅rdsadmin.get_task_status。

要查看调用 `rdsadmin.drop_database` 时返回的错误消息，请参阅the section called “存储过程错误”。

示例

以下示例删除名为 `TESTDB` 的数据库：

```
db2 "call rdsadmin.drop_database('TESTDB')"
```

rdsadmin.update_db_param

更新数据库参数。

语法

```
db2 "call rdsadmin.update_db_param(
    'database_name',
    'parameter_to_modify',
    'changed_value)"
```

参数

以下参数为必需参数：

database_name

要运行其任务的数据库的名称。数据类型为 `varchar`。

parameter_to_modify

要修改的参数的名称。数据类型为 `varchar`。有关更多信息，请参阅 Amazon RDS for Db2 参数。

changed_value

要将参数值更改为的值。数据类型为 `varchar`。

使用说明

有关如何检查数据库参数更新状态的信息，请参阅rdsadmin.get_task_status。

要查看调用 `rdsadmin.update_db_param` 时返回的错误消息，请参阅the section called “存储过程错误”。

示例

示例 1：更新参数

以下示例将名为 TESTDB 的数据库的 archretrydelay 参数更新为 100：

```
db2 "call rdsadmin.update_db_param(
    'TESTDB',
    'archretrydelay',
    '100')"
```

示例 2：延迟对象验证

以下示例推迟了对在名为 TESTDB 的数据库上创建的对象的验证，以避免依赖性检查：

```
db2 "call rdsadmin.update_db_param(
    'TESTDB',
    'auto_reval',
    'deferred_force')"
```

rdsadmin.set_configuration

配置数据库的特定设置。

语法

```
db2 "call rdsadmin.set_configuration(
    'name',
    'value')"
```

参数

以下参数为必需参数：

name

配置设置的名称。数据类型为 varchar。

#

配置设置的值。数据类型为 varchar。

使用说明

下表显示了您可以使用 `rdsadmin.set_configuration` 控制的配置设置。

名称	描述
RESTORE_DATABASE_NUM_BUFFERS	还原操作期间要创建的缓冲区数量。此值必须小于数据库实例类的总内存大小。如果未配置此设置，则 Db2 会确定在还原操作期间要使用的值。有关更多信息，请参阅 IBM Db2 文档。
RESTORE_DATABASE_PARALLELISM	在还原操作期间要创建的缓冲区操纵器的数量。此值必须小于数据库实例的 vCPU 数量的两倍。如果未配置此设置，则 Db2 会确定在还原操作期间要使用的值。有关更多信息，请参阅 IBM Db2 文档。

示例

示例 1：指定要创建的缓冲区操纵符的数量

以下示例将 RESTORE_DATABASE_PARALLELISM 配置设置为 8。

```
db2 "call rdsadmin.set_configuration(
    'RESTORE_DATABASE_PARALLELISM',
    '8')"
```

示例 2：指定要创建的缓冲区数量

以下示例将 RESTORE_DATABASE_NUM_BUFFERS 配置设置为 150。

```
db2 "call rdsadmin.set_configuration(
    'RESTORE_DATABASE_NUM_BUFFERS',
    '150')"
```

rdsadmin.show_configuration

返回可使用存储过程 `rdsadmin.set_configuration` 进行设置的当前设置。

语法

```
db2 "call rdsadmin.show_configuration(
```

```
    'name')"
```

参数

以下参数为可选参数：

name

要返回其相关信息的配置设置的名称。数据类型为 varchar。

以下配置名称有效：

- RESTORE_DATABASE_NUM_BUFFERS – 在还原操作期间要创建的缓冲区数量。
- RESTORE_DATABASE_PARALLELISM – 在还原操作期间要创建的缓冲区操纵器的数量。

使用说明

如果您未指定配置设置的名称，则 rdsadmin.show_configuration 返回您可以使用存储过程 rdsadmin.set_configuration 进行设置的所有配置设置的信息。

示例

以下示例返回有关当前 RESTORE_DATABASE_PARALLELISM 配置的信息。

```
db2 "call rdsadmin.show_configuration(
    'RESTORE_DATABASE_PARALLELISM')"
```

rdsadmin.restore_database

从 Amazon S3 存储桶还原到 RDS for Db2 数据库实例。

语法

```
db2 "call rdsadmin.restore_database(
    ?,
    'database_name',
    's3_bucket_name',
    's3_prefix',
    restore_timestamp,
    'backup_type')"
```

参数

以下输出参数是必需参数：

?

输出错误消息的参数标记。此参数仅接受 ?。

以下输入参数是必需参数：

database_name

要在 RDS for Db2 中恢复的目标数据库的名称。数据类型为 `varchar`。

例如，如果源数据库名称为 TESTDB，而您将 *database_name* 设置为 NEWDB，则 Amazon RDS 将恢复 NEWDB 作为源数据库。

s3_bucket_name

备份所在的 Amazon S3 存储桶的名称。数据类型为 `varchar`。

s3_prefix

下载期间用于文件匹配的前缀。数据类型为 `varchar`。

如果此参数为空，则将下载 Amazon S3 存储桶中的所有文件。以下是示例前缀：

```
backupfolder/SAMPLE.0.rdsdb.DBPART000.20230615010101
```

restore_timestamp

数据库备份映像的时间戳。数据类型为 `varchar`。

时间戳包含在备份文件名中。例如，20230615010101 是文件名 SAMPLE.0.rdsdb.DBPART000.20230615010101.001 的时间戳。

backup_type

备份的类型。数据类型为 `varchar`。有效值：OFFLINE、ONLINE。

使用 ONLINE 进行停机时间接近零的迁移。有关更多信息，请参阅 针对 Amazon RDS for Db2 从 Linux 迁移到 Linux，停机时间接近零。

使用说明

您可以使用此存储过程将 Db2 数据库迁移到 RDS for Db2 数据库实例。有关更多信息，请参阅 使用 Amazon 服务将数据从 Db2 迁移到 Amazon RDS for Db2。您也可以使用此存储过程创建同一数据库的多个副本但使用不同的数据库名称，这些副本使用相同的还原映像。

在调用存储过程之前，请查看以下注意事项：

- 在还原数据库之前，您必须为 RDS for Db2 数据库实例预调配存储空间，其大小等于或大于磁盘上备份和原始 Db2 数据库的大小之和。有关更多信息，请参阅 Insufficient disk space。
- 还原备份时，Amazon RDS 会在您的 RDS for Db2 数据库实例上提取备份文件。每个备份文件不得超过 5TB。如果备份文件超过 5 TB，则必须将备份文件拆分为较小的文件。
- 要使用 `rdsadmin.restore_database` 存储过程还原所有文件，请勿在文件名中的时间戳后加上文件号后缀。例如，*s3_prefix* `backupfolder/SAMPLE.0.rdsdb.DBPART000.20230615010101` 还原以下文件：

```
SAMPLE.0.rdsdb.DBPART000.20230615010101.001
SAMPLE.0.rdsdb.DBPART000.20230615010101.002
SAMPLE.0.rdsdb.DBPART000.20230615010101.003
SAMPLE.0.rdsdb.DBPART000.20230615010101.004
SAMPLE.0.rdsdb.DBPART000.20230615010101.005
```

- RDS for Db2 不支持非自动存储。有关更多信息，请参阅 Tablespaces not restored。
- RDS for Db2 不支持非防护例程。有关更多信息，请参阅 Non-fenced routines not allowed。
- 要提高数据库还原操作的性能，您可以配置供 RDS 使用的缓冲区和缓冲区操纵器的数量。要检查当前配置，请使用 the section called “rdsadmin.show_configuration”。要更改配置，请使用 the section called “rdsadmin.set_configuration”。

要在还原数据库后使数据库联机并应用其他事务日志，请参阅rdsadmin.rollforward_database。

要了解数据库的还原状态，请参阅rdsadmin.get_task_status。

要查看调用 `rdsadmin.restore_database` 时返回的错误消息，请参阅the section called “存储过程错误”。

示例

以下示例使用带有 *s3_prefix* `backupfolder/SAMPLE.0.rdsdb.DBPART000.20230615010101` 的单个文件或多个文件还原脱机备份：

```
db2 "call rdsadmin.restore_database(
    ?,
    'SAMPLE',
    'amzn-s3-demo-bucket',
    'backupfolder/SAMPLE.0.rdsdb.DBPART000.20230615010101',
    20230615010101,
    'OFFLINE')"
```

rdsadmin.rollforward_database

通过调用 rdsadmin.restore_database 还原数据库后，使数据库联机并应用其他事务日志。

语法

```
db2 "call rdsadmin.rollforward_database(
    ?,
    'database_name',
    's3_bucket_name',
    s3_prefix,
    'rollforward_to_option',
    'complete_rollforward')"
```

参数

以下输出参数是必需参数：

?

输出错误消息的参数标记。此参数仅接受 ?。

以下输入参数是必需参数：

database_name

要对其执行操作的数据库的名称。数据类型为 varchar。

s3_bucket_name

备份所在的 Amazon S3 存储桶的名称。数据类型为 varchar。

s3_prefix

下载期间用于文件匹配的前缀。数据类型为 varchar。

如果此参数为空，则将下载 S3 存储桶中的所有文件。以下示例是示例前缀：

```
backupfolder/SAMPLE.0.rdsdb.DBPART000.20230615010101
```

以下输入参数为可选参数：

rollforward_to_option

您想要向前滚动到的点。数据类型为 varchar。有效值：END_OF_LOGS、END_OF_BACKUP。默认值为 END_OF_LOGS。

complete_rollforward

指定是否完成向前滚动过程。数据类型为 varchar。默认值为 TRUE。

如果为 TRUE，则在完成后，数据库处于联机状态并且可以访问。如果为 FALSE，则数据库将保持 ROLL-FORWARD PENDING 状态。

使用说明

调用 rdsadmin.restore_database 后，必须调用 rollforward_database 以应用来自 S3 存储桶的归档日志。您还可以在调用 rdsadmin.restore_database 后使用此存储过程来还原其他事务日志。

如果将 complete_rollforward 设置为 FALSE，则您的数据库处于 ROLL-FORWARD PENDING 状态并处于脱机状态。要使数据库联机，您必须调用 rdsadmin.complete_rollforward。

有关如何检查数据库向前滚动状态的信息，请参阅rdsadmin.get_task_status。

示例

示例 1：使包含事务日志的数据库联机

以下示例向前滚动到包含事务日志的数据库的联机备份，然后使数据库联机：

```
db2 "call rdsadmin.rollforward_database(
    ?,
    null,
    null,
    'END_OF_LOGS',
    'TRUE')"
```

示例 2：使不包含事务日志的数据库联机

以下示例向前滚动到不包含事务日志的数据库的联机备份，然后使数据库联机：

```
db2 "call rdsadmin.rollforward_database(
    ?,
    'TESTDB',
    'amzn-s3-demo-bucket',
    'logsfolder/,
    'END_OF_BACKUP',
    'TRUE')"
```

示例 3：不让包含事务日志的数据库联机

以下示例向前滚动到包含事务日志的数据库的联机备份，但不会使数据库联机：

```
db2 "call rdsadmin.rollforward_database(
    ?,
    'TESTDB',
    null,
    'onlinebackup/TESTDB',
    'END_OF_LOGS',
    'FALSE')"
```

示例 4：不让包含其他事务日志的数据库联机

以下示例向前滚动到包含其他事务日志的数据库的联机备份，但不会使数据库联机：

```
db2 "call rdsadmin.rollforward_database(
    ?,
    'TESTDB',
    'amzn-s3-demo-bucket',
    'logsfolder/S0000155.LOG',
    'END_OF_LOGS',
    'FALSE')"
```

rdsadmin.complete_rollforward

从 `ROLL-FORWARD PENDING` 状态使数据库联机。

语法

```
db2 "call rdsadmin.complete_rollforward(
```

```
    ?,
    'database_name')"
```

参数

以下输出参数是必需参数：

?

输出错误消息的参数标记。此参数仅接受 ?。

以下输入参数是必需参数：

database_name

要实现联机的数据库的名称。数据类型为 varchar。

使用说明

如果您在 complete_rollforward 设置为 FALSE 的情况下调用了 rdsadmin.rollforward_database，则您的数据库处于 ROLL-FORWARD PENDING 状态并处于脱机状态。要完成向前滚动过程并使数据库联机，请调用 rdsadmin.complete_rollforward。

有关检查完成向前滚动过程的状态的信息，请参阅rdsadmin.get_task_status。

示例

以下示例使 TESTDB 数据库联机：

```
db2 "call rdsadmin.complete_rollforward(
    ?,
    'TESTDB')"
```

rdsadmin.db2pd_command

收集有关 RDS for Db2 数据库的信息。

语法

```
db2 "call rdsadmin.db2pd_command('db2pd_cmd')"
```

参数

以下输入参数是必需参数：

db2pd_cmd

要运行的 db2pd 命令的名称。数据类型为 `varchar`。

参数必须以连字符开头。有关参数列表，请参阅 IBM Db2 文档中的 db2pd - Monitor and troubleshoot Db2 database command。

不支持以下选项：

- `-addnode`
- `-alldatabases`
- `-alldbp`
- `-alldbs`
- `-allmembers`
- `-alm_in_memory`
- `-cfinfo`
- `-cfpool`
- `-command`
- `-dbpartitionnum`
- `-debug`
- `-dump`
- `-everything`
- `-file | -o`
- `-ha`
- `-interactive`
- `-member`
- `-pages`

> **Note**
>
> 支持`-pages summary`。

- `-pdcollection`
- `-repeat`
- `-stack`
- `-totalmem`

不支持 `file` 子选项，例如 `db2pd -db testdb -tcbstats file=tcbstat.out`。

不支持使用 `stacks` 选项，如 `db2pd -edus interval=5 top=10 stacks`。

使用说明

此存储过程收集的信息有助于监控 RDS for Db2 数据库并对其进行故障排除。

该存储过程使用 IBM db2pd 实用程序运行各种命令。db2pd 实用程序需要 `SYSADM` 授权，而 RDS for Db2 主用户却没有此授权。但是，使用 Amazon RDS 存储过程，主用户可以使用该实用程序运行各种命令。有关该实用程序的更多信息，请参阅 IBM Db2 文档中的 db2pd - Monitor and troubleshoot Db2 database command。

输出限制为最大 2 GB。

有关检查收集关于数据库的信息的状态的信息，请参阅 rdsadmin.get_task_status。

示例

示例 1：返回数据库实例的正常运行时间

以下示例返回 RDS for Db2 数据库实例的正常运行时间：

```
db2 "call rdsadmin.db2pd_command('-')"
```

示例 2：返回数据库的正常运行时间

以下示例返回名为 `TESTDB` 的数据库的正常运行时间：

```
db2 "call rdsadmin.db2pd_command('-db TESTDB -')"
```

示例 3：返回数据库实例的内存使用情况

以下示例返回 RDS for Db2 数据库实例的内存使用情况：

```
db2 "call rdsadmin.db2pd_command('-dbptnmem')"
```

示例 4：返回数据库实例和数据库的内存集

以下示例返回 RDS for Db2 数据库实例和名为 TESTDB 的数据库的内存集：

```
db2 "call rdsadmin.db2pd_command('-inst -db TESTDB -memsets')"
```

rdsadmin.force_application

强制应用程序退出 RDS for Db2 数据库。

语法

```
db2 "call rdsadmin.force_application(
    ?,
    'applications')"
```

参数

以下输出参数是必需参数：

?

输出错误消息的参数标记。此参数仅接受 ?。

以下输入参数是必需参数：

applications

您要强制退出 RDS for Db2 数据库的应用程序。数据类型为 varchar。有效值：ALL 或 *application_handle*。

用逗号分隔多个应用程序的名称。示例：“*application_handle_1*, *application_handle_2*”。

使用说明

此存储过程会强制所有应用程序退出数据库，以便您可以执行维护。

此存储过程使用 IBM FORCE APPLICATION 命令。FORCE APPLICATION 命令需要 SYSADM、SYSMAINT 或 SYSCTRL 授权，而 RDS for Db2 主用户却没有此类授权。但是，使用

Amazon RDS 存储过程，主用户可以使用该命令。有关更多信息，请参阅 IBM Db2 文档中的 FORCE APPLICATION command。

有关检查强制应用程序退出数据库的状态的信息，请参阅 rdsadmin.get_task_status。

示例

示例 1：指定所有应用程序

以下示例强制所有应用程序退出 RDS for Db2 数据库：

```
db2 "call rdsadmin.force_application(
    ?,
    'ALL')"
```

示例 2：指定多个应用程序

以下示例强制应用程序句柄 9991、8891 和 1192 退出 RDS for Db2 数据库：

```
db2 "call rdsadmin.force_application(
    ?,
    '9991, 8891, 1192')"
```

rdsadmin.set_archive_log_retention

配置为指定的 RDS for Db2 数据库保留归档日志文件的时间量（以小时为单位）。

语法

```
db2 "call rdsadmin.set_archive_log_retention(
    ?,
    'database_name',
    'archive_log_retention_hours')"
```

参数

以下输出参数是必需参数：

?

输出错误消息的参数标记。此参数仅接受 ?。

以下输入参数是必需参数：

database_name

要配置其归档日志保留期的数据库的名称。数据类型为 varchar。

archive_log_retention_hours

要保留归档日志文件的小时数。数据类型为 smallint。默认值为 0，最大值为 168（7 天）。

如果值为 0，则 Amazon RDS 不会保留归档日志文件。

使用说明

默认情况下，RDS for Db2 会将日志保留 5 分钟。如果您使用诸如 Amazon DMS [用于进行更改数据捕获（CDC）] 或 IBM Q Replication 之类的复制工具，建议您在这些工具中将日志保留期设置为 5 分钟以上。

您可以通过调用 the section called “rdsadmin.show_archive_log_retention” 来查看当前的归档日志保留期设置。

您无法在 rdsadmin 数据库上配置归档日志保留期设置。

示例

示例 1：设置保留时间

以下示例将名为 TESTDB 的数据库的归档日志保留时间设置为 24 小时。

```
db2 "call rdsadmin.set_archive_log_retention(
    ?,
    'TESTDB',
    '24')"
```

示例 2：禁用保留时间

以下示例禁用名为 TESTDB 的数据库的归档日志保留期。

```
db2 "call rdsadmin.set_archive_log_retention(
    ?,
    'TESTDB',
    '0')"
```

rdsadmin.show_archive_log_retention

返回指定数据库的当前归档日志保留期设置。

语法

```
db2 "call rdsadmin.show_archive_log_retention(
    ?,
    'database_name')"
```

参数

以下输出参数是必需参数：

?

输出错误消息的参数标记。此参数仅接受 ?。

以下输入参数是必需参数：

database_name

要显示其归档日志保留期设置的数据库的名称。数据类型为 varchar。

示例

以下示例显示名为 TESTDB 的数据库的归档日志保留期设置。

```
db2 "call rdsadmin.show_archive_log_retention(
    ?
    'TESTDB')"
```

rdsadmin.list_archive_log_information

返回有关归档日志文件的详细信息，例如指定数据库的各个日志文件的大小、创建日期和时间以及名称。它还返回数据库中日志文件使用的总存储量。

语法

```
db2 "call rdsadmin.list_archive_log_information(
```

```
    ?,
    'database_name')"
```

参数

以下输出参数是必需参数：

?

输出错误消息的参数标记。此参数仅接受 ?。

以下输入参数是必需参数：

database_name

要列出其归档日志信息的数据库的名称。数据类型为 varchar。

示例

以下示例返回名为 TESTDB 的数据库的归档日志信息。

```
db2 "call rdsadmin.list_archive_log_information(
    ?
    'TESTDB')"
```

用于 RDS for Db2 的存储访问的存储过程

本主题中介绍的内置存储过程可管理对使用 Amazon S3 来迁移数据的 RDS for Db2 数据库的存储访问权限。有关更多信息，请参阅 the section called “使用 Amazon S3 进行迁移”。

请参阅以下内置存储过程，了解其语法、参数、用法说明和示例。

存储过程

- rdsadmin.catalog_storage_access
- rdsadmin.uncatalog_storage_access

rdsadmin.catalog_storage_access

对存储别名进行编目，用于访问包含 Db2 数据文件的 Amazon S3 存储桶。

语法

```
db2 "call rdsadmin.catalog_storage_access(
    ?,
    'alias',
    's3_bucket_name',
    'grantee_type',
    'grantee'
    )"
```

参数

以下输出参数是必需参数：

?

输出错误消息的参数标记。数据类型为 `varchar`。

以下输入参数是必需参数：

alias

用于访问 Amazon S3 存储桶中的远程存储的别名。数据类型为 `varchar`。

s3_bucket_name

数据所在的 Amazon S3 存储桶的名称。数据类型为 `varchar`。

grantee_type

要接收授权的被授权者的类型。数据类型为 varchar。有效值：USER、GROUP。

grantee

要接收授权的用户或组。数据类型为 varchar。

使用说明

Amazon RDS 在 IAM 角色中包含您添加到 RDS for Db2 数据库实例的已编目别名。如果您从数据库实例中移除 IAM 角色，则 Amazon RDS 会删除该别名。有关更多信息，请参阅 the section called “使用 Amazon S3 进行迁移”。

有关检查对别名编目的状态的信息，请参阅 rdsadmin.get_task_status。

示例

以下示例注册一个名为 SAMPLE 的别名。用户 jorge_souza 被授予访问名为 amzn-s3-demo-bucket 的 Amazon S3 存储桶的权限。

```
db2 "call rdsadmin.catalog_storage_access(
    ?,
    'SAMPLE',
    'amzn-s3-demo-bucket',
    'USER',
    'jorge_souza')"
```

rdsadmin.uncatalog_storage_access

移除存储访问权限别名。

语法

```
db2 "call rdsadmin.uncatalog_storage_access(
    ?,
    'alias')"
```

参数

以下输出参数是必需参数：

?

输出错误消息的参数标记。数据类型为 varchar。

以下输入参数是必需参数：

alias

要删除的存储别名的名称。数据类型为 varchar。

使用说明

有关检查移除别名的状态的信息，请参阅 rdsadmin.get_task_status。

示例

以下示例移除一个名为 SAMPLE 的别名。此别名不再提供对与其关联的 Amazon S3 存储桶的访问权限。

```
db2 "call rdsadmin.uncatalog_storage_access(
    ?,
    'SAMPLE')"
```

用于 RDS for Db2 的表空间的存储过程

本主题中介绍的内置存储过程可管理 Amazon RDS for Db2 数据库的表空间。要运行这些过程，主用户必须先连接到 `rdsadmin` 数据库。

这些存储过程可用于执行各种任务。此列表并不详尽。

- 表空间的常见任务
- 生成性能报告
- 使用 db2look 复制数据库元数据
- 为 IBM Db2 Data Management Console 创建存储库数据库

请参阅以下内置存储过程，了解其语法、参数、用法说明和示例。

存储过程

- rdsadmin.create_tablespace
- rdsadmin.alter_tablespace
- rdsadmin.rename_tablespace
- rdsadmin.drop_tablespace

rdsadmin.create_tablespace

创建表空间。

语法

```
db2 "call rdsadmin.create_tablespace(
    'database_name',
    'tablespace_name',
    'buffer_pool_name',
    tablespace_page_size,
    tablespace_initial_size,
    tablespace_increase_size,
    'tablespace_type',
    'tablespace_prefetch_size')"
```

参数

以下参数为必需参数：

database_name

要在其中创建表空间的数据库的名称。数据类型为 varchar。

tablespace_name

要创建的表空间的名称。数据类型为 varchar。

表空间名称具有以下限制：

- 不能与此数据库中现有表空间的名称相同。
- 只能包含字符 _$#@a-zA-Z0-9。
- 不能以 _ 或 $ 开头。
- 不能以 SYS 开头。

以下参数可选：

buffer_pool_name

要为其分配表空间的缓冲池的名称。数据类型为 varchar。默认值是空字符串。

> **Important**
> 您必须已经具有相同页面大小的缓冲池才能与表空间相关联。

tablespace_page_size

表空间的页面大小（以字节为单位）。数据类型为 integer。有效值：4096、8192、16384、32768。默认值是通过调用 rdsadmin.create_database 创建数据库时使用的页面大小。

> **Important**
> Amazon RDS 支持 4KiB、8KiB 和 16KiB 页面的写入原子性。相比之下，32KiB 页面可能面临写入数据受到损坏 或部分数据写入桌面的风险。如果使用 32KiB 页面，我们建议启用

时间点故障恢复和自动备份。否则，您将面临无法从损坏的页面中恢复的风险。有关更多信息，请参阅the section called "备份简介"和the section called "时间点故障恢复"。

tablespace_initial_size

表空间的初始大小，以千字节（KB）为单位。数据类型为 `integer`。有效值：48 或更高。默认值为 Null。

如果您没有设置值，则 Db2 为您设置适当的值。

> **Note**
>
> 此参数不适用于临时表空间，因为系统管理临时表空间。

tablespace_increase_size

表空间已满时增加表空间的百分比。数据类型为 `integer`。有效值：`1`–`100`。默认值为 Null。

如果您没有设置值，则 Db2 为您设置适当的值。

> **Note**
>
> 此参数不适用于临时表空间，因为系统管理临时表空间。

tablespace_type

表空间的类型。数据类型为 `char`。有效值：U（对于用户数据）、T（对于用户临时数据）或 S（对于系统临时数据）。默认值为 U。

tablespace_prefetch_size

表空间的预提取页面大小。数据类型为 `char`。有效值：`AUTOMATIC`（不区分大小写），或小于或等于 32767 的非零正整数。

使用说明

RDS for Db2 始终为数据创建大型数据库。

有关检查创建表空间的状态的信息，请参阅rdsadmin.get_task_status。

示例

示例 1：创建表空间并分配缓冲池

以下示例创建一个名为 SP8 的表空间，并为名为 TESTDB 的数据库分配一个名为 BP8 的缓冲池。表空间的初始表空间页面大小为 4096 字节，初始表空间为 1000KB，表大小增加设置为 50%。

```
db2 "call rdsadmin.create_tablespace(
    'TESTDB',
    'SP8',
    'BP8',
    4096,
    1000,
    50)"
```

示例 2：创建临时表空间并分配缓冲池

以下示例创建一个名为 SP8 的临时表空间。它为名为 TESTDB 的数据库分配一个大小为 8 KiB 的名为 BP8 的缓冲池。

```
db2 "call rdsadmin.create_tablespace(
    'TESTDB',
    'SP8',
    'BP8',
    8192,
    NULL,
    NULL,
    'T')"
```

示例 3：创建表空间并分配预提取页面大小

以下示例为名为 TESTDB 的数据库创建名为 SP8 的表空间。表空间的初始表空间增加大小为 50，预提取页面大小为 800。

```
db2 "call rdsadmin.create_tablespace(
    'TESTDB',
    'SP8',
    NULL,
    NULL,
    NULL,
```

```
    NULL,
    50,
    NULL,
    '800')"
```

rdsadmin.alter_tablespace

更改表空间。

语法

```
db2 "call rdsadmin.alter_tablespace(
    'database_name',
    'tablespace_name',
    'buffer_pool_name',
    tablespace_increase_size,
    'max_size',
    'reduce_max',
    'reduce_stop',
    'reduce_value',
    'lower_high_water',
    'lower_high_water_stop',
    'switch_online',
    'tablespace_prefetch_size')"
```

参数

以下参数为必需参数：

database_name

使用表空间的数据库的名称。数据类型为 varchar。

tablespace_name

要更改的表空间的名称。数据类型为 varchar。

以下参数可选：

buffer_pool_name

要为其分配表空间的缓冲池的名称。数据类型为 varchar。默认值是空字符串。

> **Important**
> 您必须已经具有相同页面大小的缓冲池才能与表空间相关联。

tablespace_increase_size

表空间已满时增加表空间的百分比。数据类型为 integer。有效值：1 – 100。默认值为 0。

max_size

表空间的最大大小。数据类型为 varchar。有效值：*integer* K | M | G 或 NONE。默认值为 NONE。

reduce_max

指定是否将高水位降到其最大限制。数据类型为 char。默认值为 N。

reduce_stop

指定是中断前一个 reduce_max 命令还是 reduce_value 命令。数据类型为 char。默认值为 N。

reduce_value

减少表空间高水位所依据的数字或百分比。数据类型为 varchar。有效值：*integer* K| M | G 或 1–100。默认值为 N。

lower_high_water

指定是否运行 ALTER TABLESPACE LOWER HIGH WATER MARK 命令。数据类型为 char。默认值为 N。

lower_high_water_stop

指定是否运行 ALTER TABLESPACE LOWER HIGH WATER MARK STOP 命令。数据类型为 char。默认值为 N。

switch_online

指定是否运行 ALTER TABLESPACE SWITCH ONLINE 命令。数据类型为 char。默认值为 N。

tablespace_prefetch_size

表空间的预提取页面大小。数据类型为 char。有效值：AUTOMATIC（不区分大小写），或小于或等于 32767 的非零正整数。

> **Note**
>
> 此参数仅适用于 buffer_pool_name、table_increase_size、max_size 和 switch_online。参数不适用于 reduce_max、reduce_stop、reduce_value、lower_high_water 和 lower_high_water_stop。

使用说明

在调用存储过程之前，请查看以下注意事项：

- 对于用户临时数据，如果将 tablespace_type 设置为 T，则 rdsadmin.alter_tablespace 存储过程将无法在表空间上运行。
- 可选参数 reduce_max、reduce_stop、reduce_value、lower_high_water、lower_high_water_stop 和 switch_online 是互斥的。不能将它们与 rdsadmin.alter_tablespace 命令中的任何其他可选参数（如 buffer_pool_name）组合使用。有关更多信息，请参阅 Statement not valid。

有关检查更改表空间的状态的信息，请参阅rdsadmin.get_task_status。

有关调用存储过程时返回的错误消息，请参阅the section called "存储过程错误"。

示例

示例 1：降低高水位线

以下示例更改名为 SP8 的表空间，并为名为 TESTDB 的数据库分配一个名为 BP8 的缓冲池以降低高水位线。

```
db2 "call rdsadmin.alter_tablespace(
    'TESTDB',
    'SP8',
    'BP8',
    NULL,
    NULL,
    'Y')"
```

示例 2：降低高水位线

以下示例在数据库 TESTDB 中名为 TBSP_TEST 的表空间上运行 REDUCE MAX 命令。

```
db2 "call rdsadmin.alter_tablespace(
    'TESTDB',
    'TBSP_TEST',
    NULL,
    NULL,
    NULL,
    'Y')"
```

示例 3：中断命令以降低高水位线

以下示例在数据库 TESTDB 中名为 TBSP_TEST 的表空间上运行 REDUCE STOP 命令。

```
db2 "call rdsadmin.alter_tablespace(
    'TESTDB',
    'TBSP_TEST',
    NULL,
    NULL,
    NULL,
    NULL,
    'Y')"
```

示例 4：更改现有的预提取页面大小

以下示例在名为 TSBP_TEST 的表空间上运行 ALTER TABLESPACE SWITCH ONLINE 命令，并将现有的预提取页面大小更改为 64。

```
db2 "call rdsadmin.alter_tablespace(
    'TESTDB',
    'TBSP_TEST',
    NULL,
    NULL,
    NULL,
    NULL,
    NULL,
    NULL,
    NULL,
    NULL,
    ‘Y’,
    ‘64’)"
```

rdsadmin.rename_tablespace

重命名表空间。

语法

```
db2 "call rdsadmin.rename_tablespace(
    ?,
    'database_name',
    'source_tablespace_name',
    'target_tablespace_name')"
```

参数

以下参数为必需参数：

?

输出错误消息的参数标记。此参数仅接受 ?。

database_name

表空间所属的数据库的名称。数据类型为 varchar。

source_tablespace_name

要重命名的表空间的名称。数据类型为 varchar。

target_tablespace_name

表空间的名称。数据类型为 varchar。

新名称具有以下限制：

- 不能与现有表空间的名称相同。
- 只能包含字符 _$#@a-zA-Z0-9。
- 不能以 _ 或 $ 开头。
- 不能以 SYS 开头。

使用说明

有关检查重命名表空间的状态的信息，请参阅rdsadmin.get_task_status。

您无法重命名属于 rdsadmin 数据库的表空间。

示例

以下示例将名为 TESTDB 的数据库中名为 SP8 的表空间重命名为 SP9。

```
db2 "call rdsadmin.rename_tablespace(
    ?,
    'TESTDB',
    'SP8',
    'SP9')"
```

rdsadmin.drop_tablespace

删除表空间。

语法

```
db2 "call rdsadmin.drop_tablespace(
    'database_name',
    'tablespace_name')"
```

参数

以下参数为必需参数：

database_name

表空间所属的数据库的名称。数据类型为 varchar。

tablespace_name

要删除的表空间的名称。数据类型为 varchar。

使用说明

有关检查删除表空间的状态的信息，请参阅rdsadmin.get_task_status。

示例

以下示例从名为 TESTDB 的数据库中删除名为 SP8 的表空间。

```
db2 "call rdsadmin.drop_tablespace(
```

```
    'TESTDB',
    'SP8')"
```

Amazon RDS for Db2 用户定义的函数参考

以下用户定义的函数适用于运行 Db2 引擎的 Amazon RDS 数据库实例。

主题

- rdsadmin.get_task_status
- rdsadmin.list_databases

rdsadmin.get_task_status

返回任务的状态。

语法

```
db2 "select task_id, task_type, database_name, lifecycle,
    varchar(bson_to_json(task_input_params), 500) as task_params,
    cast(task_output as varchar(500)) as task_output
    from table(rdsadmin.get_task_status(task_id,'database_name','task_type'))"
```

参数

以下参数为可选参数。如果您未提供任何参数，则用户定义函数将返回所有数据库的所有任务的状态。Amazon RDS 将任务历史记录保留 35 天。

task_id

正在运行的任务的 ID。运行任务时会返回此 ID。默认值：`0`。

database_name

正在为其运行任务的数据库的名称。

task_type

要查询的任务的类型。有效值：`ADD_GROUPS`、`ADD_USER`、`ALTER_BUFFERPOOL`、`ALTER_TABLESPACE`、`CHANGE_PASSWORD`、C

使用说明

您可以使用 `rdsadmin.get_task_status` 用户定义的函数检查针对 Amazon RDS for Db2 的以下任务的状态。此列表并不详尽。

- 创建、更改或删除缓冲池
- 创建、更改或删除表空间
- 创建或删除数据库
- 从 Amazon S3 还原数据库备份
- 从 Amazon S3 向前滚动数据库日志

示例

以下示例显示在调用 `rdsadmin.get_task_status` 时返回的列。

```
db2 "describe select * from table(rdsadmin.get_task_status())"
```

以下示例列出所有任务的状态。

```
db2 "select task_id, task_type, database_name, lifecycle,
    varchar(bson_to_json(task_input_params), 500) as task_params,
    cast(task_output as varchar(500)) as task_output
    from table(rdsadmin.get_task_status(null,null,null))"
```

以下示例列出特定任务的状态。

```
db2 "select task_id, task_type, database_name,
    varchar(bson_to_json(task_input_params), 500) as task_params
    from table(rdsadmin.get_task_status(1,null,null))"
```

以下示例列出特定任务和数据库的状态。

```
db2 "select task_id, task_type, database_name,
    varchar(bson_to_json(task_input_params), 500) as task_params
    from table(rdsadmin.get_task_status(2,'SAMPLE',null))"
```

以下示例列出所有 `ADD_GROUPS` 任务的状态。

```
db2 "select task_id, task_type, database_name,
    varchar(bson_to_json(task_input_params), 500) as task_params
    from table(rdsadmin.get_task_status(null,null,'add_groups'))"
```

以下示例列出特定数据库的所有任务的状态。

```
db2 "select task_id, task_type, database_name,
    varchar(bson_to_json(task_input_params), 500) as task_params
    from table(rdsadmin.get_task_status(null,'testdb', null))"
```

以下示例将 JSON 值输出为列。

```
db2 "select varchar(r.task_type,25) as task_type, varchar(r.lifecycle,10) as lifecycle,
 r.created_at, u.* from
    table(rdsadmin.get_task_status(null,null,'restore_db')) as r,
 json_table(r.task_input_params, 'strict $' columns(s3_prefix varchar(500)
    null on empty, s3_bucket_name varchar(500) null on empty) error on error ) as U"
```

响应

`rdsadmin.get_task_status` 用户定义函数返回以下各列：

`TASK_ID`

任务的 ID。

`TASK_TYPE`

取决于输入参数。

- `ADD_GROUPS` – 添加组。
- `ADD_USER` – 添加用户。
- `ALTER_BUFFERPOOL` – 更改缓冲池。
- `ALTER_TABLESPACE` – 更改表空间。
- `CHANGE_PASSWORD` – 更改用户的密码。
- `COMPLETE_ROLLFORWARD` – 完成 `rdsadmin.rollforward_database` 任务并激活数据库。
- `CREATE_BUFFERPOOL` – 创建缓冲池。
- `CREATE_DATABASE` – 创建数据库。
- `CREATE_ROLE` – 为用户创建 Db2 角色。
- `CREATE_TABLESPACE` – 创建表空间。
- `DROP_BUFFERPOOL` – 删除缓冲池。
- `DROP_DATABASE` – 删除数据库。
- `DROP_TABLESPACE` – 删除表空间。
- `LIST_USERS` – 列出所有用户。

- REMOVE_GROUPS – 移除组。
- REMOVE_USER – 移除用户。
- RESTORE_DB – 还原完整数据库。
- ROLLFORWARD_DB_LOG – 对数据库日志执行 rdsadmin.rollforward_database 任务。
- ROLLFORWARD_STATUS – 返回 rdsadmin.rollforward_database 任务的状态。
- UPDATE_DB_PARAM – 更新数据参数。

DATABASE_NAME

与任务关联的数据库的名称。

COMPLETED_WORK_BYTES

任务还原的字节数。

DURATION_MINS

完成任务所花费的时间。

LIFECYCLE

任务的状态。可能的状态：

- CREATED – 将任务提交给 Amazon RDS 后，Amazon RDS 会将状态设置为 CREATED。
- IN_PROGRESS – 任务开始后，Amazon RDS 将状态设置为 IN_PROGRESS。将状态从 CREATED 更改为 IN_PROGRESS 可能需要多达 5 分钟时间。
- SUCCESS – 任务完成后，Amazon RDS 将状态设置为 SUCCESS。
- ERROR – 如果还原任务失败，Amazon RDS 会将状态设置为 ERROR。有关错误的更多信息，请参阅TASK_OUPUT。

CREATED_BY

创建命令的 authid。

CREATED_AT

创建任务的日期和时间。

LAST_UPDATED_AT

上次更新任务的日期和时间。

TASK_INPUT_PARAMS

参数因任务类型而异。所有输入参数都以 JSON 对象的形式表示。例如，RESTORE_DB 任务的 JSON 键如下：

- DBNAME
- RESTORE_TIMESTAMP
- S3_BUCKET_NAME
- S3_PREFIX

TASK_OUTPUT

有关任务的其他信息。如果在原生还原过程中发生错误，则此列将包含有关错误的信息。

响应示例

以下响应示例显示已成功创建名为 TESTJP 的数据库。有关更多信息，请参阅 the section called “rdsadmin.create_database” 存储过程。

```
`1 SUCCESS CREATE_DATABASE RDSDB 2023-10-24-18.32.44.962689 2023-10-24-18.34.50.038523
 1 TESTJP { "CODESET" : "IBM-437", "TERRITORY" : "JP", "COLLATION" : "SYSTEM",
 "AUTOCONFIGURE_CMD" : "", "PAGESIZE" : 4096 }
2023-10-24-18.33.30.079048 Task execution has started.

2023-10-24-18.34.50.038523 Task execution has completed successfully`.
```

以下响应示例解释了删除数据库失败的原因。有关更多信息，请参阅 the section called “rdsadmin.drop_database” 存储过程。

```
1 ERROR DROP_DATABASE RDSDB 2023-10-10-16.33.03.744122 2023-10-10-16.33.30.143797 -
 2023-10-10-16.33.30.098857 Task execution has started.
2023-10-10-16.33.30.143797 Caught exception during executing task id 1, Aborting task.
Reason Dropping database created via rds CreateDBInstance api is not allowed.
Only database created using rdsadmin.create_database can be dropped
```

以下响应示例显示数据库的成功还原。有关更多信息，请参阅 the section called “rdsadmin.restore_database” 存储过程。

```
1 RESTORE_DB  SAMPLE  SUCCESS

{ "S3_BUCKET_NAME" : "amzn-s3-demo-bucket", "S3_PREFIX" :
 "SAMPLE.0.rdsdb3.DBPART000.20230413183211.001", "RESTORE_TIMESTAMP" :
 "20230413183211", "BACKUP_TYPE" : "offline" }
```

```
2023-11-06-18.31.03.115795 Task execution has started.
2023-11-06-18.31.04.300231 Preparing to download
2023-11-06-18.31.08.368827 Download complete. Starting Restore
2023-11-06-18.33.13.891356 Task Completed Successfully
```

rdsadmin.list_databases

返回在 RDS for Db2 数据库实例上运行的所有数据库的列表。

语法

```
db2 "select * from table(rdsadmin.list_databases())"
```

使用说明

此用户定义的函数不指定数据库是处于激活还是停用状态。

如果您未在列表中看到您的数据库，可调用 the section called “rdsadmin.get_task_status” 用户定义的函数并查找错误消息。

响应

`rdsadmin.list_databases` 用户定义的函数返回以下各列：

`DATABASE_NAME`

数据库的名称。

`CREATE_TIME`

创建数据库的日期和时间。

响应示例

以下响应示例显示了数据库的列表及其创建时间。`rdsadmin` 是 Amazon RDS 管理的数据库，并且始终显示在输出中。

```
DATABASE_NAME   CREATE_TIME
--------------- --------------------------
rdsadmin        2024-10-22-03.37.48.535671
TEST            2024-10-22-03.39.36.818679
TEST1           2024-10-22-03.57.15.218009
TEST2           2024-10-22-03.59.28.029556
```

Amazon RDS for Db2 故障排除

以下内容可以帮助您解决在 RDS for Db2 中遇到的问题。

要详细了解如何排查一般 Amazon RDS 问题，请参阅Amazon RDS 故障排除。

主题

数据库连接错误

以下错误消息表示数据库连接失败，因为服务器没有足够的内存。

```
SQL1643C The database manager failed to allocate shared memory because the
database manager instance memory limit has been reached.
```

增加数据库实例的内存，然后尝试再次连接到数据库。有关内存使用情况和数据库建议的信息，请参阅the section called “多个 Db2 数据库”。有关如何更新 RDS for Db2 数据库内存的信息，请参阅 the section called “rdsadmin.update_db_param”。

文件 I/O 错误

由于不同的原因（例如在使用 LOAD 命令或通过调用 rdsadmin.restore_database 存储过程时），您可能会遇到文件 I/O 错误。

在本例中，您运行以下 LOAD 命令：

```
db2 "call sysproc.admin_cmd('load from "DB2REMOTE://s3test//public/datapump/t6.del" of
 del lobs from "DB2REMOTE://s3test/public/datapump/" modified by lobsinfile MESSAGES ON
 SERVER insert INTO RDSDB.t6 nonrecoverable ')"
```

LOAD 命令将返回以下消息：

```

  Result set 1
  --------------
```

```
  ROWS_READ            ROWS_SKIPPED         ROWS_LOADED          ROWS_REJECTED
 ROWS_DELETED         ROWS_COMMITTED       ROWS_PARTITIONED     NUM_AGENTINFO_ENTRIES
 MSG_RETRIEVAL







 MSG_REMOVAL
  -------------------- -------------------- -------------------- --------------------
 -------------------- -------------------- -------------------- --------------------
 ----------------------------------------------------------------------------------------------
 ----------------------------------------------------------------------------------------------
                    -                    -                    -                    -
                   -                    -                    -                    -
 SELECT SQLCODE, MSG FROM TABLE(SYSPROC.ADMIN_GET_MSGS('1594987316_285548770')) AS MSG




                                                                                    CALL
 SYSPROC.ADMIN_REMOVE_MSGS('1594987316_285548770')

  1 record(s) selected.

  Return Status = 0

SQL20397W  Routine "SYSPROC.ADMIN_CMD" execution has completed, but at least
one error, "SQL1652", was encountered during the execution. More information
is available.  SQLSTATE=01H52
```

要查看错误消息，请按照上一个响应中建议的方式运行 SQL 命令。SELECT SQLCODE, MSG FROM TABLE(SYSPROC.ADMIN_GET_MSGS('1594987316_285548770')) AS MSG 返回以下消息：

```
SQLCODE   MSG
---------
 ----------------------------------------------------------------------------------------------
SQL2025N  An I/O error occurred. Error code “438”. Media on which this error occurred:
 “DB2REMOTE://s3test//public/datapump/t6.del”

SQL3500W The utility is beginning the LOAD phase at time “07/05/2024 21:21:48.082954”

SQL1652N File I/O error occurred
```

Db2 诊断日志包含与以下内容类似的日志文件：

```
2024-07-05-21.20.09.440609+000 I1191321E864          LEVEL: Error
PID     : 2710                 TID : 139619509200640 PROC : db2sysc 0
INSTANCE: rdsdb                NODE : 000            DB   : NTP
APPHDL  : 0-12180              APPID: xxx.xx.x.xxx.xxxxx.xxxxxxxxxxxx
UOWID   : 5                    ACTID: 1
AUTHID  : ADMIN                HOSTNAME: ip-xx-xx-x-xx
EDUID   : 147                  EDUNAME: db2lmr 0
FUNCTION: DB2 UDB, oper system services, sqloS3Client_GetObjectInfo, probe:219
MESSAGE : ZRC=0x870F01B6=-2029059658=SQLO_FAILED
          "An unexpected error is encountered"
DATA #1 : String, 29 bytes
S3:HeadObject request failed.
DATA #2 : signed integer, 4 bytes
99
DATA #3 : String, 0 bytes
Object not dumped: Address: 0x00007EFC08A9AE38 Size: 0 Reason: Zero-length data
DATA #4 : String, 33 bytes
curlCode: 28, Timeout was reached
```

此文件 I/O 错误可能是由多种不同的情况造成的。例如，与用于创建 RDS For Db2 数据库实例的安全组关联的 VPC 可能缺少 Amazon S3 网关端点。要让 RDS for Db2 访问 Amazon S3，此端点至关重要。如果您的 RDS for Db2 数据库实例位于私有子网中，则需要 Amazon S3 网关端点。可以通过配置 Amazon RDS 子网组，来指定您的数据库实例是使用私有子网还是公有子网。有关更多信息，请参阅使用数据库子网组。

主题

- 步骤 1：为 Amazon S3 创建 VPC 网关端点
- 步骤 2：确认适用于 Amazon S3 的 VPC 网关端点存在

步骤 1：为 Amazon S3 创建 VPC 网关端点

要让 RDS for Db2 数据库实例与 Amazon S3 交互，需创建一个 VPC，然后创建一个 Amazon S3 网关端点，供私有子网使用。

为 S3 创建 VPC 网关端点

1. 创建 VPC。有关更多信息，请参阅《Amazon Virtual Private Cloud 用户指南》中的创建 VPC。

2. 创建供私有子网使用的 Amazon S3 网关端点。有关更多信息，请参阅《Amazon PrivateLink 指南》中的 Gateway endpoint。

步骤 2：确认适用于 Amazon S3 的 VPC 网关端点存在

确认您已通过 Amazon Web Services Management Console或 Amazon CLI 成功创建 Amazon S3 网关端点。

控制台

确认 Amazon S3 网关端点

1. 登录 Amazon Web Services Management Console并打开 Amazon VPC 控制台（网址为 https://console.amazonaws.cn/vpc）。
2. 在控制台的右上角，选择您的 VPC 所在的 Amazon Web Services 区域。
3. 选择您所创建的 VPC。
4. 在资源映射选项卡的网络连接下，确认已列出一个 Amazon S3 网关端点。

Amazon CLI

要确认 Amazon S3 网关端点，请运行 describe-vpc-endpoints 命令。在以下示例中，将 *vpc_id* 替换为 VPC ID，将 *region* 替换为您所在的 Amazon Web Services 区域，将 *profile* 替换为您的配置文件名称。

对于 Linux、macOS 或 Unix：

```
aws ec2 describe-vpc-endpoints \
    --filters "Name=vpc-id,Values=$vpc_id" \
    "Name=service-name,\
    Values=com.amazonaws.${region}.s3" \
    --region $region --profile=$profile \
    --query "VpcEndpoints[*].VpcEndpointId" --output text
```

对于 Windows：

```
aws ec2 describe-vpc-endpoints ^
    --filters "Name=vpc-id,Values=$vpc_id" ^
    "Name=service-name,^
    Values=com.amazonaws.${region}.s3" ^
```

```
    --region $region --profile=$profile ^
    --query "VpcEndpoints[*].VpcEndpointId" --output text
```

如果存在 Amazon S3 网关端点，此命令将生成类似于以下示例的输出。

```
[
    "vpce-0ea810434ff0b97e4"
]
```

如果 Amazon S3 网关端点不存在，此命令将生成类似于以下示例的输出。

```
[]
```

如果您看不到列出的 Amazon S3 网关端点，请执行步骤 1：为 Amazon S3 创建 VPC 网关端点。

存储过程错误

本节介绍在调用存储过程时返回的各种错误以及如何解决这些错误。

类别	存储过程错误
数据库	rdsadmin.activate_database 错误
数据库	rdsadmin.create_database 错误
数据库	rdsadmin.deactivate_database 错误
数据库	rdsadmin.drop_database 错误
数据库	rdsadmin.reactivate_database 错误
数据库	rdsadmin.restore_database 错误
数据库	rdsadmin.update_db_param 错误
表空间	rdsadmin.alter_tablespace 错误

rdsadmin.activate_database 错误

调用 the section called “rdsadmin.activate_database” 存储过程时可能会出现以下错误。

错误	错误消息
Failed to allocate shared memory	SQL1643C The database manager failed to allocate shared memory because the database manager instance memory limit has been reached.
Unable to activate because of running processes	The database can’t be activated because it's in the process of being created or restored.

无法分配共享内存

以下错误消息表明存储过程未能激活数据库，因为数据库实例没有足够的内存。

```
SQL1643C The database manager failed to allocate shared memory because the database
 manager instance memory limit has been reached.
```

增加数据库实例的内存，然后再次调用 `rdsadmin.activate_database` 存储过程。有关内存使用情况和数据库建议的信息，请参阅the section called “多个 Db2 数据库”。

由于进程正在运行，无法激活

以下错误消息表明存储过程无法激活数据库，因为 `rdsadmin.create_database` 或 `rdsadmin.restore_database` 存储过程正在运行。

```
The database can’t be activated because it's in the process of being created or
 restored.
```

等待几分钟，然后再次调用 `rdsadmin.activate_database` 存储过程。

rdsadmin.alter_tablespace 错误

调用 the section called “rdsadmin.alter_tablespace” 存储过程时可能会出现以下错误。

错误	错误消息
Statement not valid	DB21034E The command was processed as an SQL statement because it was not a valid Command Line Processor command. During SQL processing it returned:

错误	错误消息
	SQL1763N Invalid ALTER TABLESPACE statement for table space "TBSP_TEST" due to reason "12"
tablespace_prefetch_size value not valid	Invalid tablespace_prefetch_size. Set value to AUTOMATIC or to a non-zero positive numerical value.
tablespace_prefetch_size numerical value not valid	Invalid tablespace_prefetch_size. The number of pages can't be greater than 32767.
Parameter can't be used with tablespace_prefetch_size	You can't use tablespace_prefetch_size with {*parameter* }.
Tablespace change failed	The change to tablespace { *tablespace_name}* failed because you can only alter LARGE or REGULAR tablespaces.

语句无效

以下错误消息表明存储过程将互斥的可选参数与其他可选参数组合使用。对于 rdsadmin.alter_tablespace 存储过程，可选参数 reduce_max、reduce_stop、reduce_value、lower_high_water、lower_high_water_stop 和 switch_online 是互斥的。不能将它们与 rdsadmin.alter_tablespace 存储过程中的任何其他可选参数（如 buffer_pool_name）组合使用。如果将它们组合使用，则当您调用 rdsadmin.get_task_status 用户定义的函数时，Db2 将返回此错误消息。

```
DB21034E The command was processed as an SQL statement because it was not a valid
 Command Line Processor command. During SQL processing it returned:
SQL1763N Invalid ALTER TABLESPACE statement for table space "TBSP_TEST" due to reason
 "12"
```

再次调用 rdsadmin.alter_tablespace 存储过程，不要将互斥的可选参数与其他可选参数组合使用。然后调用 rdsadmin.get_task_status 用户定义的函数。有关更多信息，请参阅 rdsadmin.alter_tablespace 和 rdsadmin.get_task_status。

tablespace_prefetch_size 值无效

以下错误消息表明您未将 `tablespace_prefetch_size` 设置为 `AUTOMATIC` 或为非正数值。例如，您尝试将其设置为 `testinput`。

```
Invalid tablespace_prefetch_size. Set value to AUTOMATIC or to a non-zero positive
 numerical value.
```

再次调用 `rdsadmin.alter_tablespace` 存储过程，并将 `tablespace_prefetch_size` 设置为 `AUTOMATIC` 或非正数值。

tablespace_prefetch_size 数值无效

以下错误消息表明您将 `tablespace_prefetch_size` 设置为大于 32767 的数值。

```
Invalid tablespace_prefetch_size. The number of pages can't be greater than 32767.
```

再次调用 `rdsadmin.alter_tablespace` 存储过程，并将 `tablespace_prefetch_size` 设置为小于等于 32767 的非零整数值。

参数不能与 tablespace_prefetch_size 一起使用

以下错误消息表明您尝试将 `tablespace_prefetch_size` 与不兼容参数一起使用。

```
You can't use tablespace_prefetch_size with {parameter}.
```

再次调用 `rdsadmin.alter_tablespace` 存储过程，并且仅将 `tablespace_prefetch_size` 与兼容的参数一起使用。有关可以与 `tablespace_prefetch_size` 一起使用的其他参数的信息，请参阅 rdsadmin.alter_tablespace。

表空间更改失败

以下错误消息表明您尝试更改表空间。

```
The change to tablespace {tablespace_name} failed because you can only alter LARGE or
 REGULAR tablespaces.
```

rdsadmin.create_database 错误

调用 the section called “rdsadmin.create_database” 存储过程时可能出现以下错误。

错误	错误消息
Failed to allocate shared memory	SQL1643C The database manager failed to allocate shared memory because the database manager instance memory limit has been reached.

无法分配共享内存

以下错误消息表明存储过程无法创建数据库，因为数据库实例没有足够的内存。

```
SQL1643C The database manager failed to allocate shared memory because the database
 manager instance memory limit has been reached.
```

增加数据库实例的内存，然后再次调用 rdsadmin.create_database 存储过程。有关内存使用情况和数据库建议的信息，请参阅the section called “多个 Db2 数据库”。

要确认数据库已创建，可调用 the section called “rdsadmin.list_databases” 用户定义的函数并检查新数据库是否已列出。

rdsadmin.deactivate_database 错误

调用 the section called “rdsadmin.deactivate_database” 存储过程时可能出现以下错误。

错误	错误消息
Unable to deactivate because of running processes	The database can’t be deactivated because it's in the process of being created or restored.

由于进程正在运行，无法停用

以下错误消息表明存储过程无法停用数据库，因为 rdsadmin.create_database 或 rdsadmin.restore_database 存储过程正在运行。

```
The database can’t be deactivated because it's in the process of being created or
 restored.
```

等待几分钟，然后再次调用 rdsadmin.deactivate_database 存储过程。

rdsadmin.drop_database 错误

调用 the section called “rdsadmin.drop_database” 存储过程时可能会出现以下错误。

错误	错误消息
Database name doesn't exist	SQL0438N Application raised error or warning with diagnostic text: "Cannot drop database. Database with provided name does not exist". SQLSTATE=99993
Return status = 0	Return Status = 0
Dropping database not allowed	1 ERROR DROP_DATABASE RDSDB 2023-10-10-16.33.03.744122 2023-10-10-16.33.30.143797 - 2023-10-10-16.33.30.098857 Task execution has started. 2023-10-10-16.33.30.143797 Caught exception during executing task id 1, Aborting task. Reason Dropping database created via rds CreateDBInstance api is not allowed. Only database created using rdsadmin.create_database can be dropped

数据库名称不存在

以下错误消息表明您在 `rdsadmin.drop_database` 存储过程中传递的数据库名称不正确。

```
SQL0438N Application raised error or warning with diagnostic text: "Cannot
drop database. Database with provided name does not exist". SQLSTATE=99993
```

使用正确的数据库名称再次调用 `rdsadmin.drop_database` 存储过程。要确认数据库已删除，可调用 the section called “rdsadmin.list_databases” 用户定义的函数并检查已删除的数据库是否未列出。

返回状态 = 0

以下错误消息表明无法完成存储过程。

```
Return Status = 0
```

收到 `Return Status = 0` 后，调用 rdsadmin.get_task_status 用户定义的函数。

不允许删除数据库

以下错误消息表明您通过 Amazon RDS 控制台或 Amazon CLI 创建了数据库。只有通过调用 the section called “rdsadmin.create_database” 存储过程创建数据库时，才能使用 rdsadmin.drop_database 存储过程。

```
1 ERROR DROP_DATABASE RDSDB 2023-10-10-16.33.03.744122 2023-10-10-16.33.30.143797 -
 2023-10-10-16.33.30.098857 Task execution has started.
2023-10-10-16.33.30.143797 Caught exception during executing task id 1, Aborting task.
Reason Dropping database created via rds CreateDBInstance api is not allowed.
Only database created using rdsadmin.create_database can be dropped
```

要删除您通过 Amazon RDS 控制台或 Amazon CLI 创建的数据库，请使用客户端连接到该数据库，然后运行相应的命令。

rdsadmin.reactivate_database 错误

调用 the section called “rdsadmin.reactivate_database” 存储过程时可能出现以下错误。

错误	错误消息
Failed to allocate shared memory	SQL1643C The database manager failed to allocate shared memory because the database manager instance memory limit has been reached.
Unable to reactivate because of running processes	The database can’t be reactivated because it's in the process of being created or restored.

无法分配共享内存

以下错误消息表明存储过程未能激活数据库，因为数据库实例没有足够的内存。

```
SQL1643C The database manager failed to allocate shared memory because the database
 manager instance memory limit has been reached.
```

增加数据库实例的内存，然后再次调用 rdsadmin.activate_database 存储过程。有关内存使用情况和数据库建议的信息，请参阅the section called “多个 Db2 数据库”。

由于进程正在运行，无法重新激活

以下错误消息表明存储过程无法重新激活数据库，因为 rdsadmin.create_database 或 rdsadmin.restore_database 存储过程正在运行。

```
The database can’t be reactivated because it's in the process of being created or
 restored.
```

等待几分钟，然后再次调用 rdsadmin.reactivate_database 存储过程。

rdsadmin.restore_database 错误

调用 the section called “rdsadmin.restore_database” 存储过程时可能出现以下错误：

错误	错误消息
Insufficient disk space	Aborting task. Reason Restoring your database failed because of insufficient disk space. Increase the storage for your DB instance and rerun the rdsadmin.restore_database stored procedure.
Internal error	Caught exception during executing task id 104, Aborting task. Reason Internal Error
Non-fenced routines not allowed	Caught exception during executing task id 2, Aborting task. Reason Non fenced routines are not allowed. Please delete the routines and retry the restore.
Tablespaces not restored	Reason SQL0970N The system attempted to write to a read-only file. Reason SQL2563W The Restore process has completed successfully. However one or more table spaces from the backup were not restored.

磁盘空间不足

以下错误消息表明数据库实例没有足够的磁盘空间，无法还原数据库：

```
Aborting task. Reason Restoring your database failed because of insufficient
 disk space. Increase the storage for your DB instance and rerun the
 rdsadmin.restore_database stored procedure.
```

数据库实例上的可用空间必须是备份映像大小的两倍以上。如果备份映像已压缩，则数据库实例上的可用空间必须是备份映像大小的三倍以上。有关更多信息，请参阅the section called “增加数据库实例存储容量”。

增加磁盘空间，然后再次调用 `rdsadmin.restore_database` 存储过程。要确认数据库已还原，可调用 the section called “rdsadmin.list_databases” 用户定义的函数并检查已还原的数据库是否已列出。

内部错误

以下错误消息表明存储过程遇到了内部错误：

```
Caught exception during executing task id 104, Aborting task. Reason Internal Error
```

请联系 Amazon Support。

不支持非防护例程

以下错误消息表明您的数据库包含非防护例程：

```
Caught exception during executing task id 2, Aborting task. Reason Non fenced routines
 are not allowed. Please delete the routines and retry the restore.
```

RDS for Db2 不支持非防护例程。从源数据库中移除非防护例程，然后再次调用 `rdsadmin.restore_database`。要确认数据库已还原，可调用 the section called “rdsadmin.list_databases” 用户定义的函数并检查已还原的数据库是否已列出。有关更多信息，请参阅 the section called “非防护例程”。

未还原表空间

以下错误消息表明 RDS for Db2 已成功还原数据库，但无法还原一个或多个表空间：

```
Reason SQL0970N The system attempted to write to a read-only file.
Reason SQL2563W The Restore process has completed successfully. However one or more
 table spaces from the backup were not restored.
```

RDS for Db2 不支持非自动存储。将非自动存储转换为自动存储，然后再次调用 `rdsadmin.restore_database`。有关更多信息，请参阅 IBM Db2 文档中的 Converting a nonautomatic storage database to use automatic storage。

具有非自动 SMS 存储的数据库需要手动还原。如果数据库具有非自动 SMS 存储空间，请联系 Amazon Support。

有关非自动存储和一次性迁移的信息，请参阅the section called “迁移期间的非自动存储表空间”。

rdsadmin.update_db_param 错误

调用 the section called “rdsadmin.update_db_param” 存储过程时可能出现以下错误。

错误	错误消息
Parameter not supported or modifiable	SQL0438N Application raised error or warning with diagnostic text: "Parameter is either not supported or not modifiable to customers". SQLSTATE=99993

参数不受支持或不可修改

以下错误消息表明您尝试修改的数据库配置参数不受支持或不可修改。

```
SQL0438N Application raised error or warning with diagnostic text: "Parameter
is either not supported or not modifiable to customers". SQLSTATE=99993
```

您可以通过查看参数组来查看哪些参数可修改。有关更多信息，请参阅the section called “查看数据库参数组的参数值”。

Amazon RDS for MariaDB

Amazon RDS 支持若干版本 MariaDB 的数据库实例。有关支持的版本的完整信息，请参阅 Amazon RDS 上的 MariaDB 版本。

要创建 MariaDB 数据库实例，请使用 Amazon RDS 管理工具或界面。然后，您可以使用 Amazon RDS 工具对数据库实例执行管理操作。这些操作包括如下各项：

- 重新配置数据库实例或调整其大小
- 授权到数据库实例的连接
- 从备份或快照中创建和还原
- 创建多可用区辅助实例
- 创建只读副本
- 监控数据库实例的性能

要存储和访问数据库实例中的数据，请使用标准 MariaDB 实用程序和应用程序。

MariaDB 在所有 Amazon Web Services 区域中都可用。有关 Amazon Web Services 区域的更多信息，请参阅区域、可用区和 Local Zones。

您可以使用 Amazon RDS for MariaDB 数据库来构建符合 HIPAA 标准的应用程序。您可以在与 Amazon 签订的业务合作协议 (BAA) 下存储与医疗保健相关的信息，包括受保护医疗信息 (PHI)。有关更多信息，请参阅 HIPAA 合规性。Amazon范围内的服务已由第三方审核员完全评估，由此获得了证书、合规性证明或操作授权书（ATO）。有关更多信息，请参阅合规性计划范围内的Amazon服务。

在创建数据库实例之前，完成设置 Amazon RDS 环境 中的步骤。创建数据库实例时，RDS 主用户将获得 DBA 权限，但有一些限制。此账户用于管理任务，如创建其他数据库账户。

您可执行以下操作：

- 数据库实例
- 数据库快照
- 时间点恢复
- 自动备份
- 手动备份

您可以在基于 Amazon VPC 的虚拟私有云 (VPC) 中使用运行 MariaDB 的数据库实例。您还可以通过启用各种选项向 MariaDB 数据库实例添加特征。Amazon RDS 支持对 MariaDB 的多可用区部署，这是一种高可用性失效转移解决方案。

> **Important**
>
> 为了提供托管服务体验，Amazon RDS 未提供数据库实例的 Shell 访问权限。并且还限制了对需要高级权限的某些系统程序和表的访问。您可以使用标准 SQL 客户端（例如 mysql 客户端）访问您的数据库。但是，您无法使用远程登录或安全外壳 (SSH) 直接访问主机。

主题

Amazon RDS 上的 MariaDB 特征支持

RDS for MariaDB 支持 MariaDB 的大部分特征和功能。有些功能可能仅提供有限支持或受限制的权限。

您可以在数据库的新增功能？页面上筛选新的 Amazon RDS 特征。对于 Products (产品)，请选择 Amazon RDS。然后使用 **MariaDB 2023** 之类的关键词进行搜索。

> **Note**
> 以下列表并不详尽。

有关 Amazon RDS 上 MariaDB 功能支持的更多信息，请参阅以下主题。

主题

- Amazon RDS 上的 MariaDB 支持的存储引擎
- Amazon RDS 上的 MariaDB 的缓存预热
- Amazon RDS 不支持的 MariaDB 特征

Amazon RDS for MariaDB 各主要版本上的 MariaDB 特征支持

在以下几部分，查找有关 Amazon RDS for MariaDB 各主要版本上的 MariaDB 特征支持的信息：

主题

- Amazon RDS 上对 MariaDB 11.4 的支持
- Amazon RDS 上对 MariaDB 10.11 的支持
- Amazon RDS 对 MariaDB 10.6 的支持
- Amazon RDS 对 MariaDB 10.5 的支持
- Amazon RDS 上对 MariaDB 10.4 的支持

有关支持的 Amazon RDS for MariaDB 次要版本的信息，请参阅 Amazon RDS 上的 MariaDB 版本。

Amazon RDS 上对 MariaDB 11.4 的支持

对于运行 MariaDB 版本 11.4 或更高版本的数据库实例，Amazon RDS 支持以下新特征。

- 密码库 – RDS for MariaDB 将 OpenSSL 替换为已获得 FIPS 140-3 认证的 Amazon Libcrypto（Amazon-LC）。
- 简单密码检查插件 – 您可以使用 MariaDB 简单密码检查插件来检查密码是否包含至少特定数量的特定类型字符。有关更多信息，请参阅 the section called “密码验证插件”。

- Cracklib 密码检查插件 – 您可以使用 MariaDB Cracklib 密码检查插件来检查新密码的强度。有关更多信息，请参阅 the section called “密码验证插件”。
- InnoDB 增强功能 – 这些增强功能包括以下项目：
  - 更改缓冲区已删除。有关更多信息，请参阅 InnoDB 更改缓冲。
  - InnoDB 碎片整理已删除。有关更多信息，请参阅 InnoDB 碎片整理。
- 新权限 – 管理员用户现在也拥有 SHOW CREATE ROUTINE 权限。此权限允许被授权者查看其他用户拥有的例程的 SHOW CREATE 定义语句。有关更多信息，请参阅数据库权限。
- 复制改进 – MariaDB 版本 11.4 数据库实例支持二进制日志索引。您可以为每个二进制日志文件创建 GTID 索引。这些索引缩短了查找 GTID 所需的时间，从而提高了复制性能。有关更多信息，请参阅二进制日志索引。
- 已弃用或删除的参数 – 对于 MariaDB 版本 11.4 数据库实例，以下参数已弃用或删除：
  - 已从 optimizer_switch 中删除 engine_condition_pushdown
  - innodb_change_buffer_max_size
  - innodb_defragment
  - 已从 tls_version 中删除 TLSv1.0 和 TLSv1.1
- 参数的新默认值 – innodb_undo_tablespaces 参数的默认值从 0 更改为 3。
- 参数的新有效值 – 对于 MariaDB 版本 11.4 数据库实例，以下参数具有新的有效值：
  - binlog_row_image 参数的有效值现在包括 FULL_NODUP。
  - OLD_MODE 参数的有效值现在包括 NO_NULL_COLLATION_IDS。
- 新参数 – 对于 MariaDB 版本 11.4 数据库实例，以下参数是新的：
  - transaction_isolation 参数替换 tx_isolation 参数。
  - transaction_read_only 参数替换 tx_read_only 参数。
  - block_encryption_mode 参数定义了 AES_ENCRYPT() 和 AES_DECRYPT() 函数的默认区块加密模式。
  - character_set_collations 定义了字符集默认排序规则的覆盖。
  - binlog_gtid_index、binlog_gtid_index_page_size 和 binlog_gtid_index_span_min 定义了二进制日志 GTID 索引的属性。有关更多信息，请参阅二进制日志索引。

有关所有 MariaDB 11.4 特征的列表及其文档，请参阅 MariaDB 网站上的 MariaDB 11.4 中的更改和改进和发行说明 - MariaDB 11.4 系列。

有关不支持特征的列表，请参阅 Amazon RDS 不支持的 MariaDB 特征。

Amazon RDS 上对 MariaDB 10.11 的支持

对于运行 MariaDB 版本 10.11 或更高版本的数据库实例，Amazon RDS 支持以下新特征。

- 密码重用检查插件 – 您可以使用 MariaDB 密码重用检查插件来防止用户重用密码并设置密码的保留期。有关更多信息，请参阅密码重用检查插件。
- GRANT TO PUBLIC 授权 – 您可以向所有具有服务器访问权限的用户授予权限。有关更多信息，请参阅 GRANT TO PUBLIC。
- SUPER 和 READ ONLY ADMIN 权限分离 – 您可以删除所有用户的 READ ONLY ADMIN 权限，即使是以前拥有 SUPER 权限的用户也是如此。
- 安全 – 现在可以将选项 `--ssl` 设置为 MariaDB 客户端的默认设置。如果配置不正确，MariaDB 不再以静默方式禁用 SSL。
- SQL 命令和函数 – 您现在可以使用 `SHOW ANALYZE FORMAT=JSON` 命令和函数 `ROW_NUMBER`、`SFORMAT` 以及 `RANDOM_BYTES`。`SFORMAT` 允许字符串格式化，且默认情况下处于启用状态。只需一条命令即可将分区转换为表和将表转换为分区。围绕 `JSON_*()` 函数还有一些改进。在 10.10 及更高版本中，已弃用 `DES_ENCRYPT` 和 `DES_DECRYPT` 函数。有关更多信息，请参阅 SFORMAT。
- InnoDB 增强功能 – 这些增强功能包括以下项目：
  - 重做日志的性能改进，以减少写入放大并提高并发性。
  - 无需重新初始化数据目录，即可更改撤消表空间。此增强功能减少了控制面板开销。它要求重启，但在更改撤消表空间后不需要重新初始化。
  - 支持 `CHECK TABLE … EXTENDED` 和内部索引降序。
  - 改进了批量插入功能。
- 二进制日志更改 – 这些更改包括以下项目：
  - 分两个阶段记录 `ALTER` 以减少复制延迟。默认情况下，`binlog_alter_two_phase` 参数处于禁用状态，但可以通过参数组启用。
  - 记录 `explicit_defaults_for_timestamp`。
  - 如果事务可以安全地回滚，则不再记录 `INCIDENT_EVENT`。
- 复制改进 – 如果主服务器支持 GTID 复制，则 MariaDB 版本 10.11 数据库实例默认情况下使用此功能。另外，`Seconds_Behind_Master` 更精确。
- 客户端 – 您可以将新的命令行选项用于 `mysqlbinlog` 和 `mariadb-dump`。您可以使用 `mariadb-dump` 转储和还原历史数据。
- 系统版本控制 – 您可以修改历史记录。MariaDB 会自动创建新的分区。

- 原子 DDL – CREATE OR REPLACE 现在是原子的。语句要么成功，要么完全相反。
- 重做日志写入 – 重做日志异步写入。
- 存储函数 – 存储函数现在支持存储过程中的相同 IN、OUT 和 INOUT 参数。
- 已弃用或移除的参数 – 对于 MariaDB 版本 10.11 数据库实例，以下参数已弃用或删除：
  - innodb_change_buffering
  - innodb_disallow_writes
  - innodb_log_write_ahead_size
  - innodb_prefix_index_cluster_optimization
  - keep_files_on_create
  - old
- 动态参数 – 对于 MariaDB 版本 10.11 数据库实例，以下参数现在是动态的：
  - innodb_log_file_size
  - innodb_write_io_threads
  - innodb_read_io_threads
- 参数的新默认值 – 对于 MariaDB 版本 10.11 数据库实例，以下参数具有新的默认值：
  - explicit_defaults_for_timestamp 参数的默认值已从 OFF 更改为 ON。
  - optimizer_prune_level 参数的默认值已从 1 更改为 2。
- 参数的新有效值 – 对于 MariaDB 版本 10.11 数据库实例，以下参数具有新的有效值：
  - old 参数的有效值已合并到 old_mode 参数的有效值中。
  - histogram_type 参数的有效值现在包括 JSON_HB。
  - innodb_log_buffer_size 参数的有效值范围现在为 262144 至 4294967295（256KB 至 4096MB）。
  - innodb_log_file_size 参数的有效值范围现在为 4194304 至 512GB（4MB 至 512GB）。
  - optimizer_prune_level 参数的有效值现在包括 2。
- 新参数 – 对于 MariaDB 版本 10.11 数据库实例，以下参数是新的：
  - binlog_alter_two_phase 参数可以改善复制性能。
  - log_slow_min_examined_row_limit 参数可以改善性能。
  - log_slow_query 参数和 log_slow_query_file 参数分别为 slow_query_log 和 slow_query_log_file 的别名。
  - optimizer_extra_pruning_depth
  - system_versioning_insert_history

有关所有 MariaDB 10.11 特征的列表及其文档，请参阅 MariaDB 网站上的 MariaDB 10.11 中的更改和改进和发行说明 - MariaDB 10.11 系列。

有关不支持特征的列表，请参阅 Amazon RDS 不支持的 MariaDB 特征。

Amazon RDS 对 MariaDB 10.6 的支持

对于运行 MariaDB 版本 10.6 或更高版本的数据库实例，Amazon RDS 支持以下新特征：

- MyRocks 存储引擎 - 您可以将 MyRocks 存储引擎与 RDS for MariaDB 结合使用，以优化写入密集型高性能 Web 应用程序的存储消耗。有关更多信息，请参阅 Amazon RDS 上的 MariaDB 支持的存储引擎 和 MyRocks。
- Amazon Identity and Access Management (IAM) 数据库身份验证 - 您可以使用 IAM 数据库身份验证来提高安全性和集中管理与 MariaDB 数据库实例的连接。有关更多信息，请参阅 适用于 MariaDB、MySQL 和 PostgreSQL 的 IAM 数据库身份验证。
- 升级选项 - 您现在可以从以前的任何主要版本（10.3、10.4、10.5）升级到 RDS for MariaDB 版本 10.6。您还可以将现有 MySQL 5.6 或 5.7 数据库实例的快照还原到 MariaDB 10.6 实例。有关更多信息，请参阅 升级 MariaDB 数据库引擎。
- 延迟复制 - 您现在可以设置一个只读副本落后于源数据库的可配置时间段。在标准 MariaDB 复制配置中，源和副本之间的复制延迟最小。使用延迟复制时，您可以将故意延迟设置为灾难恢复策略。有关更多信息，请参阅 使用 MariaDB 配置延迟复制。
- Oracle PL/SQL 兼容性 - 通过使用 RDS for MariaDB 版本 10.6，您可以更轻松地将旧版 Oracle 应用程序迁移到 Amazon RDS。有关更多信息，请参阅 SQL_MODE=ORACLE。
- 原子 DDL - 您的动态数据语言 (DDL) 语句与 RDS for MariaDB 版本 10.6 结合使用可能相对安全。`CREATE TABLE`、`ALTER TABLE`、`RENAME TABLE`、`DROP TABLE`、`DROP DATABASE` 和相关的 DDL 语句现在是原子的。要么语句成功，要么完全相反。有关更多信息，请参阅原子 DDL。
- 其他增强功能 - 这些增强功能包括在 SQL 中将 JSON 数据转换为关系格式的 `JSON_TABLE` 功能，以及使用 Innodb 更快地加载空表数据。它们还包括用于分析和故障排除的新 `sys_schema`，用于忽略未使用索引的优化器增强以及性能改进。有关更多信息，请参阅 JSON_TABLE。
- 参数的新默认值 – 对于 MariaDB 10.6 版本数据库实例，以下参数具有新的默认值：
  - 以下参数的默认值已从 `utf8` 更改为 `utf8mb3`：
    - character_set_client
    - character_set_connection
    - character_set_results
    - character_set_system

尽管这些参数的默认值已更改，但功能没有变化。有关更多信息，请参阅 MariaDB 文档中的支持的字符集和排序规则。

- collation_connection 参数的默认值已从 `utf8_general_ci` 更改为 `utf8mb3_general_ci`。尽管此参数的默认值已更改，但功能没有变化。
- old_mode 参数的默认值已从未设置更改为 `UTF8_IS_UTF8MB3`。尽管此参数的默认值已更改，但功能没有变化。

有关所有 MariaDB 10.6 特征的列表及其文档，请参阅 MariaDB 网站上的 MariaDB 10.6 中的更改和改进和发行说明 - MariaDB 10.6 系列。

有关不支持特征的列表，请参阅 Amazon RDS 不支持的 MariaDB 特征。

Amazon RDS 对 MariaDB 10.5 的支持

对于运行 MariaDB 版本 10.5 或更高版本的数据库实例，Amazon RDS 支持以下新特征：

- InnoDB 增强功能 – MariaDB 10.5 版本包括 InnoDB 增强功能。有关更多信息，请参阅 MariaDB 文档中的 InnoDB：性能改进等。
- Performance Schema 更新 – MariaDB 10.5 版本包括 Performance Schema 更新。有关更多信息，请参阅 MariaDB 文档中的 Performance Schema 更新以满足 MySQL 5.7 工具和表的要求。
- InnoDB 重做日志中的一个文件 – 在 10.5 之前的 MariaDB 版本中，`innodb_log_files_in_group` 参数的值设置为 `2`。在 MariaDB 10.5 版本中，此参数的值设置为 `1`。

  如果从之前的版本升级到 MariaDB 10.5 版本，而且未修改参数，则 `innodb_log_file_size` 参数值不变。但是，它适用于一个日志文件而非两个。因此，升级后的 MariaDB 10.5 版本数据库实例使用的重做日志大小是升级前所用大小的一半。此变化可能会对性能产生明显的影响。要解决此问题，您可以将 `innodb_log_file_size` 参数的值增大两倍。有关修改参数的信息，请参阅 在 Amazon RDS 中修改数据库参数组中的参数。
- SHOW SLAVE STATUS 命令不受支持 – 在 10.5 之前的 MariaDB 版本中，`SHOW SLAVE STATUS` 命令需要 `REPLICATION SLAVE` 特权。在 MariaDB 10.5 版本中，等效的 `SHOW REPLICA STATUS` 命令需要 `REPLICATION REPLICA ADMIN` 特权。此新权限不会向 RDS 主用户授予。

  请不要使用 `SHOW REPLICA STATUS` 命令，而是运行新的 `mysql.rds_replica_status` 存储过程以返回类似信息。有关更多信息，请参阅“mysql.rds_replica_status”。

- SHOW RELAYLOG EVENTS 命令不受支持 – 在 10.5 之前的 MariaDB 版本中，SHOW RELAYLOG EVENTS 命令需要 REPLICATION SLAVE 特权。在 MariaDB 10.5 版本中，此命令需要 REPLICATION REPLICA ADMIN 特权。此新权限不会向 RDS 主用户授予。
- 参数的新默认值 – 对于 MariaDB 10.5 版本数据库实例，以下参数具有新的默认值：
  - max_connections 参数的默认值已更改为 LEAST({DBInstanceClassMemory/25165760},12000)。有关 LEAST 参数函数的信息，请参阅 数据库参数函数。
  - innodb_adaptive_hash_index 参数的默认值已更改为 OFF (0)。
  - innodb_checksum_algorithm 参数的默认值已更改为 full_crc32。
  - innodb_log_file_size 参数的默认值已更改为 2 GB。

有关所有 MariaDB 10.5 特征的列表及其文档，请参阅 MariaDB 网站上的 MariaDB 10.5 中的更改和改进和发行说明 - MariaDB 10.5 系列。

有关不支持特征的列表，请参阅 Amazon RDS 不支持的 MariaDB 特征。

Amazon RDS 上对 MariaDB 10.4 的支持

对于运行 MariaDB 版本 10.4 或更高版本的数据库实例，Amazon RDS 支持以下新特征：

- 用户账户安全增强 – 密码过期和账户锁定改进
- 优化程序增强 – 优化程序跟踪特征
- InnoDB 增强 – 即时 DROP COLUMN 支持以及 VARCHAR 和 ROW_FORMAT=DYNAMIC 的即时 ROW_FORMAT=COMPACT 扩展
- 新参数 – 包括 tcp_nodedelay、tls_version 和 gtid_cleanup_batch_size

有关所有 MariaDB 10.4 特征的列表及其文档，请参阅 MariaDB 网站上的 MariaDB 10.4 中的更改和改进和发行说明 - MariaDB 10.4 系列。

有关不支持特征的列表，请参阅 Amazon RDS 不支持的 MariaDB 特征。

Amazon RDS 上的 MariaDB 支持的存储引擎

RDS for MariaDB 支持以下存储引擎。

主题

- InnoDB 存储引擎
- MyRocks 存储引擎

目前，RDS for MariaDB 不支持其他存储引擎。

InnoDB 存储引擎

尽管 MariaDB 支持功能不同的多种存储引擎，但并非所有引擎都针对恢复和数据耐久性而进行了优化。InnoDB 是 Amazon RDS 上的 MariaDB 数据库实例的推荐存储引擎。Amazon RDS 的时间点还原和快照还原等特征要求使用可恢复的存储引擎，且仅支持 MariaDB 版本的推荐存储引擎。

有关更多信息，请参阅 InnoDB。

MyRocks 存储引擎

MyRocks 存储引擎在 RDS for MariaDB 版本 10.6 及更高版本中提供。在生产数据库中使用 MyRocks 存储引擎之前，我们建议您执行彻底的基准测试和测试，以验证您的使用案例相比 InnoDB 的任何潜在优势。

MariaDB 版本 10.6 的默认参数组包括 MyRocks 参数。有关更多信息，请参阅 MariaDB 的参数 和 Amazon RDS 的参数组。

要创建使用 MyRocks 存储引擎的表，请在 `CREATE TABLE` 语句中指定 `ENGINE=RocksDB`。以下示例创建使用 MyRocks 存储引擎的表。

```
CREATE TABLE test (a INT NOT NULL, b CHAR(10)) ENGINE=RocksDB;
```

我们强烈建议您不要运行同时跨 InnoDB 和 MyRocks 表的事务。MariaDB 不保证跨存储引擎事务的 ACID（原子性、一致性、隔离性、持久性）。尽管可以在数据库实例中同时使用 InnoDB 和 MyRocks 表，但我们不建议使用这种方法，除非在从一个存储引擎迁移到另一个存储引擎的过程中。数据库实例中同时存在 InnoDB 和 MyRocks 表时，每个存储引擎都有自己的缓冲池，这可能会导致性能降低。

MyRocks 不支持 `SERIALIZABLE` 隔离或 gap 锁定。因此，通常不能将 MyRocks 与基于语句的复制结合使用。有关更多信息，请参阅 MyRocks 和复制。

目前，您只能修改以下 MyRocks 参数：

- `rocksdb_block_cache_size`
- `rocksdb_bulk_load`

- rocksdb_bulk_load_size
- rocksdb_deadlock_detect
- rocksdb_deadlock_detect_depth
- rocksdb_max_latest_deadlocks

MyRocks 存储引擎和 InnoDB 存储引擎可根据 rocksdb_block_cache_size 和 innodb_buffer_pool_size 参数的设置争用内存。在某些情况下，您可能只打算在特定数据库实例上使用 MyRocks 存储引擎。如果是这样，我们建议将 innodb_buffer_pool_size minimal 参数设置为最小值并将 rocksdb_block_cache_size 设置为尽可能高。

您可以使用 DescribeDBLogFiles 和 DownloadDBLogFilePortion 操作来访问 MyRocks 日志文件。

有关 MyRocks 的更多信息，请参阅 MariaDB 网站上的 MyRocks。

Amazon RDS 上的 MariaDB 的缓存预热

InnoDB 高速缓存预热可为您的 MariaDB 实例提供性能收益，方式是在数据库实例关闭时保存缓冲池的当前状态，然后在数据库实例启动时从保存的信息重新加载缓冲池。此方法将不再需要缓冲池使用常规数据库进行“预热”，而是预加载具有已知常见查询的页的缓冲池。有关缓存预热的更多信息，请参阅 MariaDB 文档中的转储和还原缓冲池。

默认情况下，在 MariaDB 10.3 和更高版本的数据库实例上启用缓存预热。要启用该功能，请将数据库实例的参数组中的 innodb_buffer_pool_dump_at_shutdown 和 innodb_buffer_pool_load_at_startup 参数设置为 1。更改参数组中的这些参数值将影响使用该参数组的所有 MariaDB 数据库实例。要为特定 MariaDB 数据库实例启用缓存预热，您可能需要为这些数据库实例创建新的参数组。有关参数组的信息，请参阅Amazon RDS 的参数组。

缓存预热主要为使用标准存储的数据库实例提高性能。如果您使用 PIOPS 存储，则通常看不到显著的性能收益。

> **Important**
>
> 如果您的 MariaDB 数据库实例未正常关闭 (例如，在故障转移期间)，则缓冲池状态将不会保存到磁盘。在此情况下，MariaDB 将在重新启动数据库实例时加载任何可用的缓冲池文件。这不会造成任何损害，但还原后的缓冲池可能不会反映缓冲池在重新启动前的最新状态。要确保您拥有启动时可用于预热 缓存的缓冲池的最新状态，建议您定期“按需”转储缓冲池。您可以按需转储或加载缓冲池。

您可创建事件来定期自动转储缓冲池。例如，以下语句创建一个名为 periodic_buffer_pool_dump 的事件，该事件每小时转储一次缓冲池。

```
CREATE EVENT periodic_buffer_pool_dump
   ON SCHEDULE EVERY 1 HOUR
   DO CALL mysql.rds_innodb_buffer_pool_dump_now();
```

有关更多信息，请参阅 MariaDB 文档中的事件。

按需转储和加载缓冲池

您可以使用以下存储过程按需保存和加载 缓存：

- 要将缓冲池的当前状态转储到磁盘，请调用 mysql.rds_innodb_buffer_pool_dump_now 存储过程。
- 要从磁盘加载缓冲池的已保存状态，请调用 mysql.rds_innodb_buffer_pool_load_now 存储过程。
- 要取消正在进行的加载操作，请调用 mysql.rds_innodb_buffer_pool_load_abort 存储过程。

Amazon RDS 不支持的 MariaDB 特征

Amazon RDS 上不支持以下 MariaDB 特征：

- S3 存储引擎
- 身份验证插件 – GSSAPI
- 身份验证插件 – Unix Socket
- Amazon Key Management 加密插件
- 低于 10.6 的 MariaDB 版本的延迟复制
- InnoDB 和 Aria 的本机 MariaDB 静态加密

  您可以按照 加密 Amazon RDS 资源 中的说明启用 MariaDB 数据库实例的加密。
- HandlerSocket
- 低于 10.6 的 MariaDB 版本的 JSON 表类型
- MariaDB ColumnStore
- MariaDB Galera 集群
- 多源复制

- 低于 10.6 的 MariaDB 版本的 MyRocks 存储引擎
- 适用于低于 11.4 的 MariaDB 版本的密码验证插件 `simple_password_check` 和 `cracklib_password_check`
- Spider 存储引擎
- Sphinx 存储引擎
- TokuDB 存储引擎
- 存储特定于引擎的对象属性，如 MariaDB 文档中的引擎定义的新表/字段/索引属性中所述
- 表和表空间加密
- Hashicorp 密钥管理插件
- 并行运行两个升级

为了产生托管服务体验，Amazon RDS 不允许通过 shell 访问数据库实例，而仅限访问某些需要高级特权的系统过程和表。Amazon RDS 支持使用任何标准 SQL 客户端应用程序访问数据库实例上的数据库。Amazon RDS 不允许使用 Telnet、安全外壳（SSH）或 Windows 远程桌面连接直接从主机访问数据库实例。

Amazon RDS 上的 MariaDB 版本

对于 MariaDB，按 X.Y.Z 版本组织版本号。在 Amazon RDS 术语中，X.Y 表示主要版本，Z 是次要版本号。对于 Amazon RDS 实现，如果主要版本号更改，则将版本更改视为重大更改；例如，从版本 10.5 更改为 10.6。如果只有次要版本号发生更改（例如，从版本 10.6.14 更改为 10.6.16），则将版本更改视为次要更改。

主题

Amazon RDS 上支持的 MariaDB 次要版本

Amazon RDS 目前支持 MariaDB 的以下次要版本。

> Note
>
> 只有月份和年份的发布日期均为大概日期，这些日期会在确切日期明确后进行更新。

下表显示了 Amazon RDS 当前支持的 MariaDB 11.4 的次要版本。

MariaDB 引擎版本	社区发布日期	RDS 发布日期	RDS 标准支持终止日期
11.4.7	2025 年 5 月 22 日	2025 年 6 月 4 日	2026 年 9 月
11.4.5	2025 年 2 月 4 日	2025 年 2 月 24 日	2026 年 3 月
11.4.4	2024 年 11 月 1 日	2024 年 12 月 20 日	2026 年 3 月

MariaDB 引擎版本	社区发布日期	RDS 发布日期	RDS 标准支持终止日期
11.4.3	2024 年 8 月 8 日	2024 年 10 月 15 日	2026 年 3 月

下表显示了 Amazon RDS 当前支持的 MariaDB 10.11 的次要版本。

MariaDB 引擎版本	社区发布日期	RDS 发布日期	RDS 标准支持终止日期
10.11.13	2025 年 5 月 22 日	2025 年 6 月 4 日	2026 年 9 月
10.11.11	2025 年 2 月 4 日	2025 年 2 月 24 日	2026 年 3 月
10.11.10	2024 年 11 月 1 日	2024 年 12 月 20 日	2026 年 3 月
10.11.9	2024 年 8 月 8 日	2024 年 9 月 4 日	2026 年 3 月
10.11.8	2024 年 5 月 16 日	2024 年 6 月 14 日	2025 年 9 月

下表显示了 Amazon RDS 当前支持的 MariaDB 10.6 的次要版本。

MariaDB 引擎版本	社区发布日期	RDS 发布日期	RDS 标准支持终止日期
10.6.22	2025 年 5 月 6 日	2025 年 5 月 20 日	2026 年 7 月
10.6.21	2025 年 2 月 4 日	2025 年 2 月 24 日	2026 年 3 月
10.6.20	2024 年 11 月 1 日	2024 年 12 月 20 日	2026 年 3 月
10.6.19	2024 年 8 月 8 日	2024 年 9 月 4 日	2026 年 3 月
10.6.18	2024 年 5 月 16 日	2024 年 6 月 14 日	2025 年 9 月

下表显示了 Amazon RDS 当前支持的 MariaDB 10.5 的次要版本。

MariaDB 引擎版本	社区发布日期	RDS 发布日期	RDS 标准支持终止日期
10.5.29	2025 年 5 月 6 日	2025 年 5 月 20 日	2026 年 2 月
10.5.28	2025 年 2 月 4 日	2025 年 2 月 24 日	2026 年 2 月
10.5.27	2024 年 11 月 1 日	2024 年 12 月 20 日	2026 年 2 月
10.5.26	2024 年 8 月 8 日	2024 年 9 月 4 日	2025 年 6 月
10.5.25	2024 年 5 月 16 日	2024 年 6 月 14 日	2025 年 6 月

下表显示了 Amazon RDS 当前支持的 MariaDB 10.4 的次要版本。

MariaDB 引擎版本	社区发布日期	RDS 发布日期	RDS 标准支持终止日期
10.4.34	2024 年 5 月 16 日	2024 年 6 月 14 日	2025 年 2 月 28 日
10.4.33	2024 年 2 月 7 日	2024 年 2 月 26 日	2025 年 2 月 28 日
10.4.32	2023 年 11 月 13 日	2023 年 12 月 12 日	2025 年 2 月 28 日
10.4.31	2023 年 8 月 14 日	2023 年 9 月 7 日	2025 年 2 月 28 日
10.4.30	2023 年 6 月 7 日	2023 年 6 月 22 日	2025 年 2 月 28 日
10.4.29	2023 年 5 月 10 日	2023 年 6 月 15 日	2025 年 2 月 28 日

在创建新数据库实例时，您可以指定当前支持的任何 MariaDB 版本。您可以指定主要版本（例如，MariaDB 10.5），以及指定主要版本支持的任何次要版本。如果没有指定版本，Amazon RDS 会默认使用支持的版本，通常是最新的版本。如果指定了主要版本但未指定次要版本，Amazon RDS 将默认使用您指定的主要版本的最新版本。要查看受支持版本的列表以及新建数据库实例的默认值，请使用 `describe-db-engine-versions` Amazon CLI 命令。

例如，要列出 RDS for MariaDB 支持的引擎版本，请运行以下 CLI 命令：

```
aws rds describe-db-engine-versions --engine mariadb --query "*[].
{Engine:Engine,EngineVersion:EngineVersion}" --output text
```

默认的 MariaDB 版本可能因 Amazon Web Services 区域而异。要创建具有特定次要版本的数据库实例，请在数据库实例创建过程中指定次要版本。可以通过运行以下 Amazon CLI 命令来确定 Amazon Web Services 区域的默认次要版本：

```
aws rds describe-db-engine-versions --default-only --engine mariadb
 --engine-version major_engine_version --region region --query "*[].
{Engine:Engine,EngineVersion:EngineVersion}" --output text
```

将 *major_engine_version* 替换为主要引擎版本，并将 *region* 替换为 Amazon Web Services 区域。例如，以下 Amazon CLI 命令返回 10.5 主要版本和美国西部（俄勒冈州）Amazon Web Services 区域 (us-west-2) 的默认 MariaDB 次要引擎版本：

```
aws rds describe-db-engine-versions --default-only --engine mariadb --engine-version
 10.5 --region us-west-2 --query "*[].{Engine:Engine,EngineVersion:EngineVersion}" --
output text
```

Amazon RDS 上的 MariaDB 次要版本

次要版本

- MariaDB 版本 11.4.7
- MariaDB 版本 11.4.5
- MariaDB 版本 11.4.4
- MariaDB 版本 10.11.13
- MariaDB 版本 10.11.11
- MariaDB 版本 10.11.10
- MariaDB 版本 10.6.22
- MariaDB 版本 10.6.21
- MariaDB 版本 10.6.20
- MariaDB 版本 10.5.29
- MariaDB 版本 10.5.28
- MariaDB 版本 10.5.27

MariaDB 版本 11.4.7

MariaDB 版本 11.4.7 现在可在 Amazon RDS 上使用。此版本包含由 MariaDB 社区和 Amazon RDS 添加的修复和改进。

新功能和增强功能

- 更新了时区信息，使其基于 `tzdata2025b`。

MariaDB 版本 11.4.5

MariaDB 版本 11.4.5 现在可在 Amazon RDS 上使用。此版本包含由 MariaDB 社区和 Amazon RDS 添加的修复和改进。

新功能和增强功能

- 更新了时区信息，使其基于 `tzdata2025a`。

MariaDB 版本 11.4.4

MariaDB 版本 11.4.4 现在可在 Amazon RDS 上使用。此版本包含由 MariaDB 社区和 Amazon RDS 添加的修复和改进。

新功能和增强功能

- 还原了两次导致时间点故障恢复（PITR）失败的 MariaDB 社区更改。有关更多信息，请参阅 MariaDB Server Jira 问题 MDEV-35528。

MariaDB 版本 10.11.13

MariaDB 版本 10.11.13 现在可在 Amazon RDS 上使用。此版本包含由 MariaDB 社区和 Amazon RDS 添加的修复和改进。

新功能和增强功能

- 更新了时区信息，使其基于 `tzdata2025b`。

MariaDB 版本 10.11.11

MariaDB 版本 10.11.11 现在可在 Amazon RDS 上使用。此版本包含由 MariaDB 社区和 Amazon RDS 添加的修复和改进。

新功能和增强功能

- 更新了时区信息，使其基于 `tzdata2025a`。

MariaDB 版本 10.11.10

MariaDB 版本 10.11.10 现在可在 Amazon RDS 上使用。此版本包含由 MariaDB 社区和 Amazon RDS 添加的修复和改进。

新功能和增强功能

- 还原了两次导致时间点故障恢复（PITR）失败的 MariaDB 社区更改。有关更多信息，请参阅 MariaDB Server Jira 问题 MDEV-35528。

MariaDB 版本 10.6.22

MariaDB 版本 10.6.22 现在可在 Amazon RDS 上使用。此版本包含由 MariaDB 社区和 Amazon RDS 添加的修复和改进。

新功能和增强功能

- 更新了时区信息，使其基于 `tzdata2025b`。

MariaDB 版本 10.6.21

MariaDB 版本 10.6.21 现在可在 Amazon RDS 上使用。此版本包含由 MariaDB 社区和 Amazon RDS 添加的修复和改进。

新功能和增强功能

- 更新了时区信息，使其基于 `tzdata2025a`。

MariaDB 版本 10.6.20

MariaDB 版本 10.6.20 现在可在 Amazon RDS 上使用。此版本包含由 MariaDB 社区和 Amazon RDS 添加的修复和改进。

新功能和增强功能

- 还原了两次导致时间点故障恢复（PITR）失败的 MariaDB 社区更改。有关更多信息，请参阅 MariaDB Server Jira 问题 MDEV-35528。

MariaDB 版本 10.5.29

MariaDB 版本 10.5.29 现在可在 Amazon RDS 上使用。此版本包含由 MariaDB 社区和 Amazon RDS 添加的修复和改进。

新功能和增强功能

- 更新了时区信息，使其基于 `tzdata2025b`。

MariaDB 版本 10.5.28

MariaDB 版本 10.5.28 现在可在 Amazon RDS 上使用。此版本包含由 MariaDB 社区和 Amazon RDS 添加的修复和改进。

新功能和增强功能

- 更新了时区信息，使其基于 `tzdata2025a`。

MariaDB 版本 10.5.27

MariaDB 版本 10.5.27 现在可在 Amazon RDS 上使用。此版本包含由 MariaDB 社区和 Amazon RDS 添加的修复和改进。

新功能和增强功能

- 还原了两次导致时间点故障恢复（PITR）失败的 MariaDB 社区更改。有关更多信息，请参阅 MariaDB Server Jira 问题 MDEV-35528。

Amazon RDS 上支持的 MariaDB 主要版本

RDS for MariaDB 主要版本至少会在相应社群版本的社区生命周期终止前保持可用状态。您可以参照下列日期规划您的测试和升级周期。如果 Amazon 对某个 RDS for MariaDB 版本的支持时间比原定时间更长，我们会计划更新此表格以反映较晚的日期。

> **Note**
>
> 只有月份和年份的发布日期均为大概日期，这些日期会在确切日期明确后进行更新。

您还可以运行 describe-db-major-engine-versions Amazon CLI 命令或者使用 DescribeDBMajorEngineVersions RDS API 操作，来查看有关主要引擎版本的支持日期的信息。

MariaDB 主要版本	社区发布日期	RDS 发布日期	社区生命周期终止日期	RDS 标准支持终止日期
MariaDB 11.4	2024 年 8 月 8 日	2024 年 10 月 15 日	2029 年 5 月	2029 年 5 月
MariaDB 10.11	2023 年 2 月 16 日	2023 年 8 月 21 日	2028 年 2 月 16 日	2028 年 2 月
MariaDB 10.6	2021 年 7 月 6 日	2022 年 2 月 3 日	2026 年 7 月 6 日	2026 年 7 月
MariaDB 10.5	2020 年 6 月 24 日	2021 年 1 月 21 日	2025 年 6 月 24 日	2026 年 2 月
MariaDB 10.4	2019 年 6 月 18 日	2020 年 4 月 6 日	2024 年 6 月 18 日	2025 年 2 月

使用数据库预览环境

数据库预览环境中的 RDS for MariaDB 数据库实例在功能上类似于其他 RDS for MariaDB 数据库实例。但是，您不能将数据库预览环境用于生产工作负载。

预览环境具有以下限制：

- Amazon RDS 会在创建所有数据库实例 60 天之后将其删除，包括其所有备份和快照。
- 您只能使用通用 SSD 和预置 IOPS SSD 存储。
- 您无法从 Amazon Web Services 支持 获得数据库实例的帮助。
- 您无法将数据库实例的快照复制到生产环境。

预览版支持以下选项。

- 您可以使用 db.m6i、db.r6i、db.m6g、db.m5、db.t3、db.r6g 和 db.r5 数据库实例类创建数据库实例。有关 RDS 实例类的更多信息，请参阅数据库实例类。
- 您可以同时使用单可用区和多可用区部署。
- 可以使用标准 MariaDB 转储和加载函数，从数据库预览环境中导出数据库或将数据库导入数据库预览环境。

数据库预览环境中不支持的功能

以下功能在预览环境中不可用：

- 跨区域快照复制
- 跨区域只读副本
- RDS 代理

数据库预览环境中的 MariaDB 版本 11.8

MariaDB 版本 11.8 现可在 Amazon RDS 数据库预览环境中使用。MariaDB 版本 11.8 包含一些改进，如 Changes and improvements in MariaDB 11.8 中所述。此版本还包括对向量数据类型、索引编制和搜索的支持。有关更多信息，请参阅 MariaDB 文档中的 Vector Overview。

在此版本可在所有 Amazon Web Services 区域中用于生产工作负载之前，可以使用数据库预览环境针对此版本测试您的工作负载。有关数据库预览环境的信息，请参阅the section called “数据库预览环境”。要从控制台中访问预览环境，请选择 rds-preview/。

数据库预览环境中的 MariaDB 版本 11.7

MariaDB 版本 11.7 现可在 Amazon RDS 数据库预览环境中使用。MariaDB 版本 11.7 包含一些改进，如 Changes and improvements in MariaDB 11.7 中所述。此版本还包括对向量数据类型、索引编制和搜索的支持。有关更多信息，请参阅 MariaDB 文档中的 Vector Overview。

在此版本可在所有 Amazon Web Services 区域中用于生产工作负载之前，可以使用数据库预览环境针对此版本测试您的工作负载。有关数据库预览环境的信息，请参阅the section called “数据库预览环境”。要从控制台中访问预览环境，请选择 rds-preview/。

数据库预览环境中的 MariaDB 版本 11.4

MariaDB 版本 11.4 现可在 Amazon RDS 数据库预览环境中使用。MariaDB 版本 11.4 包含一些改进，如 Changes and improvements in MariaDB 11.4 中所述。在此版本可在所有 Amazon Web Services 区域中用于生产工作负载之前，可以使用数据库预览环境针对此版本测试您的工作负载。

有关数据库预览环境的信息，请参阅the section called “数据库预览环境”。要从控制台中访问预览环境，请选择 rds-preview/。

Amazon RDS for MariaDB 已弃用的版本

Amazon RDS for MariaDB 版本 10.0、10.1、10.2 和 10.3 已弃用。

有关 Amazon RDS for MariaDB 的弃用策略的信息，请参阅 Amazon RDS 常见问题。

连接到 MariaDB 数据库实例

Amazon RDS 配置数据库实例后，您就可以使用标准 MariaDB 客户端应用程序或实用程序与该实例连接。在连接字符串中，可以将数据库实例端点中的域名系统 (DNS) 地址指定为主机参数。您还可以将数据库实例端点中的端口号指定为端口参数。

可以使用 MySQL 命令行客户端等工具连接 Amazon RDS for MariaDB 数据库实例。有关使用 MySQL 命令行的更多信息，请参阅 MariaDB 文档中的 mysql 命令行客户端。您可以使用 Heidi 这款基于 GUI 的应用程序进行连接。有关更多信息，请参阅下载 HeidiSQL 页面。有关安装 MySQL（包括 MySQL 命令行客户端）的信息，请参阅安装和升级 MySQL。

大多数 Linux 发行版包括 MariaDB 客户端，而不是 Oracle MySQL 客户端。要在 Amazon Linux 2023 上安装 MySQL 命令行客户端，请运行以下命令：

```
sudo dnf install mariadb105
```

要在 Amazon Linux 2 上安装 MySQL 命令行客户端，请运行以下命令：

```
sudo yum install mariadb
```

要在大多数基于 DEB 的 Linux 发行版上安装 MySQL 命令行客户端，请运行以下命令。

```
apt-get install mariadb-client
```

要检查 MySQL 命令行客户端的版本，请运行以下命令。

```
mysql --version
```

要阅读当前客户端版本的 MySQL 文档，请运行以下命令。

```
man mysql
```

要从基于 Amazon VPC 的 Virtual Private Cloud (VPC) 外部连接到数据库实例，该数据库实例必须可公开访问。此外，必须使用数据库实例安全组的入站规则授予访问权限，并且必须满足其他要求。有关更多信息，请参阅“无法连接到 Amazon RDS 数据库实例”。

可以在连接到 MariaDB 数据库实例时使用 SSL 加密功能。有关信息，请参阅Amazon RDS 上 MariaDB 数据库实例的 SSL/TLS 支持。

要查找并连接到 RDS for MariaDB 数据库实例，请参阅以下主题。

主题

查找 MariaDB 数据库实例的连接信息

数据库实例的连接信息包括其端点、端口和有效数据库用户，如主用户。例如，假设端点值为 `mydb.123456789012.us-east-1.rds.amazonaws.com`。在这种情况下，端口值是 `3306`，数据库用户是 `admin`。鉴于此信息，您可以在连接字符串中指定以下值：

- 对于主机或主机名或 DNS 名称，请指定 `mydb.123456789012.us-east-1.rds.amazonaws.com`。
- 对于端口，请指定 `3306`。
- 对于用户，请指定 `admin`。

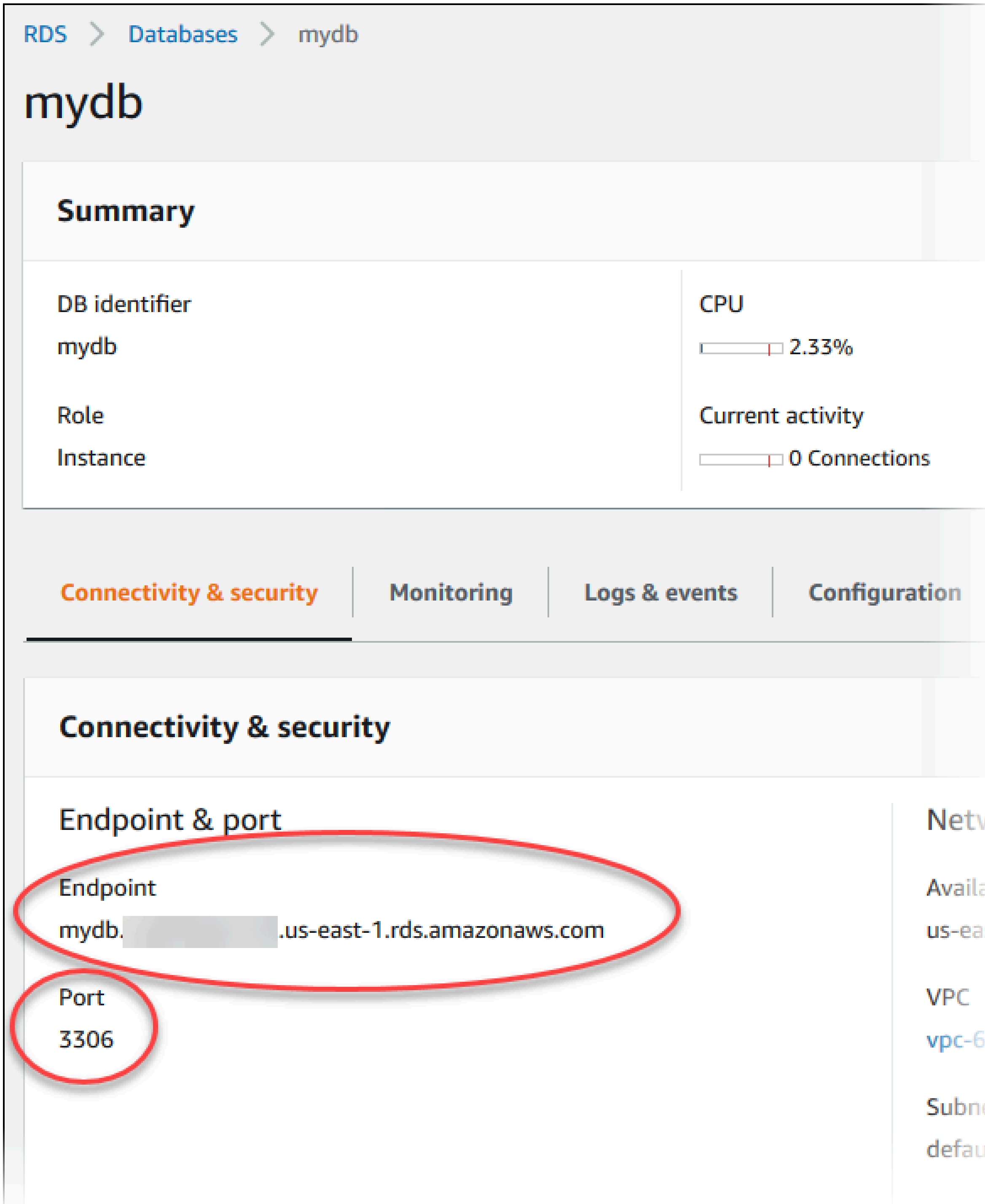
要连接到数据库实例，请使用 MariaDB 数据库引擎的任何客户端。例如，您可以使用 MySQL 命令行客户端或 MySQL Workbench。

要查找数据库实例的连接信息，您可以使用Amazon Web Services Management Console、Amazon Command Line Interface（Amazon CLI）[describe-db-instances] 命令，或 Amazon RDS API [DescribeDBInstances] 操作列出其详细信息。

控制台

在 Amazon Web Services Management Console 中查找数据库实例的连接信息

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 数据库 以显示数据库实例的列表。
3. 选择 MariaDB 数据库实例名称以显示其详细信息。
4. 在连接和安全性选项卡上，复制端点。另请注意端口号。您需要端点和端口号才能连接到数据库实例。



5. 如果需要查找主用户名，请选择 Configuration (配置) 选项卡并查看 Master username (主用户名) 值。

Amazon CLI

要使用 Amazon CLI 查找 MariaDB 数据库实例的相关连接信息，请运行 describe-db-instances 命令。在调用中，查询数据库实例 ID、端点、端口和主用户名。

对于 Linux、macOS 或 Unix：

```
aws rds describe-db-instances \
  --filters "Name=engine,Values=mariadb" \
  --query "*[].[DBInstanceIdentifier,Endpoint.Address,Endpoint.Port,MasterUsername]"
```

对于 Windows：

```
aws rds describe-db-instances ^
  --filters "Name=engine,Values=mariadb" ^
  --query "*[].[DBInstanceIdentifier,Endpoint.Address,Endpoint.Port,MasterUsername]"
```

您的输出应类似于以下内容。

```
[
    [
        "mydb1",
        "mydb1.123456789012.us-east-1.rds.amazonaws.com",
        3306,
        "admin"
    ],
    [
        "mydb2",
        "mydb2.123456789012.us-east-1.rds.amazonaws.com",
        3306,
        "admin"
    ]
]
```

RDS API

要使用 Amazon RDS API 查找数据库实例的连接信息，请调用 DescribeDBInstances 操作。在输出中，找到端点地址、端点端口和主用户名的值。

从 RDS for MariaDB 的 MySQL 命令行客户端进行连接（未加密）

> **Important**
>
> 仅当客户端和服务器位于同一 VPC 中，并且网络受信任时，才会使用未加密的 MySQL 连接。有关使用加密连接的信息，请参阅 从 MySQL 命令行客户端使用 SSL/TLS 连接到 Amazon RDS 上的 MariaDB 数据库实例（已加密）。

要使用 MySQL 命令行客户端连接到数据库实例，请在客户端计算机上的命令提示符处输入以下命令。执行此操作时，您将连接到 MariaDB 数据库实例上的数据库。将 *<endpoint>* 替换为数据库实例的 DNS 名称（端点）；并替换用于 *<mymasteruser>* 的主用户名。提示输入密码时，提供所使用的主密码。

```
mysql -h <endpoint> -P 3306 -u <mymasteruser> -p
```

在输入用户的密码后，您将看到类似于以下内容的输出。

```
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 31
Server version: 10.6.10-MariaDB-log Source distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

使用 Amazon JDBC 驱动程序和 Amazon Python 驱动程序连接到 RDS for MariaDB；

使用 Amazon JDBC 驱动程序和 Amazon Python 驱动程序连接到 RDS for MariaDB 数据库实例。有关更多信息，请参阅以下主题。

主题

- 使用 Amazon Web Services（Amazon）JDBC 驱动程序连接到 RDS for MariaDB
- 使用 Amazon Web Services（Amazon）Python 驱动程序连接到 RDS for MariaDB

使用 Amazon Web Services（Amazon）JDBC 驱动程序连接到 RDS for MariaDB

Amazon Web Services（Amazon）JDBC 驱动程序已重新设计为高级 JDBC 包装器。此包装器是对现有 JDBC 驱动程序各项功能的补充和扩展。该驱动程序与社区 MySQL Connector/J 驱动程序和社区 MariaDB Connector/J 驱动程序兼容。

要安装 Amazon JDBC 驱动程序，请附加 Amazon JDBC 驱动程序.jar 文件（位于应用程序 `CLASSPATH` 中），并保留对相应社区驱动程序的引用。按如下方式更新相应的连接 URL 前缀：

- `jdbc:mysql://` 到 `jdbc:aws-wrapper:mysql://`
- `jdbc:mariadb://` 到 `jdbc:aws-wrapper:mariadb://`

有关 Amazon JDBC 驱动程序的更多信息及其完整使用说明，请参阅 Amazon Web Services (Amazon) JDBC Driver GitHub repository。

使用 Amazon Web Services（Amazon）Python 驱动程序连接到 RDS for MariaDB

Amazon Web Services（Amazon）驱动程序设计为高级 Python 包装器。这款包装器是对开源 Psycopg 驱动程序的各项功能的补充和扩展。Amazon Python 驱动程序支持 Python 3.8 及更高版本。您可以使用 `pip` 命令和 `psycopg` 开源软件包安装 `aws-advanced-python-wrapper` 程序包。

有关 Amazon JDBC 驱动程序的更多信息及其完整使用说明，请参阅 Amazon Web Services（Amazon）Python 驱动程序 GitHub 存储库。

排除与 MariaDB 数据库实例的连接故障

导致新数据库实例连接故障的两个常见原因如下：

- 数据库实例是使用安全组创建的，而该安全组无法授权来自运行 MariaDB 应用程序或实用程序的设备或 Amazon EC2 实例的连接。数据库实例必须具有授权进行连接的 VPC 安全组。有关更多信息，请参阅 Amazon VPC 和 Amazon RDS。

  您可以在安全组中添加或编辑入站规则。对于 Source，选择 My IP。这允许从浏览器中检测到的 IP 地址访问数据库实例。
- 数据库实例是使用默认端口 3306 创建的，而您公司的防火墙规则不允许公司网络中的设备连接到该端口。要修复此故障，请重新创建使用不同端口的实例。

有关连接问题的更多信息，请参阅无法连接到 Amazon RDS 数据库实例。

保护 MariaDB 数据库实例连接安全

您可以管理 MariaDB 数据库实例的安全性。

主题

Amazon RDS 上的 MariaDB 安全性

MariaDB 数据库实例的安全性在三个级别上进行管理：

- Amazon Identity and Access Management 控制可以对数据库实例执行 Amazon RDS 管理操作的人员。使用 IAM 证书连接到Amazon时，您的 IAM 账户必须具有授予执行 Amazon RDS 管理操作所需的权限的 IAM 策略。有关更多信息，请参阅 Amazon RDS 的 Identity and Access Management。
- 创建数据库实例时，可以使用 VPC 安全组控制哪些设备和 Amazon EC2 实例能够建立到数据库实例的端点和端口的连接。可以使用安全套接字层 (SSL)/传输层安全性协议 (TLS) 建立这些连接。此外，公司的防火墙规则也可以控制公司中运行的哪些设备可以建立到数据库实例的连接。
- 建立到 MariaDB 数据库实例的连接之后，以与独立 MariaDB 实例相同的方式应用登录和权限的身份验证。`CREATE USER`、`RENAME USER`、`GRANT`、`REVOKE` 和 `SET PASSWORD` 等命令的作用与它们在独立数据库中的作用相同，就像直接修改数据库架构表。

当您创建 Amazon RDS 数据库实例时，主用户有以下默认权限：

- `alter`
- `alter routine`
- `create`
- `create routine`
- `create temporary tables`
- `create user`
- `create view`

- `delete`
- `drop`
- `event`
- `execute`
- `grant option`
- `index`
- `insert`
- `lock tables`
- `process`
- `references`
- `reload`

  该权限仅限在 MariaDB 数据库实例上使用。它不授予对 `FLUSH LOGS` 或 `FLUSH TABLES WITH READ LOCK` 操作的访问权限。
- `replication client`
- `replication slave`
- `select`
- `show create routine`

  此权限仅适用于运行 11.4 及更高版本的 MariaDB 数据库实例。
- `show databases`
- `show view`
- `trigger`
- `update`

有关这些权限的更多信息，请参阅 MariaDB 文档中的[用户账户管理]。

> **Note**
>
> 尽管您可以删除数据库实例上的主用户，但我们不建议这样做。要重新创建主用户，请使用 `ModifyDBInstance` API 或 `modify-db-instance` Amazon CLI，并用相应参数指定一个新的主用户密码。如果实例中不存在主用户，将使用指定的密码创建主用户。

要为每个数据库实例提供管理服务，需在创建数据库实例时创建 rdsadmin 用户。如果试图删掉、重命名、修改 rdsadmin 账户的密码，或者修改该账户的权限，会导致出错。

若要允许管理数据库实例，需限制标准的 kill 和 kill_query 命令。提供了 Amazon RDS 命令 mysql.rds_kill、mysql.rds_kill_query 和 mysql.rds_kill_query_id 以便在 MariaDB 和 MySQL 中使用，因此您可以结束数据库实例上的用户会话或查询。

对于 RDS for MariaDB 使用密码验证插件

从 RDS for MariaDB 版本 11.4 开始，您可以使用以下密码验证插件来增强数据库连接的安全性：

- simple_password_check – 检查密码是否包含至少特定数量的特定类型字符。
- cracklib_password_check – 检查密码是否出现在 CrackLib 库的字典文件中。

要启用这些插件，请在与数据库实例关联的数据库参数组中，将参数 simple_password_check 或 cracklib_password_check 的值设置为 FORCE_PLUS_PERMANENT。设置此值后，无法在运行时使用 UNINSTALL PLUGIN 语句卸载插件。

要禁用这些插件，请在与数据库实例关联的数据库参数组中，将参数 simple_password_check 或 cracklib_password_check 的值设置为 OFF。设置此值后，插件验证规则将不再适用于新密码。

有关在参数组中设置参数值的信息，请参阅the section called “修改数据库参数组中的参数”。

使用 SSL/TLS 加密客户端与 Amazon RDS 上 MariaDB 数据库实例的连接

安全套接字层 (SSL) 是用于保护客户端和服务器之间的网络连接的行业标准协议。在 SSL 版本 3.0 之后，此名称已更改为传输层安全性协议 (TLS)。Amazon RDS 对于 MariaDB 数据库实例支持 SSL/TLS 加密。使用 SSL/TLS，您可加密应用程序客户端和 MariaDB 数据库实例之间的连接。SSL/TLS 支持在所有 Amazon Web Services 区域中都可用。

借助 Amazon RDS，可以通过使用 SSL/TLS 加密客户端与 MariaDB 数据库实例的连接，要求所有与 MariaDB 数据库实例的连接都使用 SSL/TLS，以及使用 SSL/TLS（加密）从 MySQL 命令行客户端进行连接，从而保护传输中数据。以下各节提供了有关为 Amazon RDS 上的 MariaDB 数据库实例配置和使用 SSL/TLS 加密的指导。

主题

- Amazon RDS 上 MariaDB 数据库实例的 SSL/TLS 支持
- Amazon RDS 上 MariaDB 数据库实例的特定用户账户需要 SSL/TLS

- 所有与 Amazon RDS 上的 MariaDB 数据库实例的连接都需要 SSL/TLS
- 从 MySQL 命令行客户端使用 SSL/TLS 连接到 Amazon RDS 上的 MariaDB 数据库实例（已加密）

Amazon RDS 上 MariaDB 数据库实例的 SSL/TLS 支持

在 Amazon RDS 预置数据库实例时，Amazon RDS 创建 SSL/TLS 证书，并将该证书安装在数据库实例上。这些证书由证书颁发机构签署。SSL/TLS 证书会将数据库实例终端节点作为 SSL/TLS 证书的公用名 (CN) 包含在内以防止欺诈攻击。

有关下载证书的信息，请参阅 使用 SSL/TLS 加密与数据库实例或集群的连接。有关将 SSL/TLS 与 MySQL 一起使用的更多信息，请参阅更新应用程序以使用新的 SSL/TLS 证书连接到 MariaDB 实例。

Amazon RDS for MariaDB 支持传输层安全性协议（TLS）版本 1.3、1.2、1.1 和 1.0。TLS 支持取决于 MariaDB 次要版本。下表显示各个 MariaDB 次要版本的 TLS 支持情况。

TLS 版本	MariaDB 11.4	MariaDB 10.11	MariaDB 10.6	MariaDB 10.5	MariaDB 10.4
TLS 1.3	所有次要版本	所有次要版本	所有次要版本	所有次要版本	所有次要版本
TLS 1.2	所有次要版本	所有次要版本	所有次要版本	所有次要版本	所有次要版本
TLS 1.1	不支持	10.11.6 及更低版本	10.6.16 及更低版本	10.5.23 及更低版本	10.4.32 及更低版本
TLS 1.0	不支持	10.11.6 及更低版本	10.6.16 及更低版本	10.5.23 及更低版本	10.4.32 及更低版本

Amazon RDS 上 MariaDB 数据库实例的特定用户账户需要 SSL/TLS

对于与 Amazon RDS 上的 MariaDB 数据库实例的指定用户账户连接，可以要求进行 SSL/TLS 加密。保护敏感信息免遭未经授权的访问或拦截，对于在考虑数据机密性的情况下强制执行安全策略至关重要。

若要求特定用户的账户建立 SSL/TLS 连接，请使用以下语句之一（具体取决于 MySQL 版本），来要求用户账户 `encrypted_user` 建立 SSL/TLS 连接。

为此，请使用以下语句。

```
ALTER USER 'encrypted_user'@'%' REQUIRE SSL;
```

有关与 MariaDB 的 SSL/TLS 连接的更多信息，请参阅 MariaDB 文档中的保护客户端和服务器的连接。

所有与 Amazon RDS 上的 MariaDB 数据库实例的连接都需要 SSL/TLS

使用 `require_secure_transport` 参数要求与 MariaDB 数据库实例的所有用户连接都使用 SSL/TLS。默认情况下，`require_secure_transport` 参数设置为 OFF。您可将 `require_secure_transport` 参数设置为 ON 以要求使用 SSL/TLS 连接到数据库实例。

> **Note**
>
> 只有 MariaDB 版本 10.5 及更高版本才支持 `require_secure_transport` 参数。

您可通过更新数据库实例的数据库参数组来设置 `require_secure_transport` 参数值。您无需重启数据库实例即可使更改生效。

当数据库实例的 `require_secure_transport` 参数设置为 ON 时，如果数据库客户端能够建立加密连接，则可以连接到该数据库实例。否则，将向客户端返回类似于以下内容的错误消息：

```
ERROR 1045 (28000): Access denied for user 'USER'@'localhost' (using password: YES |
 NO)
```

有关设置参数的信息，请参阅 在 Amazon RDS 中修改数据库参数组中的参数。

有关 `require_secure_transport` 参数的更多信息，请参阅 MariaDB 文档。

从 MySQL 命令行客户端使用 SSL/TLS 连接到 Amazon RDS 上的 MariaDB 数据库实例（已加密）

如果您使用的是 MySQL 5.7 版本、MySQL 8.0 版本或 MariaDB 版本，则 `mysql` 客户端程序参数略有不同。

要查看使用的版本，请使用 `--version` 选项运行 `mysql` 命令。在以下示例中，结果显示客户端程序来自 MariaDB。

```
$ mysql --version
mysql  Ver 15.1 Distrib 10.5.15-MariaDB, for osx10.15 (x86_64) using readline 5.1
```

大多数 Linux 发行版（例如 Amazon Linux、CentOS、SUSE 和 Debian）都用 MariaDB 取代了 MySQL，mysql 版本来自 MariaDB。

要使用 SSL/TLS 连接到数据库实例，请执行以下步骤：

使用 MySQL 命令行客户端连接到使用 SSL/TLS 加密的数据库实例

1. 下载适用于所有 Amazon Web Services 区域的根证书。

   有关下载证书的信息，请参阅 使用 SSL/TLS 加密与数据库实例或集群的连接。

2. 使用 MySQL 命令行客户端，通过 SSL/TLS 加密连接到数据库实例。对于 -h 参数，替换为数据库实例的 DNS 名称（端点）。对于 --ssl-ca 参数，替换为 SSL/TLS 证书文件名。对于 -P 参数，替换为数据库实例的端口。对于 -u 参数，替换为有效数据库用户（如主用户）的用户名。根据系统提示输入主用户密码。

   以下示例说明了如何在 MariaDB 客户端和更高版本中使用 --ssl-ca 参数启动客户端。

```
mysql -h myinstance.123456789012.rds-cn-north-1.amazonaws.com --ssl-ca=global-bundle.pem --ssl -P 3306 -u myadmin -p
```

   如果要求 SSL/TLS 连接对照 SSL/TLS 证书中的端点验证数据库实例端点，请输入以下命令：

```
mysql -h myinstance.123456789012.rds-cn-north-1.amazonaws.com --ssl-ca=global-bundle.pem --ssl-verify-server-cert -P 3306 -u myadmin -p
```

   以下示例说明了如何在 MySQL 5.7 和更高版本中使用 --ssl-ca 参数启动客户端。

```
mysql -h myinstance.123456789012.rds-cn-north-1.amazonaws.com --ssl-ca=global-bundle.pem --ssl-mode=REQUIRED -P 3306 -u myadmin -p
```

3. 根据系统提示输入主用户密码。

您应该可以看到类似于如下所示的输出内容。

```
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 31
Server version: 10.6.10-MariaDB-log Source distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.
```

```
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

更新应用程序以使用新的 SSL/TLS 证书连接到 MariaDB 实例

自 2023 年 1 月 13 日起，Amazon RDS 发布了新的证书颁发机构（CA）证书，以便使用安全套接字层或传输层安全性协议（SSL/TLS）连接到 RDS 数据库实例。接下来，您可以找到有关更新应用程序以使用新证书的信息。

本主题可以帮助您确定应用程序是否需要证书验证才能连接到数据库实例。

> **Note**
>
> 某些应用程序配置为仅在它们可以成功验证服务器上的证书时才连接到 MariaDB。对于此类应用程序，您必须更新客户端应用程序信任存储，以包括新的 CA 证书。
> 您可以指定以下 SSL 模式：`disabled`、`preferred` 和 `required`。当您使用 `preferred` SSL 模式且 CA 证书不存在或不是最新时，连接会恢复为不使用 SSL，且仍能成功连接。
> 不建议使用 `preferred` 模式。在 `preferred` 模式下，如果连接遇到无效证书，则会停止使用加密并在未加密的状态下继续。

更新客户端应用程序信任存储中的 CA 证书后，可以在数据库实例上轮换这些证书。强烈建议在生产环境中实现这些过程之前，先在开发或测试环境中测试它们。

有关证书轮换的更多信息，请参阅轮换 SSL/TLS 证书。有关下载证书的更多信息，请参阅使用 SSL/TLS 加密与数据库实例或集群的连接。有关对 MariaDB 数据库实例使用 SSL/TLS 的信息，请参阅Amazon RDS 上 MariaDB 数据库实例的 SSL/TLS 支持。

主题

- 确定客户端是否需要证书验证才能连接
- 更新应用程序信任存储
- 用于建立 SSL 连接的示例 Java 代码

确定客户端是否需要证书验证才能连接

您可以检查 JDBC 客户端和 MySQL 客户端是否需要证书验证才能连接。

JDBC

以下使用 MySQL Connector/J 8.0 的示例显示了一种方法，用于检查应用程序的 JDBC 连接属性以确定成功的连接是否需要有效证书。有关 MySQL 的所有 JDBC 连接选项的更多信息，请参阅 MySQL 文档中的配置属性。

当使用 MySQL Connector/J 8.0 时，如果您的连接属性将 sslMode 设置为 VERIFY_CA 或 VERIFY_IDENTITY，则 SSL 连接需要对服务器 CA 证书进行验证，如以下示例所示。

```
Properties properties = new Properties();
properties.setProperty("sslMode", "VERIFY_IDENTITY");
properties.put("user", DB_USER);
properties.put("password", DB_PASSWORD);
```

> **Note**
> 如果您使用 MySQL Java Connector v5.1.38 或更高版本或者 MySQL Java Connector v8.0.9 或更高版本连接到数据库，即使您没有明确配置应用程序在连接到数据库时使用 SSL/TLS，这些客户端驱动程序仍默认为使用 SSL/TLS。此外，在使用 SSL/TLS 时，它们会执行部分证书验证，如果数据库服务器证书过期，则无法连接。
> 作为安全最佳实践，请指定除此处所示提示以外的密码。

MySQL

以下使用 MySQL 客户端的示例显示了两种方法，用于检查脚本的 MySQL 连接以确定成功的连接是否需要有效证书。有关 MySQL 客户端的所有连接选项的更多信息，请参阅 MySQL 文档中的加密连接的客户端配置。

当使用 MySQL 5.7 或 MySQL 8.0 客户端时，如果对于 --ssl-mode 选项，您指定 VERIFY_CA 或 VERIFY_IDENTITY，则 SSL 连接需要对服务器 CA 证书进行验证，如以下示例所示。

```
mysql -h mysql-database.rds.amazonaws.com -uadmin -ppassword --ssl-ca=/tmp/ssl-cert.pem
 --ssl-mode=VERIFY_CA
```

当使用 MySQL 5.6 客户端时，如果您指定 --ssl-verify-server-cert 选项，则 SSL 连接需要对服务器 CA 证书进行验证，如以下示例所示。

```
mysql -h mysql-database.rds.amazonaws.com -uadmin -ppassword --ssl-ca=/tmp/ssl-cert.pem
 --ssl-verify-server-cert
```

更新应用程序信任存储

有关更新 MySQL 应用程序的信任存储的信息，请参阅 MariaDB 文档中的将 TLS/SSL 与 MariaDB Connector/J 结合使用。

有关下载根证书的信息，请参阅 使用 SSL/TLS 加密与数据库实例或集群的连接。

有关导入证书的示例脚本，请参阅 将证书导入信任存储的示例脚本。

> **Note**
>
> 更新信任存储时，除了添加新证书外，还可以保留较旧证书。

如果在应用程序中使用 MariaDB Connector/J JDBC 驱动程序，请在该应用程序中设置以下属性。

```
System.setProperty("javax.net.ssl.trustStore", certs);
System.setProperty("javax.net.ssl.trustStorePassword", "password");
```

启动应用程序时，请设置以下属性。

```
java -Djavax.net.ssl.trustStore=/path_to_truststore/MyTruststore.jks -
Djavax.net.ssl.trustStorePassword=my_truststore_password com.companyName.MyApplication
```

> **Note**
>
> 作为安全最佳实践，请指定除此处所示提示以外的密码。

用于建立 SSL 连接的示例 Java 代码

以下代码说明如何设置使用 JDBC 的 SSL 连接。

```
private static final String DB_USER = "admin";

        private static final String DB_USER = "user name";
        private static final String DB_PASSWORD = "password";
        // This key store has only the prod root ca.
        private static final String KEY_STORE_FILE_PATH = "file-path-to-keystore";
        private static final String KEY_STORE_PASS = "keystore-password";

    public static void main(String[] args) throws Exception {
        Class.forName("org.mariadb.jdbc.Driver");

        System.setProperty("javax.net.ssl.trustStore", KEY_STORE_FILE_PATH);
        System.setProperty("javax.net.ssl.trustStorePassword", KEY_STORE_PASS);

        Properties properties = new Properties();
        properties.put("user", DB_USER);
        properties.put("password", DB_PASSWORD);


        Connection connection = DriverManager.getConnection("jdbc:mysql://ssl-mariadb-
public.cni62e2e7kwh.us-east-1.rds.amazonaws.com:3306?useSSL=true",properties);
        Statement stmt=connection.createStatement();

        ResultSet rs=stmt.executeQuery("SELECT 1 from dual");

        return;
    }
```

> **Important**
>
> 在确定了数据库连接使用 SSL/TLS 并更新了应用程序信任存储之后，可以更新数据库以使用 rds-ca-rsa2048-g1 证书。有关说明，请参阅 通过修改数据库实例或集群来更新 CA 证书 中的步骤 3。
>
> 作为安全最佳实践，请指定除此处所示提示以外的密码。

使用 Amazon RDS 优化读取提高 RDS for MariaDB 的查询性能

使用 Amazon RDS 优化读取，您可以更快地处理 RDS for MariaDB 查询。与不使用 RDS 优化读取的数据库实例相比，使用 RDS 优化读取的 RDS for MariaDB 数据库实例可以实现最多快 2 倍的查询处理速度。

主题

RDS 优化读取概述

当您使用开启了 RDS 优化读取的 RRDS for MariaDB 数据库实例时，您的数据库实例通过使用实例存储实现更快的查询性能。实例存储为数据库实例提供临时性块级存储。存储位于物理上附加到主机服务器的非易失性存储规范（NVMe）固态硬盘（SSD）上。此存储针对低延迟、高随机输入/输出性能和高速连续读取吞吐量进行了优化。

默认情况下，当数据库实例使用带有实例存储的数据库实例类（例如 db.m5d 或 db.m6gd）时，RDS 优化读取处于开启状态。使用 RDS 优化读取，一些临时对象存储在实例存储中。这些临时对象包括内部临时文件、内部磁盘上的临时表、内存映射文件和二进制日志（binlog）缓存文件。有关实例存储的更多信息，请参阅《适用于 Linux 实例的 Amazon Elastic Compute Cloud 用户指南》中的 Amazon EC2 实例存储。

在 MariaDB 中生成用于查询处理的临时对象的工作负载可以利用实例存储来加快查询处理速度。此类工作负载包括涉及排序、哈希聚合、高负载联接、公用表表达式（CTE）的查询以及对未索引列的查询。无论用于持久 Amazon EBS 存储的存储配置如何，这些实例存储卷都能提供更高的 IOPS 和性能。由于 RDS 优化读取将对临时对象的操作分载到实例存储，因此永久存储（Amazon EBS）的每秒进行读写操作的次数（IOPS）或吞吐量现在可用于对永久对象进行的操作。这些操作包括常规数据文件读取和写入以及后台引擎操作，例如刷新和插入缓冲区合并。

> **Note**
> 手动和自动 RDS 快照仅包含用于永久对象的引擎文件。在实例存储中创建的临时对象不包含在 RDS 快照中。

RDS 优化读取的使用案例

如果您的工作负载严重依赖于临时对象（例如内部表或文件）来执行查询，那么启用 RDS 优化读取会给您带来好处。以下使用案例是 RDS 优化读取的候选使用案例：

- 使用复杂的公用表表达式（CTE）、派生表和分组操作运行分析查询的应用程序
- 通过未优化的查询提供大量读取流量的只读副本
- 运行涉及复杂操作（例如，带 `GROUP BY` 和 `ORDER BY` 子句的查询）的按需或动态报告查询的应用程序
- 使用内部临时表进行查询处理的工作负载

  您可以监控引擎状态变量 `created_tmp_disk_tables`，以确定在数据库实例上创建的基于磁盘的临时表的数量。
- 直接或在过程中创建大型临时表以存储中间结果的应用程序
- 对非索引列执行分组或排序的数据库查询

RDS 优化读取的最佳实践

对于 RDS 优化读取使用以下最佳实践：

- 为只读查询添加重试逻辑，以防它们在执行过程中由于实例存储已满而失败。
- 使用 CloudWatch 指标 `FreeLocalStorage` 监控实例存储上的可用存储空间。如果由于数据库实例上的工作负载导致实例存储达到其极限，请修改数据库实例以使用更大的数据库实例类。
- 当您的数据库实例有足够的内存但仍达到实例存储上的存储限制时，增加 `binlog_cache_size` 值以将特定于会话的二进制日志条目保留在内存中。此配置可防止将二进制日志条目写入磁盘上的临时二进制日志缓存文件。

  `binlog_cache_size` 参数是特定于会话的。您可以更改每个新会话的值。此参数的设置可以提高高峰工作负载期间数据库实例的内存利用率。因此，请考虑根据应用程序的工作负载模式和数据库实例上的可用内存增加参数值。

- 对 binlog_format 使用 MIXED 的默认值。根据事务的大小，将 binlog_format 设置为 ROW 可能会导致实例存储上的二进制日志缓存文件很大。
- 避免在单个事务中进行批量更改。这些类型的事务会在实例存储上生成大型二进制日志缓存文件，并在实例存储已满时导致问题。考虑将写入分成多个小事务，以最大限度地减少二进制日志缓存文件的存储使用量。

使用 RDS 优化读取

当您在单可用区数据库实例部署或多可用区数据库实例部署中预调配具有以下数据库实例类之一的 RDS for MariaDB 数据库实例时，该数据库实例会自动使用 RDS 优化型读取功能。

要启用 RDS 优化读取，请执行以下操作之一：

- 使用其中一个数据库实例类创建 RDS for MariaDB 数据库实例。有关更多信息，请参阅 创建 Amazon RDS 数据库实例。
- 修改现有 RDS for MariaDB 数据库实例以使用其中一个数据库实例类。有关更多信息，请参阅 修改 Amazon RDS 数据库实例。

在支持其中一个或多个数据库实例类（具有本地 NVMe SSD 存储）的所有 Amazon Web Services 区域中，均可使用 RDS 优化型读取功能。有关数据库实例类的信息，请参阅 the section called “数据库实例类”。

数据库实例类可用性因 Amazon Web Services 区域而异。要确定特定 Amazon Web Services 区域中是否支持数据库实例类，请参阅the section called “确定 Amazon Web Services 区域 中的数据库实例类支持”。

如果您不想使用 RDS 优化读取，请修改您的数据库实例，使其不使用支持该特征的数据库实例类。

监控使用 RDS 优化读取的数据库实例

您可以使用以下 CloudWatch 指标监控使用 RDS 优化读取的数据库实例：

- FreeLocalStorage
- ReadIOPSLocalStorage
- ReadLatencyLocalStorage
- ReadThroughputLocalStorage
- WriteIOPSLocalStorage

- `WriteLatencyLocalStorage`
- `WriteThroughputLocalStorage`

这些指标提供有关可用实例存储的存储空间、IOPS 和吞吐量的数据。有关这些指标的更多信息，请参阅 Amazon RDS 的 Amazon CloudWatch 实例级指标。

RDS 优化读取的限制

以下限制适用于 RDS 优化读取：

- 以下 RDS for MariaDB 版本支持 RDS 优化读取：
  - 10.11.4 及更高的 10.11 版本
  - 10.6.7 和更高的 10.6 版本
  - 10.5.16 和更高的 10.5 版本
  - 10.4.25 和更高的 10.4 版本

  有关 RDS for MariaDB 版本的信息，请参阅 Amazon RDS 上的 MariaDB 版本。
- 您无法在支持 RDS 优化读取的数据库实例类上将临时对象的位置更改为永久存储（Amazon EBS）。
- 在数据库实例上启用二进制日志记录时，最大事务大小受实例存储大小的限制。在 MariaDB 中，任何需要的存储空间比 `binlog_cache_size` 值更多的会话都会将事务更改写入临时二进制日志缓存文件，这些文件是在实例存储上创建的。
- 实例存储已满时，事务可能会失败。

使用适用于 MariaDB 的 Amazon RDS 优化型写入功能提高写入性能

您可以使用适用于 MariaDB 的 RDS 优化写入功能来提高写入事务的性能。当 RDS for MariaDB 数据库使用 RDS 优化型写入功能时，它最多可以实现高两倍的写入事务吞吐量。

主题

RDS 优化写入概述

当您开启 RDS 优化写入功能时，您的 RDS for MariaDB 数据库在将数据刷新到持久存储时仅写入一次，无需双写缓冲区。数据库继续为可靠的数据库事务提供 ACID 属性保护，同时提高性能。

关系数据库（如 MariaDB）为可靠的数据库事务提供原子性、一致性、隔离性和持久性的 ACID 属性。为了帮助提供这些属性，MariaDB 使用一个名为双写缓冲区的数据存储区域，以防止出现不完整页面写入错误。这些错误发生在数据库更新页面时出现硬件故障的情况下，例如停电时。MariaDB 数据库可以检测不完整页面写入，并使用双写缓冲区中的页面副本进行恢复。虽然这种技术提供了保护，但也会导致额外的写入操作。有关 MariaDB 双写缓冲区的更多信息，请参阅 MariaDB 文档中的 InnoDB 双写缓冲区。

开启 RDS 优化型写入功能后，RRDS for MariaDB 数据库将在数据刷新到持久存储时仅写入一次，而不使用双写缓冲区。如果您在 RDS for MariaDB 数据库上运行写入密集型工作负载，RDS 优化型写入功能非常有用。具有写入密集型工作负载的数据库示例包括支持数字支付、金融交易和游戏应用程序的数据库。

这些数据库在使用 Amazon Nitro 系统的数据库实例类上运行。由于这些系统中的硬件配置，数据库可以在一个步骤中可靠而持久地将 16KiB 页直接写入数据文件。Amazon Nitro 系统使 RDS 优化写入成为可能。

您可以设置新的数据库参数 `rds.optimized_writes` 来控制适用于 RDS for MariaDB 数据库的 RDS 优化型写入特征。在以下版本的 RDS for MariaDB 的数据库参数组中访问此参数：

- 11.4.3 及更高的 11.4 版本

- 10.11.4 及更高的 10.11 版本
- 10.6.10 和更高的 10.6 版本

使用以下值设置参数：

- AUTO – 如果数据库支持 RDS 优化写入，则开启此功能。如果数据库不支持 RDS 优化写入，请将其关闭。此设置是默认设置。
- OFF – 关闭 RDS 优化写入，即使数据库支持此功能也不例外。

如果您将配置为使用 RDS 优化型写入特征的 RDS for MariaDB 数据库迁移到不支持该特征的数据库实例类，RDS 会自动对该数据库关闭 RDS 优化型写入特征。

关闭 RDS 优化型写入功能时，数据库使用 MariaDB 双写缓冲区。

要确定 RDS for MariaDB 数据库是否使用 RDS 优化型写入功能，请查看数据库的 `innodb_doublewrite` 参数的当前值。如果数据库使用 RDS 优化写入，则此参数设置为 `FALSE`（`0`）。

使用 RDS 优化写入

使用 RDS 控制台、Amazon CLI 或 RDS API 创建 RDS for MariaDB 数据库时，可以开启 RDS 优化型写入功能。在数据库创建过程中，当以下两个条件都适用时，RDS 优化写入功能会自动开启：

- 您可以指定支持 RDS 优化写入的数据库引擎版本和数据库实例类。
  - 以下 RDS for MariaDB 版本支持 RDS 优化型写入功能：
    - 11.4.3 及更高的 11.4 版本
    - 10.11.4 及更高的 10.11 版本
    - 10.6.10 和更高的 10.6 版本

    有关 RDS for MariaDB 版本的信息，请参阅 Amazon RDS 上的 MariaDB 版本。
  - 使用以下数据库实例类的 RDS for MariaDB 数据库支持 RDS 优化型写入功能：
    - db.m7i
    - db.m7g
    - db.m6g
    - db.m6gd
    - db.m6i

  - db.m5
  - db.m5d
  - db.r7i
  - db.r7g
  - db.r6g
  - db.r6gd
  - db.r6i
  - db.r5
  - db.r5b
  - db.r5d
  - db.x2idn
  - db.x2iedn

  有关数据库实例类的信息，请参阅 the section called “数据库实例类”。

  数据库实例类可用性因 Amazon Web Services 区域而异。要确定特定 Amazon Web Services 区域中是否支持数据库实例类，请参阅the section called “确定 Amazon Web Services 区域 中的数据库实例类支持”。
- 在与数据库关联的参数组中，`rds.optimized_writes` 参数设置为 `AUTO`。在默认参数组中，此参数始终设置为 `AUTO`。

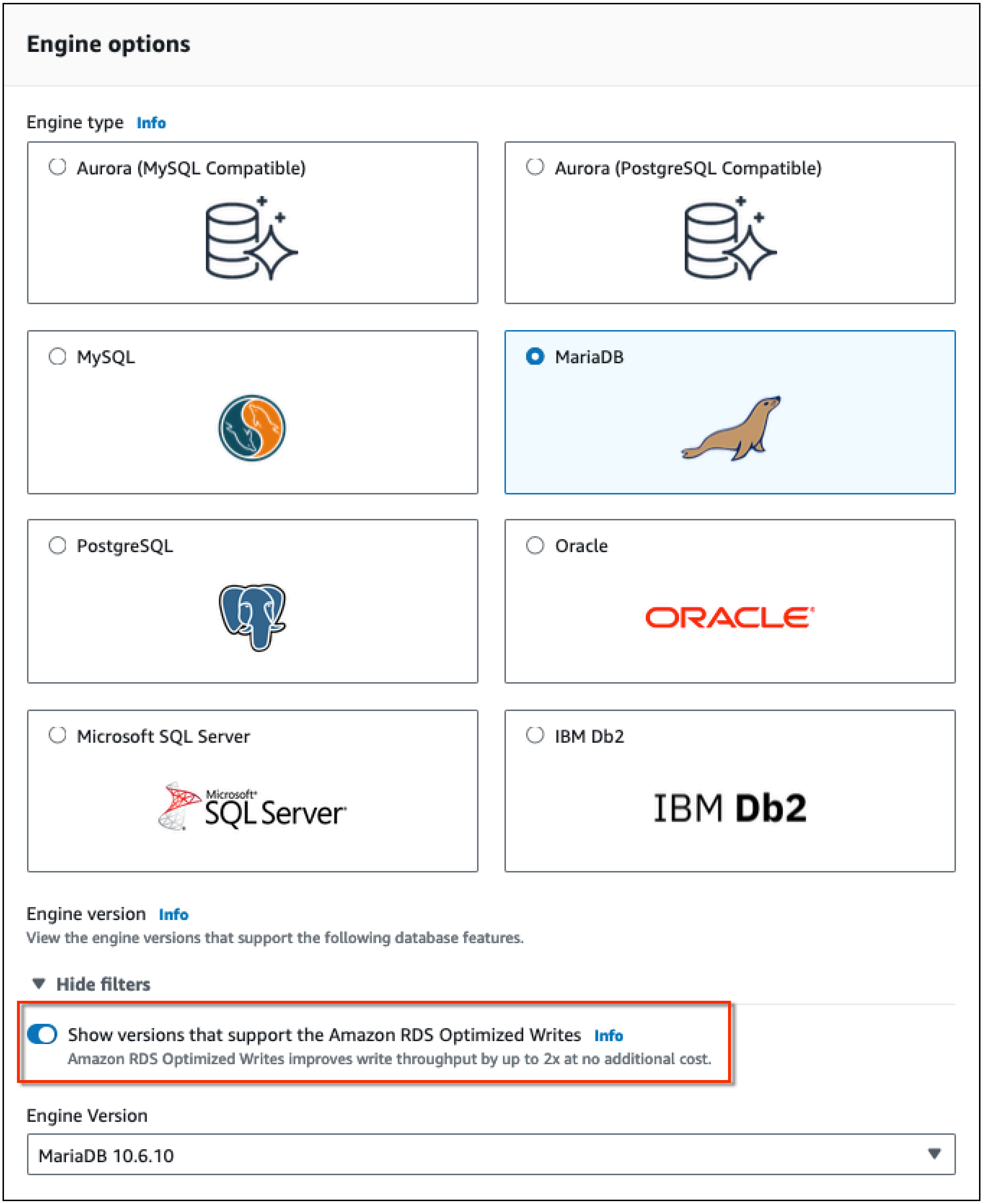
如果您需要使用支持 RDS 优化型写入特征的数据库引擎版本和数据库实例类，但您不想使用此特征，则在创建数据库时指定一个自定义参数组。在此参数组中，将 `rds.optimized_writes` 参数设置为 `OFF`。如果您希望数据库稍后使用 RDS 优化写入，可以将此参数设置为 `AUTO` 以将其开启。有关创建自定义参数组和设置参数的信息，请参阅 Amazon RDS 的参数组。

有关创建数据库实例的信息，请参阅创建 Amazon RDS 数据库实例。

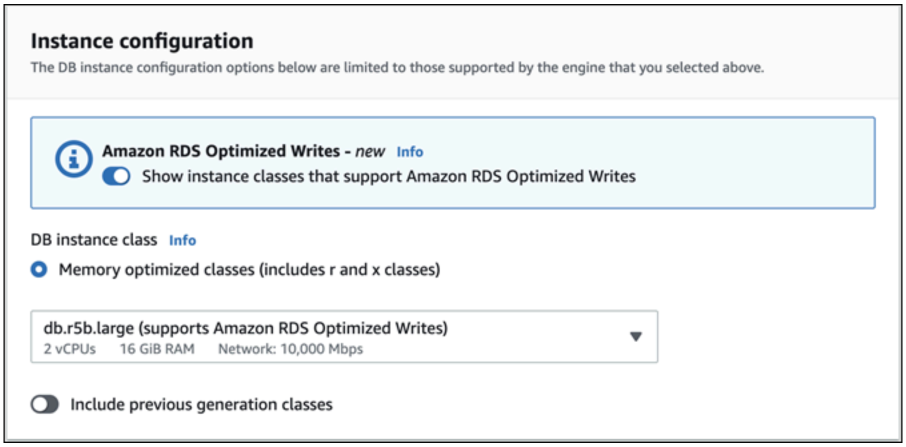
控制台

当您使用 RDS 控制台创建 RDS for MariaDB 数据库时，您可以筛选支持 RDS 优化型写入功能的数据库引擎版本和数据库实例类。开启筛选条件后，您可以从可用的数据库引擎版本和数据库实例类中进行选择。

要选择支持 RDS 优化型写入功能的数据库引擎版本，请在引擎版本中筛选支持该功能的 RDS for MariaDB 数据库引擎版本，然后选择版本。


Engine options
Engine type Info
Aurora (MySQL Compatible)
Aurora (PostgreSQL Compatible)
MySQL
MariaDB
PostgreSQL
Oracle
ORACLE
Microsoft SQL Server
Microsoft SQL Server
IBM Db2
IBM Db2
Engine version Info
View the engine versions that support the following database features.
Hide filters
Show versions that support the Amazon RDS Optimized Writes Info
Amazon RDS Optimized Writes improves write throughput by up to 2x at no additional cost.
Engine Version
MariaDB 10.6.10

在 Instance configuration（实例配置）部分中，筛选支持 RDS 优化写入的数据库实例类，然后选择数据库实例类。



做出这些选择后，您可以选择符合要求的其他设置，并使用控制台完成创建 RDS for MariaDB 数据库。

Amazon CLI

要使用 Amazon CLI 创建数据库实例，请运行 create-db-instance 命令。确保 `--engine-version` 和 `--db-instance-class` 值支持 RDS 优化写入。此外，确保与数据库实例关联的参数组将 `rds.optimized_writes` 参数设置为 `AUTO`。以下示例将默认参数组与数据库实例关联。

Example 创建使用 RDS 优化写入的数据库实例

对于 Linux、macOS 或 Unix：

```
aws rds create-db-instance \
    --db-instance-identifier mydbinstance \
    --engine mariadb \
    --engine-version 10.6.10 \
    --db-instance-class db.r5b.large \
    --manage-master-user-password \
    --master-username admin \
    --allocated-storage 200
```

对于 Windows：

```
aws rds create-db-instance ^
    --db-instance-identifier mydbinstance ^
    --engine mariadb ^
    --engine-version 10.6.10 ^
```

```
    --db-instance-class db.r5b.large ^
    --manage-master-user-password ^
    --master-username admin ^
    --allocated-storage 200
```

RDS API

您可以使用 CreateDBInstance 操作创建数据库实例。使用此操作时，请确保 EngineVersion 和 DBInstanceClass 值支持 RDS 优化写入。此外，确保与数据库实例关联的参数组将 rds.optimized_writes 参数设置为 AUTO。

在现有数据库上启用 RDS 优化写入

要修改现有 RDS for MariaDB 数据库以开启 RDS 优化写入功能，数据库必须使用支持的数据库引擎版本和数据库实例类创建。此外，数据库必须是在 2023 年 3 月 7 日发布 RDS 优化写入之后创建的，因为所需的底层文件系统配置与此次发布前创建的数据库的配置不兼容。如果满足上述条件，则可以通过将 rds.optimized_writes 参数设置为 AUTO 来开启 RDS 优化写入。

如果您的数据库不是使用支持的引擎版本、实例类或文件系统配置创建的，则可以使用 RDS 蓝绿部署来迁移到支持的配置。创建蓝绿部署时，请执行以下操作：

- 选择在绿色数据库上启用优化写入，然后指定支持 RDS 优化写入的引擎版本和数据库实例类。有关支持的引擎版本和实例类的列表，请参阅the section called “与新数据库一起使用”。
- 在存储下，选择升级存储文件系统配置。此选项将数据库升级到兼容的底层文件系统配置。

创建蓝绿部署时，如果将 rds.optimized_writes 参数设置为 AUTO，则在绿色环境中将自动启用 RDS 优化写入。然后，您可以切换蓝绿部署，以将绿色环境升级为新的生产环境。

有关更多信息，请参阅 the section called “创建蓝绿部署”。

RDS 优化写入的限制

当您从快照中还原 RDS for MariaDB 数据库时，仅当满足以下所有条件时，才能为数据库开启 RDS 优化写入功能：

- 快照是从支持 RDS 优化写入的数据库创建的。
- 快照是从发布 RDS 优化型写入功能之后创建的数据库中创建的。
- 快照还原为支持 RDS 优化写入的数据库。
- 还原的数据库与一个参数组相关联，该参数组的 rds.optimized_writes 参数设置为 AUTO。

升级 MariaDB 数据库引擎

如果 Amazon RDS 支持数据库引擎的新版本，您可以将您的数据库实例升级到新版本。MariaDB 数据库实例具有两种升级：主要版本升级和次要版本升级。

主要版本升级会包含不与现有应用程序向后兼容的数据库更改。因此，您必须手动为数据库实例执行主要版本升级。您可以通过修改数据库实例来启动主要版本升级。但是，执行主要版本升级之前，建议您按照RDS for MariaDB 的主要版本升级中的说明操作。

相比之下，次要版本升级仅包括与现有应用程序向后兼容的更改。您可以通过修改数据库实例来手动启动次要版本升级。您也可以在创建或修改数据库实例时启用自动次要版本升级选项。这样做意味着数据库实例在 Amazon RDS 测试并批准新版本后会自动升级。有关执行升级的信息，请参阅 升级数据库实例引擎版本。

如果 MariaDB 数据库实例正在使用只读副本，则必须在升级源实例之前升级所有只读副本。如果数据库实例在多可用区部署中，则同时升级写入器和备用副本。在升级完成之前，数据库实例可能无法使用。

有关 MariaDB 支持的版本以及版本管理的更多信息，请参阅 Amazon RDS 上的 MariaDB 版本。

数据库引擎升级需要停机。停机持续时间因数据库实例的大小而异。

> **Tip**
>
> 您可以使用蓝绿部署，显著减少数据库实例升级所需的停机时间。有关更多信息，请参阅 使用 Amazon RDS 蓝绿部署进行数据库更新。

主题

- MariaDB 升级的注意事项
- 查找有效的升级目标
- MariaDB 版本号
- RDS for MariaDB 中的 RDS 版本号
- RDS for MariaDB 的主要版本升级
- 升级 MariaDB 数据库实例
- RDS for MariaDB 的自动次要版本升级

- 使用只读副本在升级 RDS for MariaDB 数据库时减少停机时间

MariaDB 升级的注意事项

在升级流程期间，Amazon RDS 会创建两个或更多数据库快照。在进行任何升级更改之前，Amazon RDS 最多会拍摄两个数据库实例快照。如果无法完成数据库的升级，那么，就可以还原这些快照之一，来创建一个运行旧版本的数据库实例。升级完成后，Amazon RDS 会为数据库实例拍摄另一个快照。无论 Amazon Backup 是否管理数据库实例的备份，Amazon RDS 都会拍摄这些快照。

> **Note**
>
> 如果您将数据库实例的备份保留期设置为大于 0 的数字，则 Amazon RDS 只需创建数据库快照。要更改您的备份保留期，请参阅 修改 Amazon RDS 数据库实例。

升级完成后，您无法恢复为数据库引擎的以前版本。如果要返回以前版本，则还原创建的第一个数据库快照以新建数据库实例。

您可以控制何时将数据库实例升级到 Amazon RDS 支持的新版本。此级别的控制可帮助您保留与特定数据库版本的兼容性，并在生产中部署前使用应用程序测试新版本。在您准备好后，可在与时间表最搭配的时间执行版本升级。

如果数据库实例使用只读副本，则必须先升级所有只读副本，然后再升级源实例。

如果数据库实例在多可用区部署中，则同时升级主数据库实例和备用数据库实例。主副本和备用数据库实例将同时升级，而您会遇到中断，直到升级完成。中断时间因数据库引擎、引擎版本和数据库实例大小而异。

查找有效的升级目标

使用 Amazon Web Services Management Console 升级数据库实例时，它会显示数据库实例的有效升级目标。您还可以运行以下 Amazon CLI 命令来标识数据库实例的有效升级目标：

对于 Linux、macOS 或 Unix：

```
aws rds describe-db-engine-versions \
  --engine mariadb \
  --engine-version version_number \
  --query "DBEngineVersions[*].ValidUpgradeTarget[*].{EngineVersion:EngineVersion}" --
output text
```

对于 Windows：

```
aws rds describe-db-engine-versions ^
  --engine mariadb ^
  --engine-version version_number ^
  --query "DBEngineVersions[*].ValidUpgradeTarget[*].{EngineVersion:EngineVersion}" --
output text
```

例如，要标识 MariaDB 版本 10.5.17 数据库实例的有效升级目标，请运行以下 Amazon CLI 命令：

对于 Linux、macOS 或 Unix：

```
aws rds describe-db-engine-versions \
  --engine mariadb \
  --engine-version 10.5.17 \
  --query "DBEngineVersions[*].ValidUpgradeTarget[*].{EngineVersion:EngineVersion}" --
output text
```

对于 Windows：

```
aws rds describe-db-engine-versions ^
  --engine mariadb ^
  --engine-version 10.5.17 ^
  --query "DBEngineVersions[*].ValidUpgradeTarget[*].{EngineVersion:EngineVersion}" --
output text
```

MariaDB 版本号

RDS for MariaDB 数据库引擎的版本编号序列采用 major.minor.patch.YYYYMMDD 或 major.minor.patch 形式，例如 10.11.5.R2.20231201 或 10.4.30。所使用的格式取决于 MariaDB 引擎版本。

major

  主要版本号是版本号的整数和第一个小数点后的部分，例如 10.11。主要版本升级会增加版本号的主要版本部分。例如，从 10.5.20 升级到 10.6.12 是主要版本升级，其中 10.5 和 10.6 为主要版本号。

minor

  次要版本号是版本号的第三部分，例如 10.11.5 中的 5。

patch

补丁是版本号的第四部分，例如 10.11.5.R2 中的 R2。RDS 补丁版本包括在发布后添加到次要版本的重要错误修复。

YYYYMMDD

日期是版本号的第五部分，例如 10.11.5.R2.20231201 中的 20231201。RDS 日期版本是一个安全补丁，它包括在发布后添加到次要版本的重要安全修复程序。它不包括任何可能更改引擎行为的修复程序。

下表说明了 RDS for MariaDB 版本 10.11 的命名方案。

10.11 次要版本	命名模式
≥5	新的数据库实例使用 major.minor.patch.YYMMDD，例如 10.11.5.R2.20231201。 在下一次主要版本或次要版本升级之前，现有数据库实例可能使用 major.minor.patch，例如 10.11.5.R2。
< 5	现有数据库实例使用 major.minor.patch，例如 10.11.4.R2。

下表说明了 RDS for MariaDB 版本 10.6 的命名方案。

10.6 次要版本	命名模式
≥ 14	新的数据库实例使用 major.minor.patch.YYMMDD，例如 10.6.14.R2.20231201。 在下一次主要版本或次要版本升级之前，现有数据库实例可能使用 major.minor.patch，例如 10.6.14.R2。
< 14	现有数据库实例使用 major.minor.patch，例如 10.6.13.R2。

下表说明了 RDS for MariaDB 版本 10.5 的命名方案。

10.5 次要版本	命名模式
≥ 21	新的数据库实例使用 major.minor.patch.YYMMDD，例如 10.5.21.R2.20231201。 在下一次主要版本或次要版本升级之前，现有数据库实例可能使用 major.minor.patch，例如 10.5.21.R2。
< 21	现有数据库实例使用 major.minor.patch，例如 10.5.20.R2。

下表说明了 RDS for MariaDB 版本 10.4 的命名方案。

10.4 次要版本	命名模式
≥ 30	新的数据库实例使用 major.minor.patch.YYMMDD，例如 10.4.30.R2.20231201。 在下一次主要版本或次要版本升级之前，现有数据库实例可能使用 major.minor.patch，例如 10.4.30.R2。
< 30	现有数据库实例使用 major.minor.patch，例如 10.4.29.R2。

RDS for MariaDB 中的 RDS 版本号

RDS 版本号使用 `major.minor.patch` 或 `major.minor.patch.YYYYMMDD` 命名方案。RDS 补丁版本包括在发布后添加到次要版本的重要错误修复。RDS 日期版本（`YYMMDD`）是一个安全补丁。安全补丁不包括任何可能更改引擎行为的修复程序。

要识别数据库的 Amazon RDS 版本号，必须先使用以下命令创建 `rds_tools` 扩展：

```
CREATE EXTENSION rds_tools;
```

您可以通过以下 SQL 查询找出 RDS for MariaDB 数据库的 RDS 版本号：

```
mysql> select mysql.rds_version();
```

例如，查询 RDS for MariaDB 10.6.14 数据库将返回以下输出：

```
+----------------------+
| mysql.rds_version() |
+----------------------+
| 10.6.14.R2.20231201  |
+----------------------+
1 row in set (0.01 sec)
```

RDS for MariaDB 的主要版本升级

主要的版本升级可能包含未与现有应用程序向后兼容的数据库更改。因此，Amazon RDS 不会自动应用主要版本升级。您必须手动修改数据库实例。建议您在将任何升级应用于生产实例前全面测试这些升级。

Amazon RDS 支持 MariaDB 数据库引擎的以下主要版本就地升级：

- 任何 MariaDB 版本到 MariaDB 11.4
- 任何 MariaDB 版本到 MariaDB 10.11
- 任何 MariaDB 版本到 MariaDB 10.6
- MariaDB 10.4 到 MariaDB 10.5

如果使用的是自定义参数组并且要执行主要版本升级，则必须为新的数据库引擎版本指定默认参数组，或者为其创建自定义参数组。在完成升级后，将新的参数组与数据库实例关联需要重新启动客户启动的数据库。如果需要重新启动该实例来应用参数组更改，则该实例的参数组状态将显示 `pending-reboot`。可在 Amazon Web Services Management Console 中查看实例的参数组状态，或者通过运行“describe”调用进行查看，例如 `describe-db-instances`。

升级 MariaDB 数据库实例

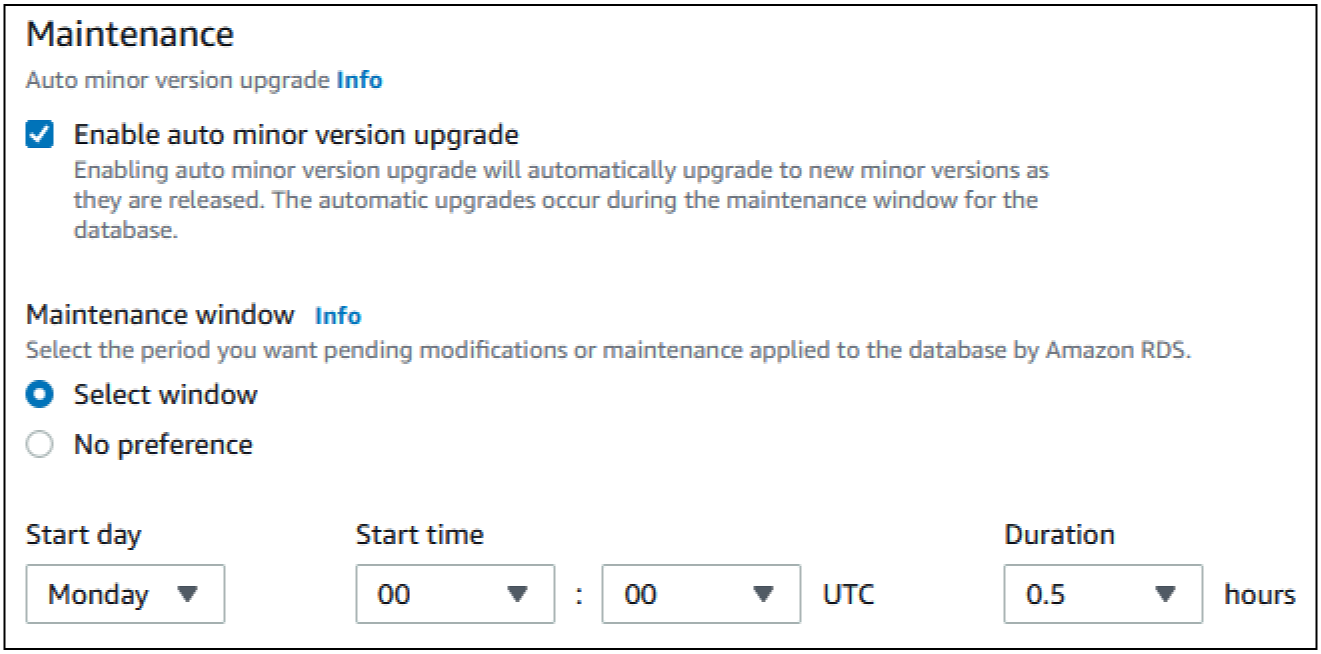
有关手动或自动升级 MariaDB 数据库实例的信息，请参阅升级数据库实例引擎版本。

RDS for MariaDB 的自动次要版本升级

如果在创建或修改数据库实例时指定以下设置，则可以让数据库实例自动升级。

- 自动次要版本升级设置已启用。
- 备份保留期设置需大于 0。

在 Amazon Web Services Management Console 中，这些设置位于 Additional configuration（其他配置）下。下图显示了 Auto minor version upgrade（自动次要版本升级）设置。



有关这些设置的更多信息，请参阅 数据库实例的设置。

对于某些 Amazon Web Services 区域中的某些 RDS for MariaDB 主要版本，RDS 将一个次要版本指定为自动升级版本。在 Amazon RDS 测试并批准次要版本后，将在维护时段内自动进行次要版本升级。RDS 不会自动将发布的较新次要版本设置为自动升级版本。在 RDS 指定较新的自动升级版本之前，将会考虑一些条件，例如：

- 已知的安全问题
- MariaDB 社群版本中的错误
- 发布次要版本后的整体队列稳定性

> **Note**
>
> 从 MariaDB 的特定次要版本开始，已删除对使用 TLS 版本 1.0 和 1.1 的支持。有关支持的 MariaDB 次要版本的信息，请参阅the section called “MariaDB 的 SSL/TLS 支持”。

您可以运行以下 Amazon CLI 命令来确定特定 Amazon Web Services 区域中指定 MariaDB 次要版本的当前自动次要版本升级目标版本。

对于 Linux、macOS 或 Unix：

```
aws rds describe-db-engine-versions \
--engine mariadb \
--engine-version minor_version \
--region region \
--query "DBEngineVersions[*].ValidUpgradeTarget[*].
{AutoUpgrade:AutoUpgrade,EngineVersion:EngineVersion}" \
--output text
```

对于 Windows：

```
aws rds describe-db-engine-versions ^
--engine mariadb ^
--engine-version minor_version ^
--region region ^
--query "DBEngineVersions[*].ValidUpgradeTarget[*].
{AutoUpgrade:AutoUpgrade,EngineVersion:EngineVersion}" ^
--output text
```

例如，以下 Amazon CLI 命令确定美国东部（俄亥俄州）Amazon Web Services 区域（us-east-2）中 MariaDB 次要版本 10.5.16 的自动次要版本升级目标。

对于 Linux、macOS 或 Unix：

```
aws rds describe-db-engine-versions \
--engine mariadb \
--engine-version 10.5.16 \
--region us-east-2 \
--query "DBEngineVersions[*].ValidUpgradeTarget[*].
{AutoUpgrade:AutoUpgrade,EngineVersion:EngineVersion}" \
--output table
```

对于 Windows：

```
aws rds describe-db-engine-versions ^
--engine mariadb ^
--engine-version 10.5.16 ^
--region us-east-2 ^
--query "DBEngineVersions[*].ValidUpgradeTarget[*].
{AutoUpgrade:AutoUpgrade,EngineVersion:EngineVersion}" ^
--output table
```

输出类似于以下内容。

```
----------------------------------
|    DescribeDBEngineVersions    |
+--------------+-----------------+
|  AutoUpgrade |  EngineVersion  |
+--------------+-----------------+
|  True        |  10.5.17        |
|  False       |  10.5.18        |
|  False       |  10.5.19        |
|  False       |  10.6.5         |
|  False       |  10.6.7         |
|  False       |  10.6.8         |
|  False       |  10.6.10        |
|  False       |  10.6.11        |
|  False       |  10.6.12        |
+--------------+-----------------+
```

在此示例中，`AutoUpgrade` 值对于 MariaDB 版本 10.5.17 为 `True`。因此，自动次要版本升级目标是 MariaDB 版本 10.5.17，它在输出中突出显示。

如果满足以下条件，则会在维护时段内自动升级 MariaDB 数据库实例：

- 自动次要版本升级设置已启用。
- 备份保留期设置需大于 0。
- 数据库实例运行的次要数据库引擎版本低于当前的自动升级次要版本。

有关更多信息，请参阅"自动升级次要引擎版本"。

使用只读副本在升级 RDS for MariaDB 数据库时减少停机时间

在大多数情况下，蓝绿部署是升级 MariaDB 数据库实例时减少停机时间的最佳选择。有关更多信息，请参阅 使用 Amazon RDS 蓝绿部署进行数据库更新。

如果您无法使用蓝绿部署，并且 MariaDB 数据库实例当前与生产应用程序结合使用，则可使用以下过程来升级数据库实例的数据库版本。此过程可以减少应用程序的停机时间。

使用只读副本，您可以提前执行大多数维护步骤，并在实际停机期间尽量减少必要的更改。利用此技巧，您可以在不对现有数据库实例进行任何更改的情况下测试和准备新的数据库实例。

以下过程显示了从 MariaDB 版本 10.5 升级到 MariaDB 版本 10.6 的示例。您可以使用相同的常规步骤升级到其他主要版本。

在使用数据库实例时升级 MariaDB 数据库

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 创建 MariaDB 10.5 数据库实例的只读副本。该过程可创建数据库的可升级副本。可能还存在数据库实例的其他只读副本。
   a. 在控制台上，选择数据库，然后选择您要升级的数据库实例。
   b. 对于 Actions (操作)，选择 Create read replica (创建只读副本)。
   c. 为只读副本提供 DB instance identifier（数据库实例标识符）的值，并确保 DB instance class（数据库实例类）和其他设置与您的 MariaDB 10.5 数据库实例匹配。
   d. 选择 Create read replica (创建只读副本)。
3. （可选）创建只读副本且状态显示为可用之后，将只读副本转换为多可用区部署并启用备份。

   默认情况下，只读副本将创建为已禁用备份的单可用区部署。由于只读副本最终成为生产数据库实例，因此最佳实践是立即配置多可用区部署并启用备份。

   a. 在控制台上，选择数据库，然后选择您刚创建的只读副本。
   b. 选择修改。
   c. 对于多可用区部署，请选择创建备用实例。
   d. 对于 Backup Retention Period（备份保留期），请选择正的非零值（如 3 天），然后选择 Continue（继续）。
   e. 对于修改计划，选择立即应用。
   f. 选择修改数据库实例。
4. 当只读副本 Status（状态）显示 Available（可用）后，将只读副本升级到 MariaDB 10.6：
   a. 在控制台上，选择数据库，然后选择您刚创建的只读副本。
   b. 选择修改。
   c. 对于 DB engine version（数据库引擎版本），选择要升级到的 MariaDB 10.6 版本，然后选择 Continue（继续）。
   d. 对于修改计划，选择立即应用。
   e. 选择 Modify DB instance (修改数据库实例)开始升级。

5. 升级完成并且 Status（状态）显示 Available（可用）之后，验证升级的只读副本是否与源 MariaDB 10.5 数据库实例保持同步。要进行验证，请连接到只读副本并运行 `SHOW REPLICA STATUS` 命令。如果 `Seconds_Behind_Master` 字段为 `0`，则复制保持最新。

   > **Note**
   >
   > 以前的 MariaDB 版本使用的是 `SHOW SLAVE STATUS`，而不是 `SHOW REPLICA STATUS`。如果您使用的 MariaDB 版本低于 10.6，请使用 `SHOW SLAVE STATUS`。

6. （可选）创建只读副本的只读副本。

   如果您希望数据库实例在提升为独立数据库实例后拥有只读副本，则可以立即创建只读副本。

   a. 在控制台上，选择数据库，然后选择您刚刚升级的只读副本。
   b. 对于 Actions (操作)，选择 Create read replica (创建只读副本)。
   c. 为只读副本提供 DB instance identifier（数据库实例标识符）的值，并确保 DB instance class（数据库实例类）和其他设置与您的 MariaDB 10.5 数据库实例匹配。
   d. 选择 Create read replica (创建只读副本)。

7. （可选）为只读副本配置自定义数据库参数组。

   如果您希望数据库实例在提升为独立数据库实例后使用自定义参数组，则可以立即创建数据库参数组，并将其与只读副本相关联。

   a. 为 MariaDB 10.6 创建自定义数据库参数组。有关说明，请参阅 在 Amazon RDS 中创建数据库参数组。
   b. 修改要在刚创建的数据库参数组中更改的参数。有关说明，请参阅在 Amazon RDS 中修改数据库参数组中的参数。
   c. 在控制台中，选择数据库，然后选择只读副本。
   d. 选择修改。
   e. 对于 DB parameter group（数据库参数组），请选择刚创建的 MariaDB 10.6 数据库参数组，然后选择 Continue（继续）。
   f. 对于修改计划，选择立即应用。
   g. 选择 Modify DB instance (修改数据库实例)开始升级。

8. 使您的 MariaDB 10.6 只读副本成为独立的数据库实例。

> **Important**
>
> 将 MariaDB 10.6 只读副本提升为独立的数据库实例后，它不再是 MariaDB 10.5 数据库实例的副本。建议您在源 MariaDB 10.5 数据库实例处于只读模式并且所有写入操作都暂停的维护时段期间，提升 MariaDB 10.6 只读副本。提升完成之后，您可以将写入操作定向到已升级的 MariaDB 10.6 数据库实例，以确保不会丢失写入操作。
>
> 此外，建议您在提升 MariaDB 10.6 只读副本之前，对 MariaDB 10.6 只读副本执行所有必需的数据定义语言（DDL）操作。例如，创建索引。此方法可避免在提升 MariaDB 10.6 只读副本之后对其性能造成任何负面影响。要提升只读副本，请使用以下过程。

   a. 在控制台上，选择数据库，然后选择您刚刚升级的只读副本。
   b. 对于操作，请选择提升。
   c. 选择是，为只读副本实例启用自动备份。有关更多信息，请参阅“备份简介”。
   d. 选择 Continue (继续)。
   e. 选择 Promote Read Replica。

9. 您现在有了 MariaDB 数据库的升级版本。此时，您可以将应用程序定向到新的 MariaDB 10.6 数据库实例。

将数据导入到 Amazon RDS for MariaDB 数据库实例

您可以使用多种不同的技术将数据导入到 RDS for MariaDB 数据库实例。最佳方法取决于多种因素：

- 数据的来源
- 数据量
- 一次性导入，还是持续导出
- 停机时间量

如果还要随数据一起迁移应用程序，则务必考虑停机时间量。

下表中列出了用于将数据导入到 RDS for MariaDB 数据库实例的技巧：

> **Note**
>
> Amazon RDS 仅支持从 Amazon S3 导入到 RDS for MySQL 数据库实例。目前不支持将使用 `mariadb-backup` 创建的备份导入到 RDS for MariaDB 中。

来源	数据量	一次性或持续	应用程序停机时间	方法	更多信息
本地或 Amazon EC2 上的现有 MariaDB 数据库	任何	持续	最低	配置复制，使用现有 MariaDB 数据库作为复制源。 当外部实例为 MariaDB 版本 10.0.24 或更高版本时，您可以使用 MariaDB 全局事务标识符 (GTID) 配置复制到 MariaDB 数据库实例，对于 10.0.24 之前的版本，则可以使用 MariaDB 实例的二进制日志坐标来完成该操作。MariaDB GTID 与 MySQL GTID 的实施不同，后者不受 Amazon RDS 支持。	配置与外部源实例之间的二进制日志文件位置复制 将数据导入到 Amazon RDS for MariaDB

来源	数据量	一次性或持续	应用程序停机时间	方法	更多信息
					数据库实例并减少停机时间
任何现有数据库	任何	一次性或持续	最低	使用 Amazon Database Migration Service 可在最少停机时间下迁移数据库，对于很多数据库引擎，将继续进行持续复制。	Amazon Database Migration Service 用户指南中的什么是 Amazon Database Migration Service 和将与 MySQL 兼容的数据库作为 Amazon DMS 的目标
现有 MariaDB 数据库实例	任何	一次性或持续	最低	为正在进行的复制创建只读副本。提升只读副本用于一次性创建新数据库实例。	使用数据库实例只读副本

来源	数据量	一次性或持续	应用程序停机时间	方法	更多信息
现有 MariaDB 数据库	小型	一次性	一段时间	使用命令行实用程序，将数据直接复制到您的 MariaDB 数据库实例。	将数据从外部 MariaDB 数据库导入到 Amazon RDS for MariaDB 数据库实例
未存储在现有数据库中的数据	中	一次性	一段时间	使用 MariaDB `LOAD DATA LOCAL INFILE` 语句创建平面文件并导入这些文件。	从任意源将数据导入 Amazon RDS for MariaDB 数据库实例

Note

`mysql` 系统数据库包含登录到数据库实例并访问数据所需的身份验证和授权信息。在数据库实例中删掉、改动、重命名或截取 `mysql` 数据库中的表、数据或其他内容可能会导致错误，并可能导致无法访问数据库实例和数据。如果发生此情况，您可以使用 Amazon CLI restore-db-instance-from-db-snapshot 命令从快照中还原数据库实例。您可以使用 restore-db-instance-to-point-in-time 命令恢复数据库实例。

为 MariaDB 导入数据的注意事项

以下内容包含与将数据加载到 MariaDB 中相关的技术信息。此内容面向熟悉 MariaDB 服务器架构的用户。

二进制日志记录

与禁用日志记录相比，启用二进制日志记录会降低数据加载性能，并且需要多达四倍的额外磁盘空间。用于加载数据的事务大小直接影响系统性能和磁盘空间需求，也即，事务越大，需要的资源越多。

事务大小

事务大小会影响 MariaDB 数据加载的以下方面：

- 资源消耗
- 磁盘空间利用率
- 恢复进程
- 恢复时间
- 输入格式（平面文件或 SQL）

本部分介绍了事务大小如何影响二进制日志记录以及让案例在大数据加载过程中禁用二进制日志记录。可以通过设置 Amazon RDS 自动备份保留期来启用和禁用二进制日志记录。非零值会启用二进制日志记录，零会将其禁用。有关更多信息，请参阅 备份保留期。

本节还介绍大型事务对 InnoDB 的影响以及将事务大小保持为较小的重要性。

小型事务

对于小型事务，二进制日志记录会使加载数据所需的磁盘写入次数翻倍。这种影响会严重降低其他数据库会话的性能，并增加数据加载所需的时间。所经历的降级在一定程度上取决于以下因素：

- 上传速率
- 在加载期间发生的其它数据库活动
- Amazon RDS 数据库实例的容量

此外，二进制日志占用的磁盘空间大致等于加载的数据量，直至备份和移除日志。Amazon RDS 通过频繁备份和移除二进制日志来尽量减少占用的磁盘空间量。

大型事务

对于大型事务，二进制日志记录会使 IOPS 和磁盘使用量增加两倍，原因如下：

- 二进制日志缓存将事务数据临时存储在磁盘上。
- 此缓存会随着事务大小而增长，这会占用磁盘空间。
- 当事务（提交或回滚）完成时，系统会将缓存复制到二进制日志。

此过程会创建数据的三个副本：

- 原始数据
- 磁盘上的缓存
- 最终二进制日志条目

每个写入操作都会产生额外的 IO，从而进一步影响性能。

因此，与禁用日志记录相比，二进制日志记录需要三倍的磁盘空间。例如，在单个事务中加载 10 GiB 的数据会创建三个副本：

- 表数据为 10 GiB
- 二进制日志缓存为 10 GiB
- 二进制日志文件为 10 GiB

所需的临时磁盘空间总量为 30 GiB。

磁盘空间重要注意事项：

- 缓存文件会一直存在，直到会话结束或新事务创建另一个缓存。
- 二进制日志会保留直至对其进行备份，可能会在很长一段时间内占用 20 GiB（缓存和日志）。

如果您使用 `LOAD DATA LOCAL INFILE` 来加载数据，数据恢复会创建第四个副本，以防数据库必须从加载前进行的备份中恢复。在恢复期间，MariaDB 将二进制日志中的数据提取到平面文件中。然后，MariaDB 运行 `LOAD DATA LOCAL INFILE`。在前一个示例的基础上，此恢复需要的临时磁盘空间总量为 40 GiB，也即表、缓存、日志和本地文件各为 10 GiB。如果没有至少 40 GiB 的可用磁盘空间，则恢复将失败。

优化大数据加载

对于大数据加载，请禁用二进制日志记录，以减少开销和磁盘空间要求。可以通过将备份保留期设置为 0 来禁用二进制日志记录。完成加载后，将备份保留期还原为适当的非零值。有关更多信息，请参阅修改 Amazon RDS 数据库实例和设置表中的备份保留期。

> **Note**
>
> 如果数据库实例是只读副本的源数据库实例，则您无法将备份保留期设置为 0。

在加载数据之前，我们建议您创建数据库快照。有关更多信息，请参阅 管理手动备份。

InnoDB

以下有关撤消日志记录和恢复选项的信息支持将 InnoDB 事务保持为较小，以优化数据库性能。

了解 InnoDB 撤消日志记录

撤消是一种日志记录机制，可启用事务回滚并支持多版本并发控制（MVCC）。

对于 MariaDB 10.11 及更低版本，撤消日志存储在 InnoDB 系统表空间（通常为 ibdata1）中，并保留至清除线程移除它们为止。因此，大数据加载事务可能会导致系统表空间变得相当大，并占用磁盘空间，除非重新创建数据库，否则无法回收这些磁盘空间。

对于所有 MariaDB 版本，清除线程必须等待以移除所有撤消日志，直到最旧的活动事务提交或回滚。如果数据库在加载过程中正在处理其它事务，则其撤消日志也会累积并且无法移除，即使事务提交并且没有其它事务需要 MVCC 的撤消日志，也是如此。在这种情况下，所有事务（包括只读事务）都会变慢。之所以发生这种变慢过程，是因为所有事务都访问任何事务（而不仅仅是加载事务）更改的所有行。实际上，事务必须扫描一些撤消日志，在撤消日志清理期间，长时间运行的加载事务已阻止清除这些撤消日志。这会影响访问已修改行的任何操作的性能。

InnoDB 事务恢复选项

尽管 InnoDB 优化了提交操作，但大型事务回滚速度很慢。为了加快恢复速度，请执行时间点故障恢复或还原数据库快照。有关更多信息，请参阅时间点故障恢复和还原到数据库实例。

数据导入格式

MariaDB 支持两种数据导入格式：平面文件和 SQL。查看有关每种格式的信息，以确定最适合您需求的选项。

平面文件

对于小型事务，请使用 `LOAD DATA LOCAL INFILE` 加载平面文件。与使用 SQL 相比，这种数据导入格式具有以下优势：

- 减少网络流量
- 降低数据传输成本
- 减少了数据库处理开销
- 处理速度更快

`LOAD DATA LOCAL INFILE` 会将整个平面文件作为一个事务进行加载。将各个文件的大小保持为较小可获得以下优势：

- 恢复功能：可以保持跟踪哪些文件已加载。如果在加载期间出现问题，可以从停止处继续加载。可能需要将一些数据重新传输到 Amazon RDS，但使用小文件，就可以将重新传输的量减少至最低水平。
- 并行数据加载：如果 IOPS 和网络带宽足以支持单个文件加载，那么并行加载就可以节省时间。
- 加载速率控制：如果数据加载对其它进程产生负面影响，则可以通过增加文件之间的间隔来控制加载速率。

大型事务会降低使用 `LOAD DATA LOCAL INFILE` 导入数据的优势。当您无法将大量数据分解成较小的文件时，可以考虑使用 SQL。

SQL

与平面文件相比，SQL 的一个主要优势为：您可以轻松地将事务大小保持为较小。但是，SQL 的加载时间可能比平面文件长得多。此外，在发生故障后，可能难以确定从哪里恢复，而您无法重新启动 mariadb-dump 文件。如果加载 mariadb-dump 文件时发生故障，则必须在加载可以恢复前修改或替换该文件。或者，在纠正故障的原因后，您可以恢复到加载前的时间点，并重新发送该文件。有关更多信息，请参阅 时间点故障恢复。

使用 Amazon RDS 数据库快照提供数据库检查点

如果您在没有二进制日志记录的情况下长时间（如数小时或数天）加载数据，请使用数据库快照提供定期检查点来实现数据安全。每个数据库快照都会创建数据库实例的一致副本，该副本在系统故障或数据损坏事件期间用作恢复点。由于数据库快照速度很快，因此频繁的检查点对加载性能的影响微乎其微。您可以在不影响数据库持久性或恢复能力的情况下，删除以前的数据库快照。有关数据库快照的更多信息，请参阅管理手动备份。

减少数据库加载时间

以下各项是减少加载时间的其它技巧：

- 在将数据加载到 MariaDB 数据库之前，请创建所有二级索引。与其它数据库系统不同，MariaDB 在添加或修改二级索引时会重建整个表。此过程会创建一个包含索引更改的新表，复制所有数据，然后删除原始表。
- 按主键顺序加载数据。对于 InnoDB 表，这可以将加载时间缩短 75% 至 80%，并将数据文件大小减少 50%。
- 通过将 `foreign_key_checks` 设置为 `0` 来禁用外键约束。对于使用 `LOAD DATA LOCAL INFILE` 加载的平面文件，这通常是必需的。对于任何加载，禁用外键检查会加快数据加载速度。加载完成后，通过将 `foreign_key_checks` 设置为 `1` 来重新启用约束并验证数据。
- 并行加载数据，除非接近资源限制。要实现跨多个表段的并行加载，请在适当时使用分区表。
- 要减少 SQL 执行开销，请将多条 `INSERT` 语句合并为单个多值 `INSERT` 操作。`mariadb-dump` 自动实现此优化。
- 通过将 `innodb_flush_log_at_trx_commit` 设置为 `0` 来减少 InnoDB 日志 IO 操作。加载完成后，将 `innodb_flush_log_at_trx_commit` 还原为 `1`。

> **Warning**
>
> 将 `innodb_flush_log_at_trx_commit` 设置为 `0` 会导致 InnoDB 每秒都刷新其日志，而不是在每次提交时刷新。此设置可以提高性能，但在系统故障期间可能会有事务丢失的风险。

- 如果您要将数据加载到没有只读副本的数据库实例，请将 `sync_binlog` 设置为 `0`。加载完成后，将 `sync_binlog parameter` 还原为 `1`。
- 在将数据库实例转换为多可用区部署之前，将数据加载到单可用区实例中。如果数据库实例已经使用多可用区部署，我们建议您不要切换为单可用区部署来加载数据。这样做所带来的改进微不足道

将数据从外部 MariaDB 数据库导入到 Amazon RDS for MariaDB 数据库实例

您可以将数据从现有 MariaDB 或 MySQL 数据库导入到 RDS for MariaDB 或 RDS for MySQL 数据库实例。为此，请使用 [mysqldump](https://dev.mysql.com/doc/refman/8.0/en/mysqldump.html) 复制数据库，然后通过管道将其直接传输到 RDS for MariaDB 或 RDS for MySQL 数据库实例。`mysqldump` 命令行实用程序通常用于创建备份以及将数据从一个 MariaDB 或 MySQL 服务器传输到另一个 MariaDB 或 MySQL 服务器。该实用程序包含在 MySQL 和 MariaDB 客户端软件中。

在 MariaDB 10.5 中，客户端被称为 mariadb-dump。从 MariaDB 11.0.1 开始，您必须使用 `mariadb-dump` 而不是 `mysqldump`。

> **Note**
>
> 如果您使用 MySQL 数据库实例导入或导出大量数据，则可以使用 `xtrabackup` 备份文件和 Amazon S3 更可靠、更块地将数据移入和移出 Amazon RDS。有关更多信息，请参阅 将备份还原到 Amazon RDS for MySQL 数据库实例。
>
> Amazon RDS 不支持 MariaDB 的 `xtrabackup` 或 `mariabackup`。Amazon RDS 仅支持从 Amazon S3 为 MySQL 导入。

将数据从外部数据库移动到 Amazon RDS 数据库实例的典型 `mysqldump` 命令类似以下示例。将值替换为您自己的信息。对于 MariaDB 11.0.1 及更高版本，请将 `mysqldump` 替换为 `mariadb-dump`，并将 `mysql` 替换为 `mariadb`。

```
mysqldump -u local_user \
    --databases database_name \
    --single-transaction \
    --compress \
    --order-by-primary  \
    --routines=0 \
    --triggers=0 \
    --events=0 \
    -plocal_password | mysql -u RDS_user \
        --port=port_number \
        --host=host_name \
        -pRDS_password
```

> **Important**
>
> 确保 -p 选项和输入的密码之间不留空格。
>
> 作为安全最佳实践，请指定此示例中显示的提示以外的凭证。

确保您了解以下建议和注意事项：

- 从转储文件中排除以下架构：
  - `sys`

  - performance_schema
  - information_schema

  默认情况下，mysqldump 和 mariadb-dump 实用程序排除这些架构。

- 如果您需要迁移用户和权限，请考虑使用生成数据控制语言 (DCL) 的工具 (如 pt-show-grants 实用程序) 来重新创建用户和权限。
- 要执行导入，请确保执行此操作的用户有权访问数据库实例。有关更多信息，请参阅 使用安全组控制访问权限。

所用参数如下所示：

- -u *local_user* – 用于指定用户名。在第一次使用该参数时，请指定您使用 --databases 参数确定的本地 MariaDB 或 MySQL 数据库中的用户账户名称。
- --databases *database_name* – 用于指定本地 MariaDB 或 MySQL 实例上您要导入 Amazon RDS 的数据库的名称。
- --single-transaction – 用于确保从本地数据库加载的所有数据都与单一时间点保持一致。如果在 mysqldump 读取数据期间有其他进程更改数据，使用该参数有助于保持数据完整性。
- --compress – 用于降低网络带宽消耗，方式为将数据从本地数据库发送到 Amazon RDS 之前压缩数据。
- --order-by-primary – 用于减少加载时间，方式为根据主键对每个表中的数据进行排序。
- --routines：如果您要复制的数据库中存在诸如存储过程或函数之类的例程，则使用此参数。将此参数设置为 0，这将在导入过程中排除例程。稍后在 Amazon RDS 数据库中手动重新创建例程。
- --triggers：如果要复制的数据库中存在触发器，则使用此参数。将此参数设置为 0，这将在导入过程中排除触发器。稍后在 Amazon RDS 数据库中手动重新创建触发器。
- --events：如果要复制的数据库中存在事件，则使用此参数。将此参数设置为 0，这将在导入过程中排除事件。稍后在 Amazon RDS 数据库中手动重新创建事件。
- -p*local_password* – 用于指定密码。在第一次使用该参数时，请为您使用第一个 -u 参数标识的用户账户指定密码。
- -u *RDS_user* – 用于指定用户名。在第二次使用该参数时，请为您使用 --host 参数标识的 MariaDB 或 MySQL 数据库实例指定默认数据库上的用户账户的名称。
- --port *port_number* – 用于为您的 MariaDB 或 MySQL 数据库实例指定端口。默认情况下该值为 3306，除非您在创建数据库实例时更改了该值。

- `--host` *`host_name`* – 用于从 Amazon RDS 数据库实例端点指定域名系统 (DNS) 名称，例如，`myinstance.123456789012.us-east-1.rds.amazonaws.com`。您可以在 Amazon RDS 控制台上的数据库实例详细信息中找到端点值。
- `-p`*`RDS_password`* – 用于指定密码。在第二次使用该参数时，您为第二个 `-u` 参数确定的用户账户指定密码。

确保在 Amazon RDS 数据库中手动创建任何存储过程、触发器、函数或事件。如果您所复制的数据库中有上述任一对象，则在运行 `mysqldump` 或 `mariadb-dump` 时会排除这些对象。为此，使用 `mysqldump` 或 `mariadb-dump` 命令时请包含以下参数：

- `--routines=0`
- `--triggers=0`
- `--events=0`

示例

以下示例将本地主机上的 `world` 示例数据库复制到 RDS for MySQL 数据库实例中。将值替换为您自己的信息。要将示例数据库复制到 RDS for MariaDB 数据库实例，请将 `mysqldump` 替换为 `mariadb-dump`，并将 `mysql` 替换为 `mariadb`。

对于 Linux、macOS 或 Unix：

```
sudo mysqldump -u local_user \
    --databases world \
    --single-transaction \
    --compress \
    --order-by-primary  \
    --routines=0 \
    --triggers=0 \
    --events=0 \
    -plocal_password | mysql -u rds_user \
        --port=3306 \
        --host=my_instance.123456789012.us-east-1.rds.amazonaws.com \
        -pRDS_password
```

对于 Windows：

在已通过右键单击 Windows 程序菜单上的命令提示符并选择以管理员身份运行打开的命令提示符中，运行以下命令：将值替换为您自己的信息。要将示例数据库复制到 RDS for MariaDB 数据库实例，请将 mysqldump 替换为 mariadb-dump，并将 mysql 替换为 mariadb。

```
mysqldump -u local_user ^
    --databases world ^
    --single-transaction ^
    --compress ^
    --order-by-primary  ^
    --routines=0 ^
    --triggers=0 ^
    --events=0 ^
    -plocal_password | mysql -u RDS_user ^
        --port=3306 ^
        --host=my_instance.123456789012.us-east-1.rds.amazonaws.com ^
        -pRDS_password
```

Note

作为安全最佳实践，请指定示例中显示的提示以外的凭证。

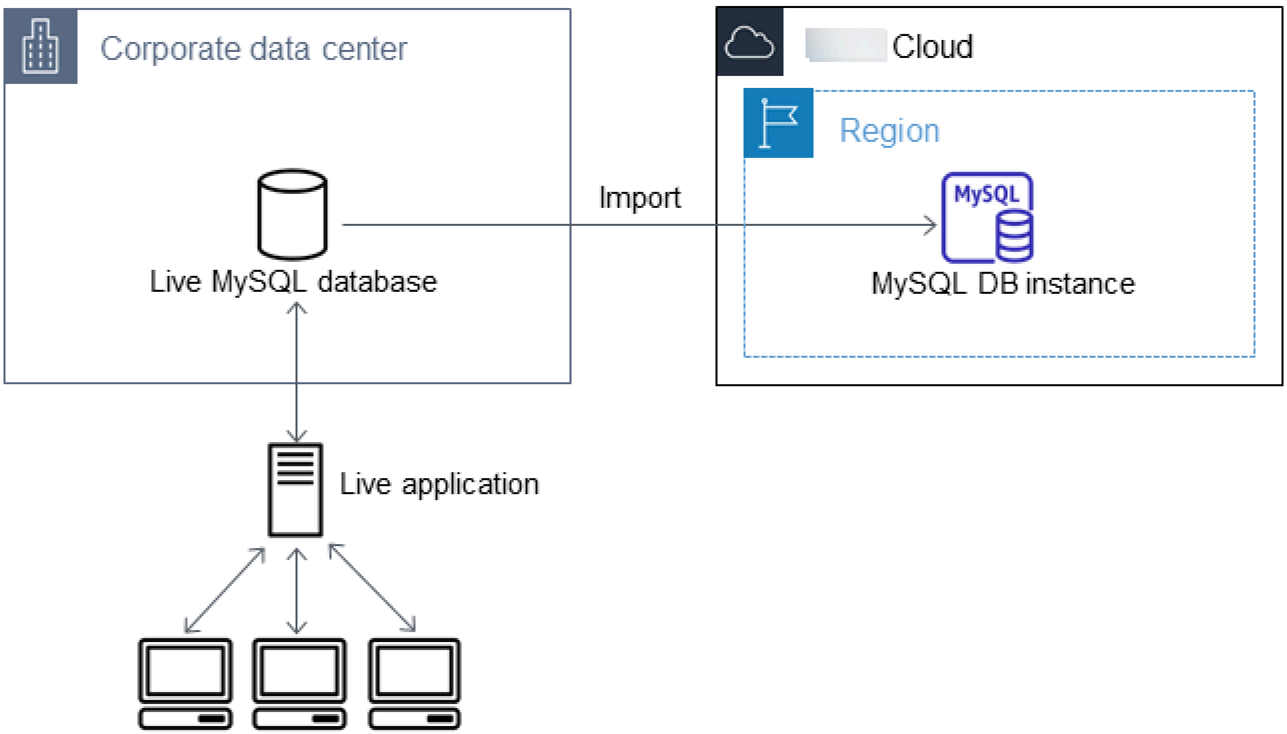
将数据导入到 Amazon RDS for MariaDB 数据库实例并减少停机时间

有时，您可能需要将数据从支持活动应用程序的外部 MariaDB 或 MySQL 数据库导入到 RDS for MariaDB 数据库实例、RDS for MySQL 数据库实例或 RDS for MySQL 多可用区数据库集群。使用以下过程可最大程度地减小对应用程序可用性产生的影响。如果使用非常大的数据库，该步骤也很有用。使用该过程，您可以通过减少跨网络传送到 Amazon 的数据量以降低导入成本。

在该过程中，您将数据库数据的副本传输到 Amazon EC2 实例，并将数据导入到新的 Amazon RDS 数据库中。然后，使用复制功能使 Amazon RDS 数据库与活动的外部实例保持同步，再将应用程序重新导向到 Amazon RDS 数据库。如果外部实例是 MariaDB 10.0.24 或更高版本，并且目标实例是 RDS for MariaDB，则基于全局事务标识符（GTID）配置 MariaDB 复制。否则，将基于二进制日志坐标配置复制。如果您的外部数据库支持基于 GTID 的复制，我们建议使用它，因为基于 GTID 复制是一种更可靠的方法。有关更多信息，请参阅 MariaDB 文档中的全局事务处理 ID。

> **Note**
>
> 如果您想要将数据导入到 RDS for MySQL 数据库实例中且您的方案支持此操作，则建议使用备份文件和 Amazon S3 将数据移入和移出 Amazon RDS。有关更多信息，请参阅 将备份还原到 Amazon RDS for MySQL 数据库实例。

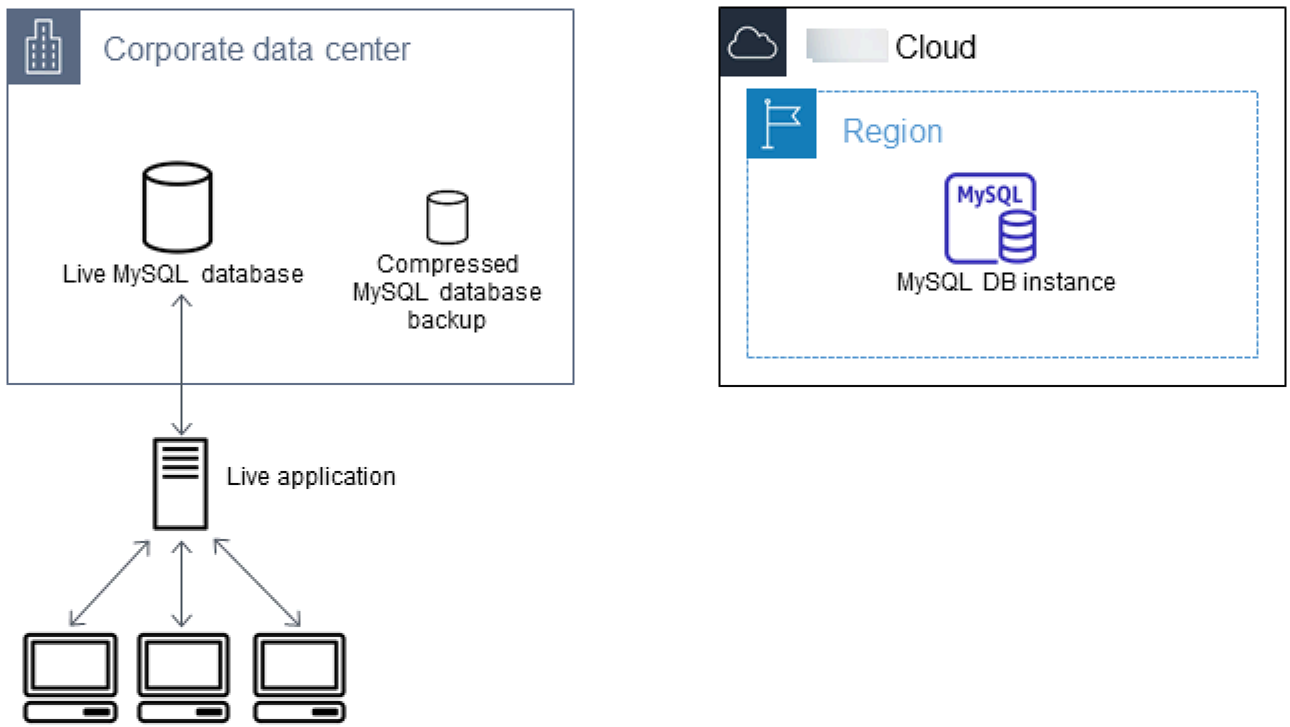
下图显示将外部 MySQL 数据库导入到 Amazon RDS 上的 MySQL 数据库。



任务 1：创建现有数据库的副本

要将大量数据迁移到停机时间最少的 RDS for MariaDB 或 RDS for MySQL 数据库，第一步是创建源数据的副本。

下图显示了创建 MySQL 数据库的备份。



您可以使用 `mysqldump` 实用程序创建 SQL 或分隔的文本格式的数据库备份。在 MariaDB 10.5 中，客户端被称为 [mariadb-dump](https://mariadb.com/kb/en/mariadb-dump/)。从 MariaDB 11.0.1 开始，您必须使用 `mariadb-dump` 而不是 `mysqldump`。建议您在非生产环境中对每种格式运行测试，以便了解哪种方法能最大程度地减少 mysqldump 运行的时间。

我们还建议您对照将分隔文本格式用于加载所获得的好处来权衡 `mysqldump` 性能。使用分隔的文本格式的备份将为所转储的每个表创建制表符分隔的文本文件。为减少导入数据库所需的时间，您可以使用 `LOAD DATA LOCAL INFILE` 命令并行加载这些文件。有关更多信息，请参阅加载数据（适用于 MariaDB）或加载数据（适用于 MySQL）。

在开始备份操作之前，确保对将复制到 Amazon RDS 的 MariaDB 或 MySQL 数据库设置复制选项。复制选项包括开启二进制日志记录和设置唯一的服务器 ID。设置这些选项将促使您的服务器开始记录数据库事务，并做好在该过程的后面充当源复制实例的准备。

确保您了解以下建议和注意事项：

- 将 `--single-transaction` 选项与 `mysqldump` 结合使用，因为它会转储数据库的一致状态。为确保转储文件有效，请勿在 `mysqldump` 运行时运行数据定义语言 (DDL) 语句。您可以为这些操作计划维护时段。
- 从转储文件中排除以下架构：
  - `sys`

- `performance_schema`
- `information_schema`

默认情况下，`mysqldump` 和 `mariadb-dump` 实用程序排除这些架构。

- 如果您需要迁移用户和权限，请考虑使用生成数据控制语言 (DCL) 的工具 (如 pt-show-grants 实用程序) 来重新创建用户和权限。

设置复制选项

1. 编辑 `my.cnf` 文件。此文件通常位于 `/etc` 下。

```
sudo vi /etc/my.cnf
```

将 `log_bin` 和 `server_id` 选项添加到 `[mysqld]` 节。`log_bin` 选项为二进制日志文件提供文件名标识符。`server_id` 选项为源-副本关系中的服务器提供唯一标识符。

以下示例显示 `my.cnf` 文件已更新的 `[mysqld]` 部分：

```
[mysqld]
log-bin=mysql-bin
server-id=1
```

有关更多信息，请参阅 MySQL 文档中的 Setting the Replication Source Configuration。

2. 要使用多可用区数据库集群进行复制，请将 `ENFORCE_GTID_CONSISTENCY` 和 `GTID_MODE` 参数设置为 `ON`。

```
mysql> SET @@GLOBAL.ENFORCE_GTID_CONSISTENCY = ON;
```

```
mysql> SET @@GLOBAL.GTID_MODE = ON;
```

使用数据库实例进行复制不需要这些设置。

3. 重新启动 `mysql` 服务。

```
sudo service mysqld restart
```

创建现有数据库的备份副本

1. 使用 mysqldump 实用程序或 mariadb-dump 实用程序创建数据的备份，并指定 SQL 或分隔文本格式。

   对于 MySQL 8.0.25 及更低版本，指定 --master-data=2 以创建可用于启动服务器间复制的备份文件。对于 MySQL 8.0.26 及更高版本，指定 --source-data=2 以创建可用于启动服务器间复制的备份文件。有关更多信息，请参阅 MySQL 文档中的 mysqldump — A Database Backup Program。

   要提高性能和确保数据完整性，请对于 mysqldump 使用 --order-by-primary 和 --single-transaction 选项。

   要避免在备份中包括 MySQL 系统数据库，请不要将 --all-databases 选项与 mysqldump 结合使用。有关更多信息，请参阅 MySQL 文档中的 Creating a Data Snapshot Using mysqldump。

   如有必要，可使用 chmod 来确保要在其中创建备份文件的目录是可写的。

   > **Important**
   >
   > 在 Windows 中，以管理员身份运行命令窗口。

   - 要生成 SQL 输出，请使用以下命令：

     对于 Linux、macOS 或 Unix：

     ```
     sudo mysqldump \
         --databases database_name \
         --master-data=2  \
         --single-transaction \
         --order-by-primary \
         -r backup.sql \
         -u local_user \
         -ppassword
     ```

     > **Note**
     >
     > 作为安全最佳实践，请指定示例中显示的提示以外的凭证。

对于 Windows：

```
mysqldump ^
    --databases database_name ^
    --master-data=2  ^
    --single-transaction ^
    --order-by-primary ^
    -r backup.sql ^
    -u local_user ^
    -ppassword
```

> **Note**
> 作为安全最佳实践，请指定示例中显示的提示以外的凭证。

- 要生成分隔文本输出，请使用以下命令：

  对于 Linux、macOS 或 Unix：

```
sudo mysqldump \
    --tab=target_directory \
    --fields-terminated-by ',' \
    --fields-enclosed-by '"' \
    --lines-terminated-by 0x0d0a \
    database_name \
    --master-data=2 \
    --single-transaction \
    --order-by-primary \
    -ppassword
```

  对于 Windows：

```
mysqldump ^
    --tab=target_directory ^
    --fields-terminated-by "," ^
    --fields-enclosed-by """ ^
    --lines-terminated-by 0x0d0a ^
    database_name ^
    --master-data=2 ^
    --single-transaction ^
```

```
    --order-by-primary ^
    -ppassword
```

Note

作为安全最佳实践，请指定示例中显示的提示以外的凭证。

确保在 Amazon RDS 数据库中手动创建任何存储过程、触发器、函数或事件。如果您所复制的数据库中有上述任一对象，则在运行 mysqldump 时排除这些对象。为此，请在 mysqldump 命令中包含以下参数：

- --routines=0
- --triggers=0
- --events=0

对于 MySQL 8.0.22 及更低版本以及 MariaDB 版本，当您运行 mysqldump 并指定分隔文本格式时，将返回 CHANGE MASTER TO 注释。此注释包含主日志文件名称和位置。对于 MySQL 8.0.23 及更高版本，如果您使用分隔文本格式运行 mysqldump，将返回 CHANGE REPLICATION SOURCE TO 注释。此注释包含源日志文件名称和位置。如果外部实例不是 MariaDB 10.0.24 和更高版本，也不是 MySQL 8.0.23 和更高版本，请注意 MASTER_LOG_FILE 和 MASTER_LOG_POS 的值。设置复制时需要这些值。

对于 MariaDB 版本和 MySQL 8.0.22 及更低版本，将返回以下输出：

```
-- Position to start replication or point-in-time recovery from
--
-- CHANGE MASTER TO MASTER_LOG_FILE='mysql-bin-changelog.000031',
 MASTER_LOG_POS=107;
```

对于 MySQL 8.0.23 及更高版本，将返回以下输出：

```
-- Position to start replication or point-in-time recovery from
--
-- CHANGE SOURCE TO SOURCE_LOG_FILE='mysql-bin-changelog.000031',
 SOURCE_LOG_POS=107;
```

对于 MySQL 8.0.22 及更低版本，如果您使用的是 SQL 格式，则可以在备份文件的 CHANGE MASTER TO 注释中获得主日志文件名称和位置。对于 MySQL 8.0.23 及更高版本，如果您使用的

是 SQL 格式，则可以在备份文件的 CHANGE REPLICATION SOURCE TO 注释中获得源日志文件名称和位置。如果外部实例是 MariaDB 10.0.24 或更高版本，则可在下一个步骤中获得 GTID。

2. 如果您使用的外部实例是 MariaDB 10.0.24 或更高版本，请使用基于 GTID 的复制。在外部 MariaDB 实例上运行 SHOW MASTER STATUS 以获取二进制日志文件名称和位置，然后通过在外部 MariaDB 实例上运行 BINLOG_GTID_POS 将它们转换为 GTID。

```
SELECT BINLOG_GTID_POS('binary_log_file_name', binary_log_file_position);
```

请记下返回的 GTID。您需要此 GTID 来配置复制。

3. 压缩复制后的数据可减少将数据复制到 Amazon RDS 数据库所需的网络资源量。记下备份文件的大小。您在确定要创建的 Amazon EC2 实例的大小时需要此信息。完成操作后，请使用 GZIP 或您的首选压缩实用程序来压缩备份文件。

   • 要压缩 SQL 输出，请使用以下命令：

   ```
   gzip backup.sql
   ```

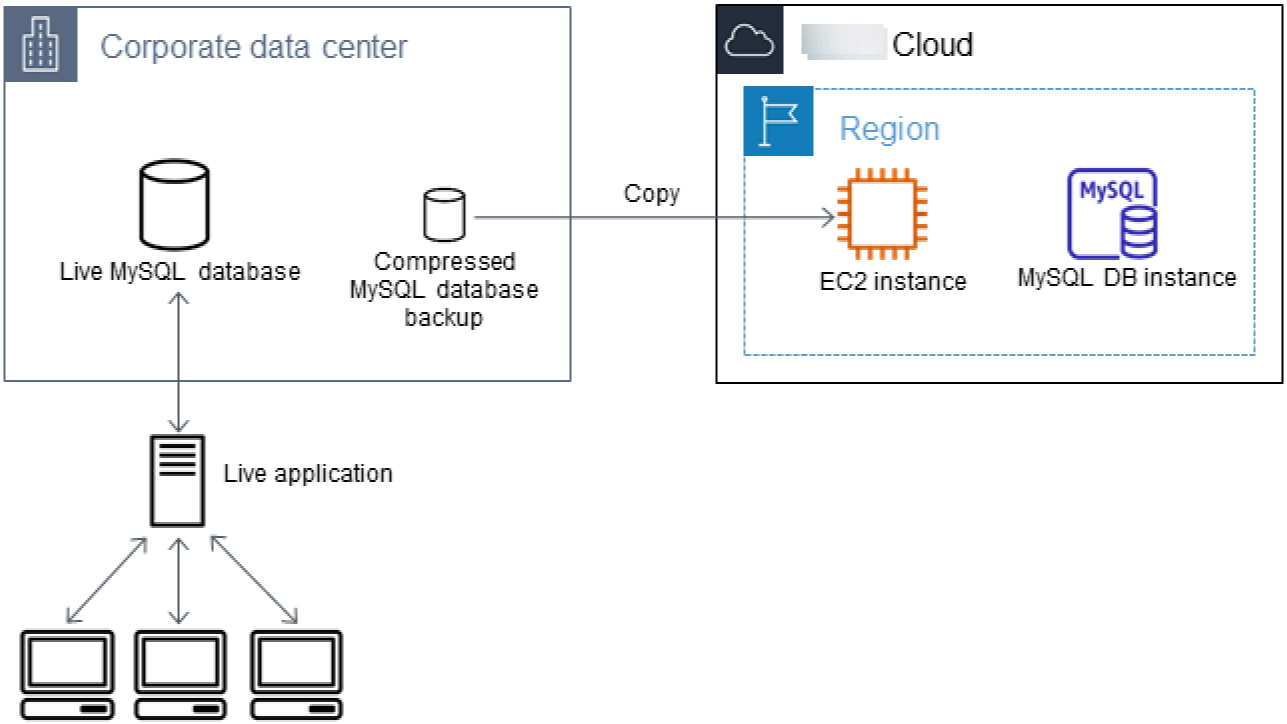
   • 要压缩分隔文本输出，请使用以下命令：

   ```
   tar -zcvf backup.tar.gz target_directory
   ```

任务 2：创建 Amazon EC2 实例和复制压缩的数据库

将压缩的数据库备份文件复制到 Amazon EC2 实例所占用的网络资源少于直接在数据库实例之间复制未压缩的数据所占用的网络资源。当您的数据位于 Amazon EC2 中后，可以直接从中将数据复制到 MariaDB 或 MySQL 数据库。要节省网络资源成本，您的 Amazon EC2 实例必须与 Amazon RDS 数据库实例位于同一 Amazon Web Services 区域内。使 Amazon EC2 实例位于 Amazon RDS 数据库所在的相同 Amazon Web Services 区域内，还可以减少导入期间的网络延迟。

下图显示了将数据库备份复制到 Amazon EC2 实例。



创建 Amazon EC2 实例并复制数据

1. 在您打算在其中创建 Amazon RDS 数据库的 Amazon Web Services 区域中，创建虚拟私有云（VPC）、VPC 安全组和 VPC 子网。确保您的 VPC 安全组的入站规则允许应用程序连接到 Amazon 所需的 IP 地址。您可以指定一系列 IP 地址（例如 `203.0.113.0/24`）或另一个 VPC 安全组。可以使用 Amazon VPC 控制台来创建和管理 VPC、子网和安全组。有关更多信息，请参阅《Amazon Virtual Private Cloud 用户指南》中的 Amazon VPC 入门。
2. 打开 Amazon EC2 控制台，并选择将包含 Amazon EC2 实例和 Amazon RDS 数据库的 Amazon Web Services 区域。使用您在步骤 1 中创建的 VPC、子网和安全组来启动 Amazon EC2 实例。在数据库备份文件未压缩的情况下，确保为该文件选择具有足够存储空间的实例类型。有关 Amazon EC2 实例的详细信息，请参阅《Amazon Elastic Compute Cloud 用户指南》中的开始使用 Amazon EC2。
3. 要从 Amazon EC2 实例连接到 Amazon RDS 数据库，请编辑 VPC 安全组。添加指定 EC2 实例的私有 IP 地址的入站规则。您可以在 EC2 控制台窗口中的 Instance 窗格的 Details 选项卡上找到私有 IP 地址。要编辑 VPC 安全组并添加入站规则，请在 EC2 控制台导航窗格中选择 Security Groups（安全组），然后为 MySQL 或 Aurora 添加指定 EC2 实例的私有 IP 地址的入站规则。要了解如何向 VPC 安全组添加入站规则，请参阅《Amazon Virtual Private Cloud 用户指南》中的安全组规则。

4. 将压缩后的数据库备份文件从本地系统复制到 Amazon EC2 实例中。如有必要，可使用 chmod 来确保您具有 Amazon EC2 实例目标目录的写入权限。您可以使用 scp 或 Secure Shell (SSH) 客户端复制文件。以下命令是一个示例 scp 命令：

```
scp -r -i key pair.pem backup.sql.gz ec2-user@EC2 DNS:/target_directory/backup.sql.gz
```

> **Important**
>
> 复制敏感数据时，务必使用安全网络传输协议。

5. 连接到 Amazon EC2 实例，并使用以下命令安装最新更新和 MySQL 客户端工具：

```
sudo yum update -y
sudo yum install mysql -y
```

有关更多信息，请参阅《Amazon Elastic Compute Cloud 用户指南》中的连接到您的实例（适用于 Linux 实例）。

> **Important**
>
> 此示例介绍了如何在适用于 Amazon Linux 发行版的 Amazon Machine Image (AMI) 上安装 MySQL 客户端。此示例不在其它发行版（例如 Ubuntu 或 Red Hat Enterprise Linux）上安装 MySQL 客户端。有关安装 MySQL 的信息，请参阅 MySQL 文档中的 Installing MySQL。

6. 在连接到 Amazon EC2 实例时，请解压缩数据库备份文件。下面是一些命令示例。

- 要解压缩 SQL 输出，请使用以下命令：

```
gzip backup.sql.gz -d
```

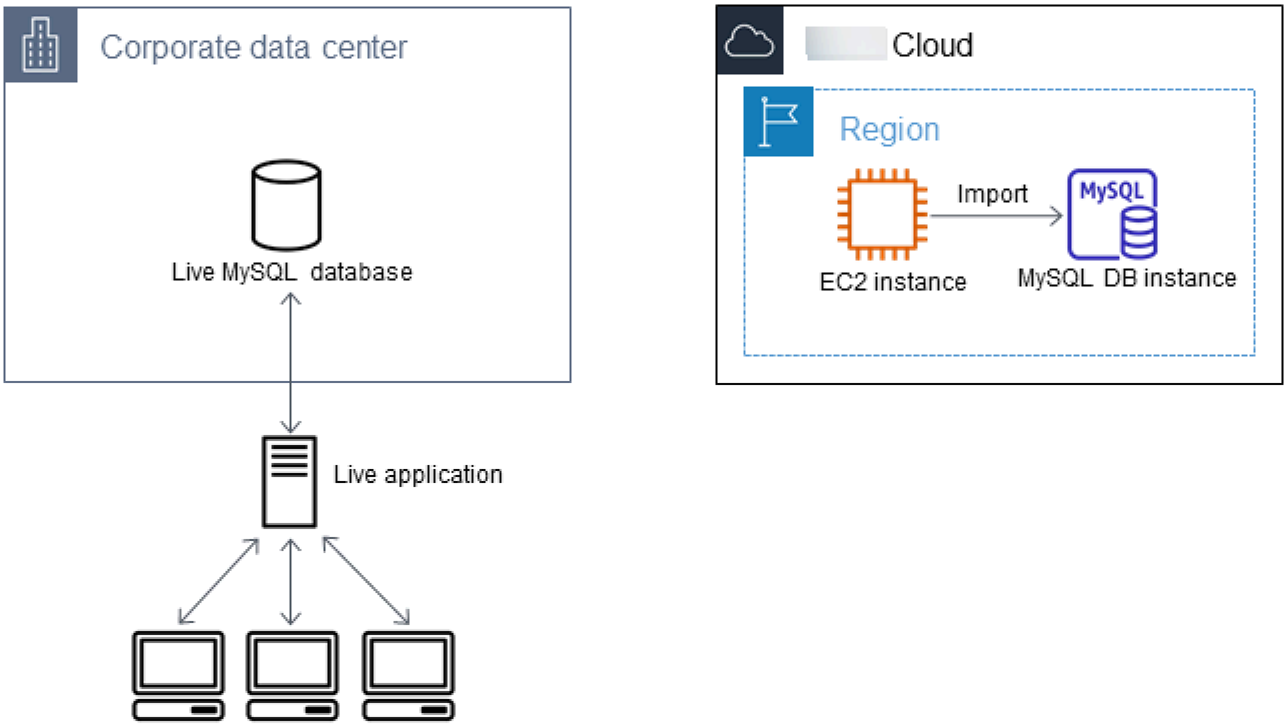
- 要解压缩分隔的文本输出，请使用以下命令：

```
tar xzvf backup.tar.gz
```

任务 3：创建 MySQL 或 MariaDB 数据库，并从 Amazon EC2 实例导入数据

通过在 Amazon EC2 实例所在的相同 Amazon Web Services 区域中创建 RDS for MariaDB 数据库实例、RDS for MySQL 数据库实例或 RDS for MySQL 多可用区数据库集群，与通过互联网相比，可以更快地从 Amazon EC2 导入数据库备份文件。

下图显示了将备份从 Amazon EC2 实例导入到 MySQL 数据库。



创建 MariaDB 或 MySQL 数据库并导入数据

1. 确定支持此 Amazon RDS 数据库的预期工作负载所需的数据库实例类和存储空间量。作为过程的一部分，确定数据加载过程所需的足够空间量和处理能力。此外，确定处理生产工作负载所需的项目。可以根据源 MariaDB 或 MySQL 数据库的大小和资源来进行估计。有关更多信息，请参阅 数据库实例类。
2. 在包含您的 Amazon EC2 实例的 Amazon Web Services 区域中创建数据库实例或多可用区数据库集群。

   要创建 RDS for MySQL 多可用区数据库集群，请按创建 Amazon RDS 的多可用区数据库集群中的说明操作。

   要创建 RDS for MariaDB 或 RDS for MySQL 数据库实例，请按照创建 Amazon RDS 数据库实例中的说明操作并使用以下准则：

- 指定与源数据库实例兼容的数据库引擎版本。
  - 如果源实例是 MySQL 5.5.x，则 Amazon RDS 数据库实例必须是 MySQL。
  - 如果源实例是 MySQL 5.6.x 或 5.7.x，则 Amazon RDS 数据库实例必须是 MySQL 或 MariaDB。
  - 如果源实例是 MySQL 8.0.x，则 Amazon RDS 数据库实例必须是 MySQL 8.0.x。
  - 如果源实例是 MySQL 8.4.x，则 Amazon RDS 数据库实例必须是 MySQL 8.4.x。
  - 如果源实例是 MariaDB 5.5 或更高版本，则 Amazon RDS 数据库实例必须是 MariaDB。
- 指定用于 Amazon EC2 实例的相同虚拟私有云（VPC）和 VPC 安全组。此方式将确保 Amazon EC2 实例和 Amazon RDS 实例通过网络相互可见。确保您的数据库实例可公开访问。要按下一节所述设置源数据库的复制，您的数据库实例必须可公开访问。
- 在导入数据库备份之前，不要配置多个可用区、备份保留或只读副本。在完成该导入操作后，您可以为生产实例配置多可用区和备份保留期。

3. 检查 Amazon RDS 数据库的默认配置选项。如果数据库的默认参数组没有所需的配置选项，请查找具有这些选项的参数组，或创建新的参数组。有关创建参数组的详细信息，请参阅 Amazon RDS 的参数组。
4. 以主用户身份连接到新 Amazon RDS 数据库。创建为需要访问数据库实例的管理员、应用程序和服务提供支持所需的用户。Amazon RDS 数据库的主机名为此数据库实例的端点值（不包含端口号），例如 `mysampledb.123456789012.us-west-2.rds.amazonaws.com`。您可以在 Amazon RDS 控制台上的数据库详细信息中找到端点值。
5. 连接到 Amazon EC2 实例。有关更多信息，请参阅《Amazon Elastic Compute Cloud 用户指南》中的连接到您的实例（适用于 Linux 实例）。
6. 使用 `mysql` 命令以远程主机身份从 Amazon EC2 实例连接到 Amazon RDS 数据库。以下命令是一个示例：

   ```
   mysql -h host_name -P 3306 -u db_master_user -p
   ```

   host_name 是 Amazon RDS 数据库端点。
7. 在 `mysql` 提示符下，运行 `source` 命令并向它传递您的数据库转储文件名。此命令将数据加载到 Amazon RDS 数据库实例。

   - 对于 SQL 格式，请使用以下命令：

     ```
     mysql> source backup.sql;
     ```

- 对于分隔文本格式，请先创建数据库（如果它不是您在设置 Amazon RDS 数据库时创建的默认数据库）。

  ```
  mysql> create database database_name;
  mysql> use database_name;
  ```

  然后，创建表。

  ```
  mysql> source table1.sql
  mysql> source table2.sql
  etc...
  ```

  然后，导入数据。

  ```
  mysql> LOAD DATA LOCAL INFILE 'table1.txt' INTO TABLE table1 FIELDS TERMINATED BY
   ',' ENCLOSED BY '"' LINES TERMINATED BY '0x0d0a';
  mysql> LOAD DATA LOCAL INFILE 'table2.txt' INTO TABLE table2 FIELDS TERMINATED BY
   ',' ENCLOSED BY '"' LINES TERMINATED BY '0x0d0a';
  etc...
  ```

  要提高性能，您可以从多个连接并行执行这些操作，以便同时创建并加载所有表。

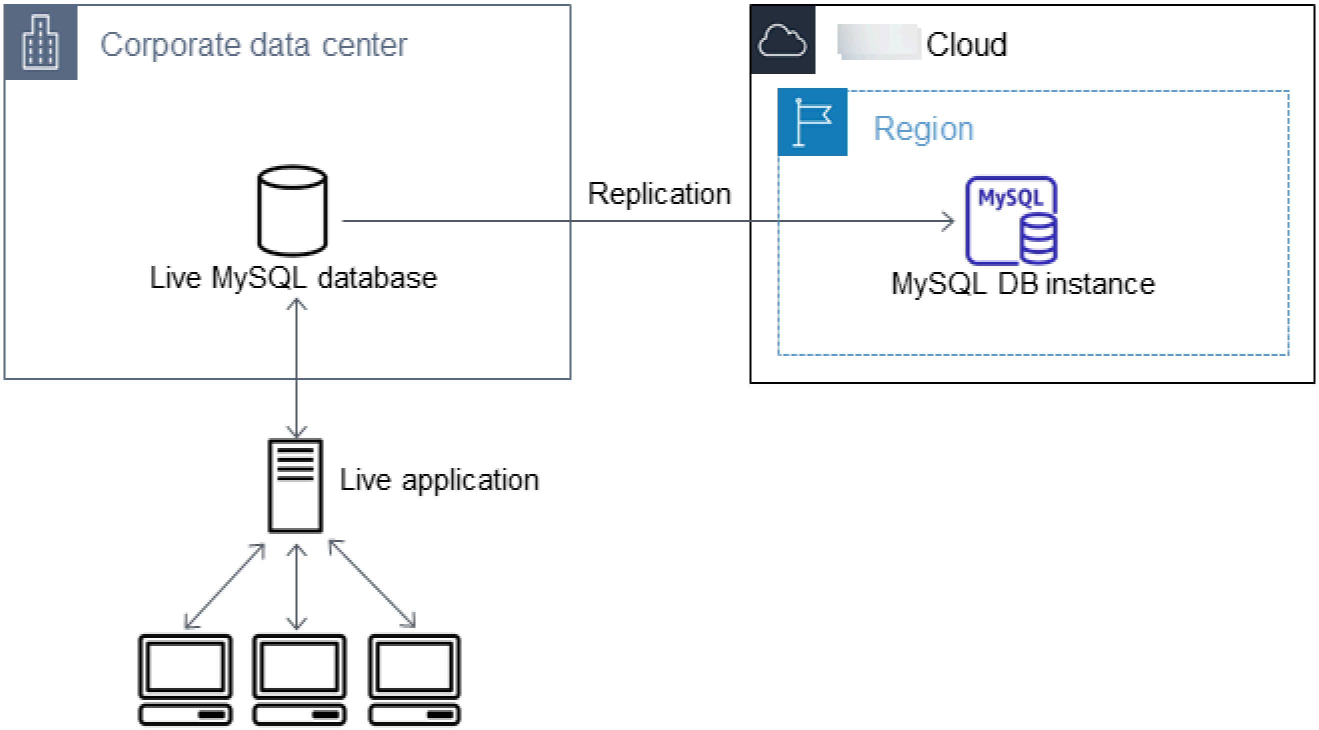
  > **Note**
  >
  > 如果您在最初转储表时已将任何数据格式化选项与 `mysqldump` 结合使用，则务必将相同的选项与 `LOAD DATA LOCAL INFILE` 结合使用，来确保正确地解读数据文件内容。

8. 对导入的数据库中的一个或两个表运行简单 `SELECT` 查询，来验证导入是否成功。

如果您不再需要该过程中使用的 Amazon EC2 实例，请终止 EC2 实例以减少Amazon资源使用。要终止 EC2 实例，请参阅《Amazon Elastic Compute Cloud 用户指南》中的终止实例。

任务 4：将数据从外部数据库复制到新的 Amazon RDS 数据库

在将数据复制并传输到 MariaDB 或 MySQL 数据库期间，您的源数据库可能已更新。这样，您可以使用复制功能使复制的数据库与源数据库保持同步。



对 Amazon RDS 数据库启动复制功能所需的权限受到限制且对 Amazon RDS 主用户不可用。因此，请使用适用于您的主要引擎版本的 Amazon RDS 存储过程：

- mysql.rds_set_external_master（RDS for MariaDB 和 RDS for MySQL 主要版本 8.0 及更低版本）
- mysql.rds_set_external_source（RDS for MySQL 主要版本 8.4 及更高版本）
- mysql.rds_set_external_master_gtid（配置复制）和 mysql.rds_start_replication（开始复制）

启动复制

在任务 1 中，当您设置复制选项时，您开启了二进制日志记录，并为源数据库设置了唯一服务器 ID。现在您可以将 Amazon RDS 数据库设置为副本，并将活动数据库作为源数据库实例。

1. 在 Amazon RDS 控制台中，将托管源数据库的服务器的 IP 地址添加到 Amazon RDS 数据库的 VPC 安全组。有关配置 VPC 安全组的更多信息，请参阅《Amazon Virtual Private Cloud 用户指南》中的配置安全组规则。

   您可能还需要配置本地网络，以允许来自 Amazon RDS 数据库的 IP 地址的连接，以便它能与源实例进行通信。要查找 Amazon RDS 数据库的 IP 地址，请使用 `host` 命令：

   ```
   host host_name
   ```

host_name 是 Amazon RDS 数据库端点中的 DNS 名称，例如 myinstance.123456789012.us-east-1.rds.amazonaws.com。您可以在 Amazon RDS 控制台上的数据库实例详细信息中找到端点值。

2. 通过使用所选的客户端，连接到源实例并创建将用于复制的用户。此账户仅用于复制，并且必须仅供您的域使用以增强安全性。以下命令是一个示例：

```
CREATE USER 'repl_user'@'mydomain.com' IDENTIFIED BY 'password';
```

> **Note**
>
> 作为安全最佳实践，请指定除此处所示提示以外的凭证。

3. 对于源 实例，向复制用户授予 REPLICATION CLIENT 和 REPLICATION SLAVE 权限。例如，要为您的域的“repl_user”用户授予对所有数据库的 REPLICATION CLIENT 和 REPLICATION SLAVE 权限，请发出以下命令。

```
GRANT REPLICATION CLIENT, REPLICATION SLAVE ON *.* TO 'repl_user'@'mydomain.com';
```

4. 如果您使用 SQL 格式创建备份文件，并且外部实例不是 MariaDB 10.0.24 或更高版本，请通过运行以下命令查看该文件的内容：

```
cat backup.sql
```

该文件包括 CHANGE MASTER TO 注释，其中包含主日志文件名称和位置。在将 --master-data 选项与 mysqldump 结合使用时，此注释将包含在备份文件中。请注意 MASTER_LOG_FILE 和 MASTER_LOG_POS 的值。

```
--
-- Position to start replication or point-in-time recovery from
--

-- CHANGE MASTER TO MASTER_LOG_FILE='mysql-bin-changelog.000031', MASTER_LOG_POS=107;
```

如果您使用分隔文本格式创建备份文件，并且外部实例不是 MariaDB 10.0.24 或更高版本，您应该已在任务 1 下“创建现有数据库的备份副本”过程的步骤 1 中获得二进制日志坐标。

如果外部实例是 MariaDB 10.0.24 或更高版本，您应该已在任务 1 下“创建现有数据库的备份副本”过程的步骤 2 中获得作为复制起点的 GTID。

5. 将 Amazon RDS 数据库设置为副本。如果外部实例不是 MariaDB 10.0.24 或更高版本，请使用适当的 Amazon RDS 存储过程，以主用户身份连接到 Amazon RDS 数据库，并将源数据库标识为源复制实例。

   - mysql.rds_set_external_master（RDS for MariaDB 和 RDS for MySQL 主要版本 8.0 及更低版本）
   - mysql.rds_set_external_source（RDS for MySQL 主要版本 8.4 及更高版本）

   如果您具有 SQL 格式的备份文件，则使用您在步骤 4 中确定的主日志文件名和主日志位置。如果您使用的是分隔文本格式，则使用您在创建备份文件时确定的名称和位置。下面是一些命令示例：

   MySQL 8.4 及更高版本

```
CALL mysql.rds_set_external_source ('myserver.mydomain.com', 3306,
    'repl_user', 'password', 'mysql-bin-changelog.000031', 107, 1);
```

   MariaDB 以及 MySQL 8.0 和更低版本

```
CALL mysql.rds_set_external_master ('myserver.mydomain.com', 3306,
    'repl_user', 'password', 'mysql-bin-changelog.000031', 107, 1);
```

   Note

   作为安全最佳实践，请指定除此处所示提示以外的凭证。

   如果外部实例是 MariaDB 10.0.24 或更高版本，请使用 mysql.rds_set_external_master_gtid 命令以主用户身份连接到 Amazon RDS 数据库，并将源数据库标识为源复制实例。使用您在任务 1 下“创建现有数据库的备份副本”过程的步骤 2 中确定的 GTID。以下命令是一个示例：

```
CALL mysql.rds_set_external_master_gtid ('source_server_ip_address', 3306,
 'ReplicationUser', 'password', 'GTID', 1);
```

   source_server_ip_address 是源复制实例的 IP 地址。当前不支持 EC2 私有 DNS 地址。

> **Note**
>
> 作为安全最佳实践，请指定除此处所示提示以外的凭证。

6. 在 Amazon RDS 数据库上，要开始复制，请运行以下 mysql.rds_start_replication 命令：

```
CALL mysql.rds_start_replication;
```

7. 在 Amazon RDS 数据库上，要确定副本何时与源复制实例实现同步，请运行 SHOW REPLICA STATUS 命令。`SHOW REPLICA STATUS` 命令的结果包括 `Seconds_Behind_Master` 字段。当 `Seconds_Behind_Master` 字段返回 0 时，副本将与源复制实例保持同步。

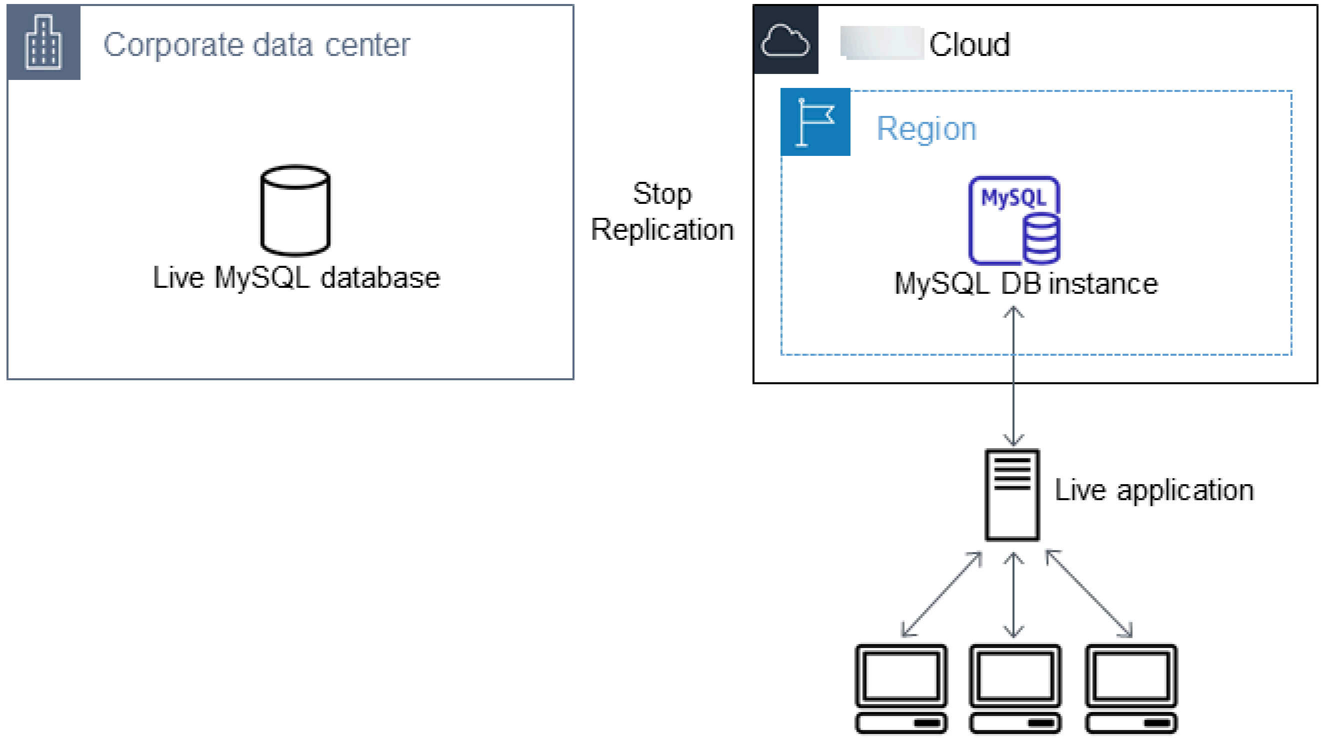
> **Note**
>
> 以前的 MySQL 版本使用的是 `SHOW SLAVE STATUS`，而不是 `SHOW REPLICA STATUS`。如果您使用的 MySQL 版本低于 8.0.23，那么请使用 `SHOW SLAVE STATUS`。

   对于 MariaDB 10.5、10.6 或 10.11 数据库实例，请运行 mysql.rds_replica_status 存储过程而不是 MySQL 命令。

8. 在 Amazon RDS 数据库同步后，开启自动备份以便您可以在需要时还原该数据库。您可以使用 Amazon RDS 控制台开启或修改 Amazon RDS 数据库的自动备份。有关更多信息，请参阅 备份简介。

任务 5：将您的活动应用程序重定向到 Amazon RDS 实例

在 MariaDB 或 MySQL 数据库与源复制实例保持同步后，您现在可以更新活动的应用程序以使用 Amazon RDS 实例。



将活动应用程序重新导向到 MariaDB 或 MySQL 数据库并停止复制

1. 要添加 Amazon RDS 数据库的 VPC 安全组，请添加托管应用程序的服务器的 IP 地址。有关修改 VPC 安全组的更多信息，请参阅《Amazon Virtual Private Cloud 用户指南》中的配置安全组规则。
2. 验证 SHOW REPLICA STATUS 命令结果中的 `Seconds_Behind_Master` 字段是否为 0，这表示副本与源复制实例保持同步。

```
SHOW REPLICA STATUS;
```

> **Note**
>
> 以前的 MySQL 版本使用的是 `SHOW SLAVE STATUS`，而不是 `SHOW REPLICA STATUS`。如果您使用的 MySQL 版本低于 8.0.23，那么请使用 `SHOW SLAVE STATUS`。

   对于 MariaDB 10.5、10.6 或 10.11 数据库实例，请运行 mysql.rds_replica_status 过程而不是 MySQL 命令。
3. 在事务完成后关闭与源的所有连接。
4. 更新应用程序以使用 Amazon RDS 数据库。此更新通常涉及更改连接设置以标识 Amazon RDS 数据库的主机名和端口、用于进行连接的用户账户和密码以及要使用的数据库。

5. 连接到数据库实例。

   对于多可用区数据库集群，连接至写入器数据库实例。
6. 使用以下 mysql.rds_stop_replication 命令停止 Amazon RDS 实例的复制：

```
CALL mysql.rds_stop_replication;
```

7. 在 Amazon RDS 数据库上运行相应的 Amazon RDS 存储过程以重置复制配置，以便不再将该实例标识为副本。
   - mysql.rds_reset_external_master（RDS for MariaDB 和 RDS for MySQL 主要版本 8.0 及更低版本）
   - mysql.rds_reset_external_source（RDS for MySQL 主要版本 8.4 及更高版本）

   MySQL 8.4 及更高版本

```
CALL mysql.rds_reset_external_source;
```

   MariaDB 以及 MySQL 8.0 和更低版本

```
CALL mysql.rds_reset_external_master;
```

8. 开启其他 Amazon RDS 特征，例如多可用区支持和只读副本。有关更多信息，请参阅配置和管理 Amazon RDS 的多可用区部署和使用数据库实例只读副本。

从任意源将数据导入 Amazon RDS for MariaDB 数据库实例

以下步骤将引导您完成将数据导入到 Amazon RDS 数据库实例的过程：

步骤 1：创建包含待加载数据的平面文件

使用通用格式存储待加载的数据，如逗号分隔值 (CSV)。每个表都必须拥有其自己的文件，您不能将多个表的数据合并到同一文件。每个文件的名称都必须与对应的表相同。可以根据个人喜好设置文件扩展名。例如，如果表名称为 `sales`，文件名可能为 `sales.csv` 或 `sales.txt`。

如果可能，请按照正在加载的表的主键对数据进行排序。这样做可以显著优化加载时间并尽可能减少磁盘存储要求。

该过程的速度和效率取决于能否将文件的大小保持在较小状态。如果在未压缩状态下单个文件的大小超过 1 GiB，则需将此文件分割为多个文件，然后分别加载各个文件。

在类 Unix 的系统（包括 Linux）中，请使用 `split` 命令。例如，以下命令会将 `sales.csv` 文件分割为多个小于 1 GiB 的文件，且只在换行符处进行分割 (-C 1024m)。新文件的名称包括升序数字后缀。以下命令生成名称如 `sales.part_00` 和 `sales.part_01` 的文件。

```
split -C 1024m -d sales.csv sales.part_
```

其他操作系统也具有类似的实用程序。

您可以将平面文件存储在任何地方。但是，当在步骤 5 中加载数据时，必须从文件存在的相同位置调用 `mysql` Shell，或者在运行 `LOAD DATA LOCAL INFILE` 时使用文件的绝对路径。

步骤 2：阻止任何应用程序访问目标数据库实例

开始大量加载前，阻止所有应用程序活动访问您打算加载到的目标数据库实例。建议您这样做，特别是当其他会话将要修改正在加载的表或引用的表时。这可以降低加载过程中发生约束冲突的风险并提高加载性能。它还可以将数据库实例还原到加载前的某一时间点，而不会丢失未参与加载的进程所做的更改。

当然，这可能并不可行或不现实。如果您无法在加载前阻止应用程序访问数据库实例，应采取步骤以确保数据的可用性和完整性。所需的具体步骤在很大程度上根据特定的使用案例和现场要求而有所不同。

步骤 3：创建数据库快照

如果您计划将数据加载到不包含数据的新数据库实例，可以跳过此步骤。否则，我们建议您在加载数据之前和之后创建目标 Amazon RDS 数据库实例的数据库快照。Amazon RDS 数据库快照是数据库实例的完整备份，可用于将数据库实例还原到已知状态。在启动数据库快照时，针对数据库实例的 I/O 操作将在备份数据库过程中暂停。

通过在即将开始加载前创建数据库快照，您可以在需要时将数据库还原到加载前的状态。通过使用在加载后立即创建的数据库快照，您可以在遇到问题时不必再次加载数据。还可以在加载后使用数据库快照将数据导入到新的数据库实例中。

以下示例运行 Amazon CLI [create-db-snapshot](https://docs.aws.amazon.com/cli/latest/reference/rds/create-db-snapshot.html) 命令来创建 `AcmeRDS` 实例的数据库快照，并为该数据库快照指定标识符 `"preload"`。

对于 Linux、macOS 或 Unix：

```
aws rds create-db-snapshot \
    --db-instance-identifier AcmeRDS \
```

```
    --db-snapshot-identifier preload
```

对于 Windows：

```
aws rds create-db-snapshot ^
    --db-instance-identifier AcmeRDS ^
    --db-snapshot-identifier preload
```

也可以使用从数据库快照还原功能，以创建用于试运行的测试数据库实例或撤消在加载期间进行的更改。

请务必注意，从数据库快照还原数据库将创建新的数据库实例，与所有数据库实例一样，该数据库实例具有唯一的标识符和端点。要还原数据库实例但不改变端点，请先删除该数据库实例，以使您能够重新使用此端点。

例如，要创建一个适用于试运行或其他测试的数据库实例，您可为数据库实例指定它自己的标识符。在示例中，标识符为 `AcmeRDS-2`"。该示例使用与 `AcmeRDS-2` 关联的端点连接到数据库实例。有关更多信息，请参阅 [restore-db-instance-from-db-snapshot]。

对于 Linux、macOS 或 Unix：

```
aws rds restore-db-instance-from-db-snapshot \
    --db-instance-identifier AcmeRDS-2 \
    --db-snapshot-identifier preload
```

对于 Windows：

```
aws rds restore-db-instance-from-db-snapshot ^
    --db-instance-identifier AcmeRDS-2 ^
    --db-snapshot-identifier preload
```

要重新使用现有的端点，请首先删除该数据库实例，然后为还原的数据库指定相同的标识符。有关更多信息，请参阅 [delete-db-instance]。

以下示例还在删除数据库实例之前为其制作最终的数据库快照。您可以自由选择，但建议您这样做。

对于 Linux、macOS 或 Unix：

```
aws rds delete-db-instance \
```

```
    --db-instance-identifier AcmeRDS \
    --final-db-snapshot-identifier AcmeRDS-Final

aws rds restore-db-instance-from-db-snapshot \
    --db-instance-identifier AcmeRDS \
    --db-snapshot-identifier preload
```

对于 Windows：

```
aws rds delete-db-instance ^
    --db-instance-identifier AcmeRDS ^
    --final-db-snapshot-identifier AcmeRDS-Final

aws rds restore-db-instance-from-db-snapshot ^
    --db-instance-identifier AcmeRDS ^
    --db-snapshot-identifier preload
```

步骤 4（可选）：关闭 Amazon RDS 自动备份

> **Warning**
>
> 如果需要执行时间点故障恢复，请勿关闭自动备份。

关闭自动备份是一种性能优化，加载数据时不需要这样做。关闭自动备份会清除所有现有备份。因此，在关闭自动备份后，将无法使用时间点故障恢复功能。关闭自动备份不会影响手动数据库快照。仍然可以还原所有现有的手动数据库快照。

关闭自动备份可减少约 25% 的加载时间，并减少加载期间所需的存储空间。如果您计划将数据加载到不包含任何数据的新数据库实例，关闭备份就可轻松加速加载并避免使用备份所需的额外存储空间。但是，在某些情况下，您可能会计划将数据加载到已包含数据的数据库实例。如果是这样，请权衡关闭备份的优势与无法执行时间点恢复功能的影响。

数据库实例默认开启自动备份 (保留期为一天)。要关闭自动备份，请将备份保留期设置为零。加载后，可以将备份保留期设置为非零值，重新开启自动备份。要开启或关闭备份，Amazon RDS 将关闭并重新启动数据库实例，以开启或关闭 MariaDB 或 MySQL 日志记录。

运行 Amazon CLI `modify-db-instance` 命令，将备份保留期设置为零并立即应用更改。要将保留期设置为零，需要重新启动数据库实例，因此，等到重新启动完成后再继续操作。有关更多信息，请参阅 modify-db-instance。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier AcmeRDS \
    --apply-immediately \
    --backup-retention-period 0
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier AcmeRDS ^
    --apply-immediately ^
    --backup-retention-period 0
```

您可以使用 Amazon CLI describe-db-instances 命令检查数据库实例的状态。以下示例显示 AcmeRDS 数据库实例的数据库实例状态：

```
aws rds describe-db-instances --db-instance-identifier AcmeRDS --query "*[].
{DBInstanceStatus:DBInstanceStatus}"
```

当数据库实例状态为 available 时，您已准备好进行下一步。

步骤 5：加载数据

要将平面文件中的行读取到数据库表中，请使用 MySQL LOAD DATA LOCAL INFILE 语句。

> **Note**
>
> 必须从平面文件存在的相同位置调用 mysql Shell，或者在运行 LOAD DATA LOCAL INFILE 时使用文件的绝对路径。

以下示例说明如何将数据从名为 sales.txt 的文件加载到数据库中名为 Sales 的表中：

```
mysql> LOAD DATA LOCAL INFILE 'sales.txt' INTO TABLE Sales FIELDS TERMINATED BY ' '
 ENCLOSED BY '' ESCAPED BY '\\';
Query OK, 1 row affected (0.01 sec)
Records: 1  Deleted: 0  Skipped: 0  Warnings: 0
```

有关 LOAD DATA 语句的更多信息，请参阅 MySQL 文档中的 LOAD DATA Statement。

步骤 6：重新开启 Amazon RDS 自动备份

如果您在步骤 4 中关闭了 Amazon RDS 自动备份，则在加载完成后，通过将备份保留期重新设置为其预加载值来开启自动备份。如步骤 4 中所述，Amazon RDS 会重新启动数据库实例，因此，请为短暂中断做好准备。

以下示例运行 Amazon CLI [modify-db-instance](https://docs.aws.amazon.com/cli/latest/reference/rds/modify-db-instance.html) 命令来开启 `AcmeRDS` 数据库实例的自动备份，并将保留期设置为一天：

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier AcmeRDS \
    --backup-retention-period 1 \
    --apply-immediately
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier AcmeRDS ^
    --backup-retention-period 1 ^
    --apply-immediately
```

在 Amazon RDS 中使用 MariaDB 复制

一般使用只读副本配置 Amazon RDS 数据库实例之间的复制。有关只读副本的一般信息，请参阅使用数据库实例只读副本。有关使用 Amazon RDS for MariaDB 上的只读副本的特定信息，请参阅使用 MariaDB 只读副本。

还可以为 MariaDB 数据库实例配置基于二进制日志坐标的复制。对于 MariaDB 实例，您也可以配置基于复制的全局事务 ID (GTID)，它将提供更高的崩溃安全性。有关更多信息，请参阅 配置与外部源实例之间的基于 GTID 的复制。

下面是可用于 RDS for MariaDB 的其他复制选项：

- 您可以设置 RDS for MariaDB 数据库实例与 Amazon RDS 外部的 MySQL 或 MariaDB 实例之间的复制。有关配置外部源复制的信息，请参阅配置与外部源实例之间的二进制日志文件位置复制。
- 可以配置复制以导入来自 Amazon RDS 外部的 MySQL 或 MariaDB 实例的数据库，或将数据库导出至此类实例。有关更多信息，请参阅 将数据导入到 Amazon RDS for MariaDB 数据库实例并减少停机时间 和 使用复制从 MySQL 数据库实例中导出数据。

对于其中任何复制选项，您可以使用基于行的复制、基于语句的复制或混合复制。基于行的复制仅复制因 SQL 语句而导致更改的行。基于语句的复制将复制整个 SQL 语句。混合复制尽可能使用基于语句的复制，但当运行对于基于语句的复制不安全的 SQL 语句时切换到基于行的复制。在大多数情况下，建议使用混合复制。数据库实例的二进制日志格式确定复制是基于行、基于语句还是混合的。有关设置二进制日志格式的信息，请参阅配置 MariaDB 二进制日志记录。

有关 MariaDB 版本之间的复制兼容性的信息，请参阅 MariaDB 文档中的 Replication Compatibility。

主题

使用 MariaDB 只读副本

接下来，您可以找到有关使用 Amazon RDS for MariaDB 上的只读副本的特定信息。有关只读副本及其使用说明的一般信息，请参阅使用数据库实例只读副本。

- 使用 MariaDB 配置延迟复制
- 使用 MariaDB 更新只读副本
- 使用 MariaDB 处理多可用区只读副本部署
- 将级联只读副本用于 RDS for MariaDB
- 监控 MariaDB 只读副本
- 开始和停止 MariaDB 只读副本复制
- 排查 MariaDB 只读副本问题

使用 MariaDB 配置只读副本

确保先通过将备份保留期设置为一个非零值来在源数据库实例上启用自动备份，然后 MariaDB 数据库实例才能充当复制源。此要求也适用于作为一个只读副本的源数据库实例的另一个只读副本。

您可以从同一区域内的一个数据库实例创建多达 15 个只读副本。为了高效地进行复制，每个只读副本具有的计算和存储资源的量应与源数据库实例的一样多。如果扩展源数据库实例，则还应扩展只读副本。

RDS for MariaDB 支持级联只读副本。要了解如何配置级联只读副本，请参阅 将级联只读副本用于 RDS for MariaDB。

您可以在引用相同源数据库实例的同时，运行多个只读副本创建和删除操作。当您执行这些操作时，应将每个源实例的只读副本数限制在 15 个以内。

使用 MariaDB 配置复制筛选条件

您可以使用复制筛选条件来指定使用只读副本的数据库和表。复制筛选条件可以将数据库和表包含在复制之中或排除在复制之外。

以下是复制筛选条件的一些使用案例：

- 缩减只读副本的大小。使用复制筛选，您可以排除只读副本上不需要的数据库和表。
- 出于安全原因，将数据库和表从只读副本中排除。
- 在不同只读副本中为特定使用案例复制不同的数据库和表。例如，您可以使用特定的只读副本进行分析或分片。
- 对于在不同 Amazon Web Services 区域中具有只读副本的数据库实例，要在不同的 Amazon Web Services 区域中复制不同的数据库或表。

> **Note**
>
> 您还可以使用复制筛选条件来指定使用入站复制拓扑中主 MariaDB 数据库实例的数据库和表。有关此配置的更多信息，请参阅配置与外部源实例之间的二进制日志文件位置复制。

主题

设置 RDS for MariaDB 的复制筛选参数

要配置复制筛选条件，请在只读副本上设置以下复制筛选参数：

- `replicate-do-db` –将更改复制到指定的数据库。为只读副本设置此参数时，仅复制参数中指定的数据库。
- `replicate-ignore-db` –不将更改复制到指定的数据库。为只读副本设置 `replicate-do-db` 参数时，不会评估此参数。
- `replicate-do-table` –将更改复制到指定的表。为只读副本设置此参数时，仅复制参数中指定的表。此外，如果设置了 `replicate-do-db` 或 `replicate-ignore-db` 参数，则包含指定表的数据库必须包含在使用只读副本的复制中。
- `replicate-ignore-table` –不将更改复制到指定的表。为只读副本设置 `replicate-do-table` 参数时，不会评估此参数。
- `replicate-wild-do-table` – 根据指定的数据库和表名模式复制表。支持 `%` 和 `_` 通配符。设置 `replicate-do-db` 或 `replicate-ignore-db` 参数时，请确保包含指定表的数据库包含在使用只读副本的复制中。
- `replicate-wild-ignore-table` –不基于指定的数据库和表名模式复制表。支持 `%` 和 `_` 通配符。为只读副本设置 `replicate-do-table` 或 `replicate-wild-do-table` 参数时，不会评估此参数。

将按这些参数列出的顺序对其进行评估。有关这些参数如何运行的更多信息，请参阅 MariaDB 文档 。

默认情况下，这些参数中的每个参数都具有一个空值。在每个只读副本上，您可以使用这些参数来设置、更改和删除复制筛选条件。设置其中一个参数时，请用逗号将各筛选条件分开。

您可以在 % 和 _ 参数中使用 `replicate-wild-do-table` 和 `replicate-wild-ignore-table` 通配符。% 通配符可以匹配任意数量的字符，而 _ 通配符只能匹配一个字符。

源数据库实例的二进制日志记录格式对于复制非常重要，因为它决定了数据更改的记录。`binlog_format` 参数的设置将决定复制是基于行还是基于语句的复制。有关更多信息，请参阅"配置 MariaDB 二进制日志记录"。

> **Note**
> 无论源数据库实例上的 `binlog_format` 设置如何，所有数据定义语言 (DDL) 语句都将作为语句进行复制。

RDS for MariaDB 的复制筛选限制

以下限制适用于对 RDS for MariaDB 进行复制筛选：

- 每个复制筛选参数不得超过 2000 个字符。
- 复制筛选条件中不支持逗号。
- 不支持用于二进制日志筛选的 MariaDB `binlog_do_db` 和 `binlog_ignore_db` 选项。
- 复制筛选不支持 XA 事务。

  有关更多信息，请参阅 MySQL 文档中的XA 事务限制。
- RDS for MariaDB 版本 10.2 不支持复制筛选。

RDS for MariaDB 的复制筛选示例

要为只读副本配置复制筛选，请修改与只读副本关联的参数组中的复制筛选参数。

> **Note**
> 您无法修改默认参数组。如果只读副本使用默认参数组，请创建新的参数组并将其与只读副本关联。有关数据库参数组的更多信息，请参阅 Amazon RDS 的参数组。

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 在参数组中设置参数。有关设置参数的信息，请参阅 在 Amazon RDS 中修改数据库参数组中的参数。在参数组中设置参数时，与参数组关联的所有数据库实例都使用参数设置。如果在参数组中设置复制筛选参数，请确保参数组仅与只读副本关联。将源数据库实例的复制筛选参数留空。

以下示例使用 Amazon CLI 设置参数。这些示例将 `ApplyMethod` 设置为 `immediate`，以便在 CLI 命令完成后立即发生参数更改。如果希望在只读副本重新启动后应用待处理的更改，请将 `ApplyMethod` 设置为 `pending-reboot`。

以下示例设置了复制筛选条件：

- Including databases in replication
- Including tables in replication
- Including tables in replication with wildcard characters
- Escaping wildcard characters in names
- Excluding databases from replication
- Excluding tables from replication
- Excluding tables from replication using wildcard characters

Example 将数据库包含在复制之中

以下示例将 `mydb1` 和 `mydb2` 数据库包含在复制之内。为只读副本 `replicate-do-db` 设置时，仅复制参数中指定的数据库。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-parameter-group \
  --db-parameter-group-name myparametergroup \
  --parameters "[{"ParameterName": "replicate-do-db", "ParameterValue": "mydb1,mydb2",
 "ApplyMethod":"immediate"}]"
```

对于 Windows：

```
aws rds modify-db-parameter-group ^
  --db-parameter-group-name myparametergroup ^
  --parameters "[{"ParameterName": "replicate-do-db", "ParameterValue": "mydb1,mydb2",
 "ApplyMethod":"immediate"}]"
```

Example 将表包含在复制之中

以下示例将数据库 table1 中的 table2 和 mydb1 表包含在复制之中。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-parameter-group \
  --db-parameter-group-name myparametergroup \
  --parameters "[{"ParameterName": "replicate-do-table", "ParameterValue": 
 "mydb1.table1,mydb1.table2", "ApplyMethod":"immediate"}]"
```

对于 Windows：

```
aws rds modify-db-parameter-group ^
  --db-parameter-group-name myparametergroup ^
  --parameters "[{"ParameterName": "replicate-do-table", "ParameterValue": 
 "mydb1.table1,mydb1.table2", "ApplyMethod":"immediate"}]"
```

Example 使用通配符将表包含在复制之中

以下示例将数据库 orders 中名称以 returns 和 mydb 开头的表包含在复制之中。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-parameter-group \
  --db-parameter-group-name myparametergroup \
  --parameters "[{"ParameterName": "replicate-wild-do-table", "ParameterValue": 
 "mydb.orders%,mydb.returns%", "ApplyMethod":"immediate"}]"
```

对于 Windows：

```
aws rds modify-db-parameter-group ^
  --db-parameter-group-name myparametergroup ^
  --parameters "[{"ParameterName": "replicate-wild-do-table", "ParameterValue": 
 "mydb.orders%,mydb.returns%", "ApplyMethod":"immediate"}]"
```

Example 转义名称中的通配符

以下示例展示了如何使用转义字符 \ 来转义名称中的通配符。

假设数据库 mydb1 中有多个以 my_table 开头的表名，而且您希望将这些表包含在复制之中。表名包括下划线（下划线也是通配符），因此该示例将表名中的下划线转义。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-parameter-group \
  --db-parameter-group-name myparametergroup \
  --parameters "[{"ParameterName": "replicate-wild-do-table", "ParameterValue": "my
\_table%", "ApplyMethod":"immediate"}]"
```

对于 Windows：

```
aws rds modify-db-parameter-group ^
  --db-parameter-group-name myparametergroup ^
  --parameters "[{"ParameterName": "replicate-wild-do-table", "ParameterValue": "my
\_table%", "ApplyMethod":"immediate"}]"
```

Example 将数据库排除在复制之外

以下示例将 mydb1 和 mydb2 数据库排除在复制之外。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-parameter-group \
  --db-parameter-group-name myparametergroup \
  --parameters "[{"ParameterName": "replicate-ignore-db", "ParameterValue":
 "mydb1,mydb2", "ApplyMethod":"immediate"}]"
```

对于 Windows：

```
aws rds modify-db-parameter-group ^
  --db-parameter-group-name myparametergroup ^
  --parameters "[{"ParameterName": "replicate-ignore-db", "ParameterValue":
 "mydb1,mydb2", "ApplyMethod":"immediate"}]"
```

Example 将表排除在复制之外

以下示例将数据库 table1 中的表 table2 和 mydb1排除在复制之外。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-parameter-group \
  --db-parameter-group-name myparametergroup \
  --parameters "[{"ParameterName": "replicate-ignore-table", "ParameterValue":
 "mydb1.table1,mydb1.table2", "ApplyMethod":"immediate"}]"
```

对于 Windows：

```
aws rds modify-db-parameter-group ^
  --db-parameter-group-name myparametergroup ^
  --parameters "[{"ParameterName": "replicate-ignore-table", "ParameterValue":
 "mydb1.table1,mydb1.table2", "ApplyMethod":"immediate"}]"
```

Example 使用通配符将表排除在复制之外

以下示例将数据库 `orders` 中名称以 `returns` 和 mydb 开头的表排除在复制之外。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-parameter-group \
  --db-parameter-group-name myparametergroup \
  --parameters "[{"ParameterName": "replicate-wild-ignore-table", "ParameterValue":
 "mydb.orders%,mydb.returns%", "ApplyMethod":"immediate"}]"
```

对于 Windows：

```
aws rds modify-db-parameter-group ^
  --db-parameter-group-name myparametergroup ^
  --parameters "[{"ParameterName": "replicate-wild-ignore-table", "ParameterValue":
 "mydb.orders%,mydb.returns%", "ApplyMethod":"immediate"}]"
```

查看只读副本的复制筛选条件

您可以通过以下方式查看只读副本的复制筛选条件：

- 检查与只读副本关联的参数组中复制筛选参数的设置。

  有关说明，请参阅在 Amazon RDS 中查看数据库参数组的参数值。
- 在 MariaDB 客户端中，连接到只读副本并运行 `SHOW REPLICA STATUS` 语句。

在输出中，以下字段显示了只读副本的复制筛选条件：

- `Replicate_Do_DB`
- `Replicate_Ignore_DB`
- `Replicate_Do_Table`
- `Replicate_Ignore_Table`
- `Replicate_Wild_Do_Table`
- `Replicate_Wild_Ignore_Table`

有关这些字段的更多信息，请参阅 MySQL 文档中的检查复制状态。

> **Note**
>
> 以前的 MariaDB 版本使用的是 `SHOW SLAVE STATUS`，而不是 `SHOW REPLICA STATUS`。如果您使用的 MariaDB 版本低于 10.5，那么请使用 `SHOW SLAVE STATUS`。

使用 MariaDB 配置延迟复制

可以使用延迟复制作为灾难恢复策略。使用延迟复制，可指定最短的时间（以秒为单位）以延迟从源到只读副本的复制。如果出现灾难（如意外删除了表），可完成以下步骤以快速从灾难恢复：

- 在导致灾难的更改发送到它之前停止复制到只读副本。

  要停止复制，请使用 mysql.rds_stop_replication 存储过程。
- 按照将只读副本提升为独立的数据库实例中的说明，将只读副本提升为新的源数据库实例。

> **Note**
>
> - MariaDB 10.6 及更高版本支持延迟复制。
> - 使用存储过程配置延迟复制。无法使用Amazon Web Services Management Console、Amazon CLI 或 Amazon RDS API 配置延迟复制。
> - 您可以在延迟复制配置中使用基于全局事务标识符 (GTID) 的复制。

主题

- 在创建只读副本时配置延迟复制
- 修改现有只读副本的延迟复制
- 提升只读副本

在创建只读副本时配置延迟复制

要为未来通过数据库实例创建的任何只读副本配置延迟复制，请使用 mysql.rds_set_configuration 参数运行 `target delay` 存储过程。

在创建只读副本时配置延迟复制

1. 通过使用 MariaDB 客户端，以主用户身份连接到作为只读副本源的 MariaDB 数据库实例。
2. 使用 mysql.rds_set_configuration 参数运行 `target delay` 存储过程。

   例如，运行以下存储过程，针对从当前数据库实例创建的任何只读副本，指定将复制延迟至少一小时（3600 秒）。

   ```
   call mysql.rds_set_configuration('target delay', 3600);
   ```

   > **Note**
   > 运行此存储过程之后，将为使用 Amazon CLI 或 Amazon RDS API 创建的任何只读副本配置延迟指定秒数的复制。

修改现有只读副本的延迟复制

要修改现有只读副本的延迟复制，请运行 mysql.rds_set_source_delay 存储过程。

修改现有只读副本的延迟复制

1. 使用 MariaDB 客户端，以主用户的身份连接到只读副本。
2. 使用 mysql.rds_stop_replication 存储过程停止复制。
3. 运行 mysql.rds_set_source_delay 存储过程。

   例如，运行以下存储过程，指定复制到只读副本延迟至少一小时（3600 秒）。

   ```
   call mysql.rds_set_source_delay(3600);
   ```

4. 使用 mysql.rds_start_replication 存储过程开始复制。

提升只读副本

在灾难恢复方案中，当复制停止后，您可以将只读副本提升为新的源数据库实例。有关提升只读副本的信息，请参阅 将只读副本提升为独立的数据库实例。

使用 MariaDB 更新只读副本

只读副本旨在支持读取查询，但您可能需要偶尔进行更新。例如，您可能需要添加索引，以加快访问副本的特定类型的查询。您可通过在只读副本的数据库参数组中将 `read_only` 参数设置为 0 来启用更新。

使用 MariaDB 处理多可用区只读副本部署

您可从单可用区或多可用区数据库实例部署中创建只读副本。您可以使用多可用区部署提高关键数据的持久性和可用性，但无法使用多可用区为只读查询提供辅助服务。您可以改为从大流量、多可用区数据库实例创建只读副本以分载只读查询。如果多可用区部署的源实例故障转移到辅助可用区，则任何关联的只读副本都将自动切换为使用辅助可用区（现在为主可用区）作为其复制源。有关更多信息，请参阅“配置和管理 Amazon RDS 的多可用区部署”。

您可以将只读副本创建为多可用区数据库实例。Amazon RDS 会在另一个可用区中创建您的副本的备用，用于支持副本的故障转移。创建您的只读副本作为多可用区数据库实例与源数据库是否为多可用区数据库实例无关。

将级联只读副本用于 RDS for MariaDB

RDS for MariaDB 支持级联只读副本。使用级联只读副本，您可以扩展读取操作，而不会增加源 RDS for MariaDB 数据库实例的开销。

使用级联只读副本，您的 RDS for MariaDB 数据库实例会将数据发送到链中的第一个只读副本。然后，该只读副本将数据发送到链中的第二个副本，依此类推。最终结果是，链中的所有只读副本都具有来自 RDS for MariaDB 数据库实例的更改，但不会只在源数据库实例上产生开销。

您可以从源 RDS for MariaDB 数据库实例在链中创建一系列最多三个只读副本。例如，假设您具有 RDS for MariaDB 数据库实例 `mariadb-main`。您可执行以下操作：

- 从 `mariadb-main` 开始，创建链中的第一个只读副本 `read-replica-1`。
- 接下来，从 `read-replica-1`，创建链中的下一个只读副本 `read-replica-2`。

- 最后，从 read-replica-2，创建链中的第三个只读副本 read-replica-3。

除了 mariadb-main 系列中的第三个级联只读副本之外，您无法创建另一个只读副本。从 RDS for MariaDB 源数据库实例到一系列级联只读副本末尾的完整实例系列最多可以包含四个数据库实例。

为了使级联只读副本正常工作，每个源 RDS For MariaDB 数据库实例必须启用自动备份。要对只读副本开启自动备份，请先创建只读副本，然后修改只读副本以开启自动备份。有关更多信息，请参阅 创建只读副本。

与任何只读副本一样，您可以升级属于级联一部分的只读副本。从只读副本链中升级只读副本将从链中移除该副本。例如，假设您希望将一些工作负载从 mariadb-main 数据库实例转移到新实例，以仅供会计部门使用。假设该示例中的链有三个只读副本，您决定升级 read-replica-2。该链受到如下影响：

- 升级 read-replica-2 会将其从复制链中移除。
  - 现在它是一个完全读/写数据库实例。
  - 它继续复制到 read-replica-3，就像在升级之前那样。
- 您的 mariadb-main 继续复制到 read-replica-1。

有关升级只读副本的更多信息，请参阅将只读副本提升为独立的数据库实例。

监控 MariaDB 只读副本

对于 MariaDB 只读副本，您可以通过在 Amazon CloudWatch 中查看 Amazon RDS ReplicaLag 指标来监控复制滞后。ReplicaLag 指标报告 Seconds_Behind_Master 命令的 SHOW REPLICA STATUS 字段的值。

> **Note**
> 以前的 MariaDB 版本使用的是 SHOW SLAVE STATUS，而不是 SHOW REPLICA STATUS。如果您使用的 MariaDB 版本低于 10.5，那么请使用 SHOW SLAVE STATUS。

MariaDB 复制滞后的常见原因如下所示：

- 网络中断。
- 向只读副本上带索引的表写入。如果只读副本上的 read_only 参数未设置为 0，它可以中断复制。

- 使用 MyISAM 等非事务性存储引擎。仅 MariaDB 上的 InnoDB 存储引擎支持复制。

当 `ReplicaLag` 指标达到 0 时，即表示副本已赶上源数据库实例进度。如果 `ReplicaLag` 指标返回 -1，则当前未激活复制。`ReplicaLag` = -1 等效于 `Seconds_Behind_Master` = `NULL`。

开始和停止 MariaDB 只读副本复制

可通过调用系统存储过程 mysql.rds_stop_replication 和 mysql.rds_start_replication，在 Amazon RDS 数据库实例上停止再重新开始复制过程。对于长时间运行的操作 (如创建大型索引)，在两个 Amazon RDS 实例之间进行复制时可以这样做。在导入或导出数据库时，也需要停止再开始复制。有关更多信息，请参阅"将数据导入 Amazon RDS for MySQL 数据库实例并减少停机时间"和"使用复制从 MySQL 数据库实例中导出数据"。

如果复制连续 30 天停止，不论是手动还是由于复制错误，Amazon RDS 将结束源数据库实例与所有只读副本之间的复制。这样做是为了防止源数据库实例上的存储需求增长以及长故障转移时间。只读副本数据库实例仍可用。但是无法恢复复制，因为在结束复制后，已从源数据库实例中删除只读副本所需的二进制日志。您可为源数据库实例创建新的只读副本来重新建立复制。

排查 MariaDB 只读副本问题

MariaDB 的复制技术是异步的。由于它们是异步的，因此，源数据库实例上偶发的 `BinLogDiskUsage` 会增多，而只读副本上应有 `ReplicaLag`。例如，对源数据库实例的大量写入操作可以并行进行。与之对比的是，对只读副本的写入操作使用单个 I/O 线程串行进行，这会导致源实例与只读副本之间存在滞后。有关 MariaDB 文档中只读副本的更多信息，请转到复制概述。

您可通过多种方式来减少对源数据库实例的更新与对只读副本的后续更新之间的滞后，例如：

- 将只读副本的存储大小和数据库实例类调整到与源数据库实例类似。
- 确保源数据库实例和只读副本使用的数据库参数组中的参数设置相兼容。有关更多信息和示例，请参阅本部分后面的有关 `max_allowed_packet` 参数的讨论。

Amazon RDS 监控只读副本的复制状态，如果由于任何原因停止复制，则将只读副本实例的 `Replication State` 字段更新为 `Error`。可能会有这样的例子，在您的只读副本上运行的 DML 查询与对源数据库实例的更新冲突。

可通过查看 `Replication Error` 字段来检查 MariaDB 引擎引发的关联错误的详细信息。还生成指示只读副本状态的事件，包括 RDS-EVENT-0045、RDS-EVENT-0046 和 RDS-EVENT-0047。有关这些事件和事件订阅的详细信息，请参阅 使用 Amazon RDS 事件通知。如果返回 MariaDB 错误消息，则检查 MariaDB 错误消息文档中的错误。

一个可导致复制出错的常见问题是只读副本的 max_allowed_packet 参数的值小于源数据库实例的 max_allowed_packet 参数的值。max_allowed_packet 参数是可在数据库参数组中进行设置的自定义参数，用于指定可在数据库上运行的最大 DML 代码大小。有时候，与源数据库实例关联的数据库参数组中的 max_allowed_packet 参数值，要小于与源的只读副本关联的数据库参数组中的 max_allowed_packet 参数值。在这些情况下，复制过程可能会引发错误 (数据包大于 'max_allowed_packet' 字节) 并停止复制。通过将源和只读副本设置为使用具有相同 max_allowed_packet 参数值的数据库参数组，即可更正此错误。

其他可导致复制错误的常见情况包括：

- 对只读副本上的表进行写入操作。如果是在只读副本上创建索引，则需要将 read_only 参数设置为 0 才能创建索引。如果对只读副本上的表进行写入操作，则可能会中断复制。
- 使用非事务性存储引擎，如 MyISAM。只读副本需要使用事务性存储引擎。仅 MariaDB 上的 InnoDB 存储引擎支持复制。
- 使用不安全的不确定性查询，如 SYSDATE()。有关更多信息，请参阅确定二进制日志记录中的安全和不安全语句。

如果您确定可安全跳过错误，那么可以按照跳过 RDS for MySQL 的当前复制错误中描述的步骤操作。否则，可删除只读副本，然后使用相同的数据库实例标识符创建实例，以使终端节点保持与旧只读副本的终端节点相同。如果复制错误得到纠正，则 Replication State 将更改为 replicating。

对于 MariaDB 数据库实例，在某些情况下，如果某些二进制日志 (binlog) 事件在故障期间未刷新，则只读副本将无法切换到辅助可用区。在这些情况下，请手动删除并重新创建只读副本。您可通过设置以下参数值来降低发生这种情况的可能性：sync_binlog=1 和 innodb_flush_log_at_trx_commit=1。这些设置可能降低性能，因此，请先测试其影响，然后在生产环境中实施更改。

配置与外部源实例之间的基于 GTID 的复制

您可以将基于全局事务标识符 (GTID) 的复制从版本 10.0.24 或更高版本的外部 MariaDB 实例设置到 RDS for MariaDB 数据库实例中。对 Amazon RDS 设置外部源实例和副本时，请遵循以下准则：

- 监控作为副本的 RDS for MariaDB 数据库实例的故障转移事件。如果发生失效转移，则可能会在具有不同的网络地址的新主机上重新创建作为副本的数据库实例。有关如何监控故障转移事件的信息，请参阅使用 Amazon RDS 事件通知。
- 在源实例上保留二进制日志 (binlog)，直到您确认将其应用于副本。这种维护将确保您可以在发生故障时还原源实例。

- 在 Amazon RDS 上对 MariaDB 数据库实例启用自动备份。通过启用自动备份，可以确保在需要重新同步源实例和副本时能够将副本还原到特定时间点。有关备份和时间点还原的信息，请参阅备份、还原和导出数据。

> **Note**
>
> 对 MariaDB 数据库实例启动复制功能所需的权限受到限制，且对 Amazon RDS 主用户不可用。因此，您必须使用 Amazon RDS mysql.rds_set_external_master_gtid 和 mysql.rds_start_replication 命令来设置活动数据库和 RDS for MariaDB 数据库之间的复制。

要启动外部源实例和 Amazon RDS 上的 MariaDB 数据库实例之间的复制，请使用以下步骤。

启动复制

1. 将源 MariaDB 实例设为只读：

```
mysql> FLUSH TABLES WITH READ LOCK;
mysql> SET GLOBAL read_only = ON;
```

2. 获取外部 MariaDB 实例的当前 GTID。您可以通过使用 `mysql` 或您选择的查询编辑器运行 `SELECT @@gtid_current_pos;` 来执行该操作。

   GTID 采用 `<domain-id>-<server-id>-<sequence-id>` 格式。典型的 GTID 将类似于 **`0-1234510749-1728`**。有关 GTID 及其组成部分的更多信息，请参阅 MariaDB 文档中的全局事务 ID。

3. 使用 `mysqldump` 将数据库从外部 MariaDB 实例复制到 MariaDB 数据库实例。对于非常大的数据库，您可能需要使用将数据导入 Amazon RDS for MySQL 数据库实例并减少停机时间中的过程。

   对于 Linux、macOS 或 Unix：

```
mysqldump \
    --databases database_name \
    --single-transaction \
    --compress \
    --order-by-primary \
    -u local_user \
    -plocal_password | mysql \
        --host=hostname \
        --port=3306 \
```

```
        -u RDS_user_name \
        -pRDS_password
```

对于 Windows：

```
mysqldump ^
    --databases database_name ^
    --single-transaction ^
    --compress ^
    --order-by-primary \
    -u local_user \
    -plocal_password | mysql ^
        --host=hostname ^
        --port=3306 ^
        -u RDS_user_name ^
        -pRDS_password
```

> **Note**
>
> 确保 `-p` 选项和输入的密码之间没有空格。
> 作为安全最佳实践，请指定除此处所示提示以外的密码。

在 `--host` 命令中使用 `--user (-u)`、`--port`、`-p` 和 `mysql` 选项可指定用于连接到 MariaDB 数据库实例的主机名、用户名、端口和密码。主机名是 MariaDB 数据库实例终端节点中的 DNS 名称，例如，`myinstance.123456789012.us-east-1.rds.amazonaws.com`。您可以在 Amazon RDS 管理控制台上的实例详细信息中找到终端节点值。

4. 再次将源 MariaDB 实例设为可写。

```
mysql> SET GLOBAL read_only = OFF;
mysql> UNLOCK TABLES;
```

5. 在 Amazon RDS 管理控制台中，将承载外部 MariaDB 数据库的服务器的 IP 地址添加到 MariaDB 数据库实例的 VPC 安全组。有关修改 VPC 安全组的更多信息，请转到 Amazon Virtual Private Cloud 用户指南 中的[您的 VPC 的安全组]。

   在满足以下条件时，IP 地址可能会发生更改：

   - 您正在使用公有 IP 地址在外部源实例和数据库实例之间进行通信。

- 外部源实例已停止并重新启动。

如果满足这些条件，请在添加 IP 地址之前先对其进行验证。

您可能还需要配置本地网络，允许来自 MariaDB 数据库实例 IP 地址的连接，使它能与外部 MariaDB 实例进行通信。要查找 MariaDB 数据库实例的 IP 地址，请使用 `host` 命令。

```
host db_instance_endpoint
```

主机名是 MariaDB 数据库实例终端节点中的 DNS 名称。

6. 通过使用所选的客户端，连接到外部 MariaDB 实例并创建将用于复制的 MariaDB 用户。此账户仅用于复制，并且必须仅供您的域使用以增强安全性。示例如下：

```
CREATE USER 'repl_user'@'mydomain.com' IDENTIFIED BY 'password';
```

> **Note**
> 作为安全最佳实践，请指定除此处所示提示以外的密码。

7. 对于外部 MariaDB 实例，向复制用户授予 `REPLICATION CLIENT` 和 `REPLICATION SLAVE` 权限。例如，要为您的域的“`REPLICATION CLIENT`”用户授予对所有数据库的 `REPLICATION SLAVE` 和 `repl_user` 权限，请发出以下命令。

```
GRANT REPLICATION CLIENT, REPLICATION SLAVE ON *.* TO 'repl_user'@'mydomain.com';
```

8. 将 MariaDB 数据库实例设置为副本。通过使用 mysql.rds_set_external_master_gtid 命令，以主用户身份连接到 MariaDB 数据库实例，并将外部 MariaDB 数据库标识为复制源实例。使用您在步骤 2 中确定的 GTID。示例如下：

```
CALL mysql.rds_set_external_master_gtid ('mymasterserver.mydomain.com', 3306,
 'repl_user', 'password', 'GTID', 1);
```

> **Note**
> 作为安全最佳实践，请指定除此处所示提示以外的密码。

9. 在 MariaDB 数据库实例上，发出 mysql.rds_start_replication 命令以启动复制：

```
CALL mysql.rds_start_replication;
```

配置与外部源实例之间的二进制日志文件位置复制

您可以使用二进制日志文件复制来设置 RDS for MySQL 或 MariaDB 数据库实例与 Amazon RDS 外部的 MySQL 或 MariaDB 实例之间的复制。

主题

开始前的准备工作

您可以使用复制的事务的二进制日志文件位置配置复制。

对 Amazon RDS 数据库实例启动复制功能所需的权限受到限制且对 Amazon RDS 主用户不可用。因此，请确保使用 Amazon RDS mysql.rds_set_external_master（RDS for MariaDB 和 RDS for MySQL 主要版本 8.0 及更低版本）或 mysql.rds_set_external_source（RDS for MySQL 主要版本 8.4 及更高版本）以及 mysql.rds_start_replication 命令来设置活动数据库和 Amazon RDS 数据库之间的复制。

要为 MySQL 或 MariaDB 数据库设置二进制日志记录格式，请更新 `binlog_format` 参数。如果您的数据库实例使用默认数据库实例参数组，请创建一个新的数据库参数组来修改 `binlog_format` 参数。在 MariaDB 和 MySQL 8.0 及更低版本中，`binlog_format` 默认设置为 `MIXED`。不过，如果您需要特定的二进制日志 (binlog) 格式，也可以将 `binlog_format` 设置为 `ROW` 或 `STATEMENT`。重启您的数据库实例以使更改生效。在 MySQL 8.4 及更高版本中，`binlog_format` 默认设置为 `ROW`。

有关设置 `binlog_format` 参数的信息，请参阅为单可用区数据库配置 RDS for MySQL 二进制日志记录。有关不同 MySQL 复制类型的含义的信息，请参阅 MySQL 文档中的基于语句和基于行的复制的优点和缺点。

配置与外部源实例之间的二进制日志文件位置复制

对 Amazon RDS 设置外部源实例和副本时，请遵循以下准则：

- 监控作为副本的 Amazon RDS 数据库实例的失效转移事件。如果发生失效转移，则可能会在具有不同的网络地址的新主机上重新创建作为副本的数据库实例。有关如何监控失效转移事件的信息，请参阅使用 Amazon RDS 事件通知。

- 在您的源实例上维护二进制日志，直至您确认其已应用于副本。该维护确保在发生故障时可以还原源实例。
- 对 Amazon RDS 数据库实例启用自动备份。通过启用自动备份，可以确保在需要重新同步源实例和副本时能够将副本还原到特定时间点。有关备份和时间点还原的信息，请参阅备份、还原和导出数据。

配置与外部源实例之间的二进制日志文件复制

1. 将源 MySQL 或 MariaDB 实例设置为只读。

```
mysql> FLUSH TABLES WITH READ LOCK;
mysql> SET GLOBAL read_only = ON;
```

2. 对源 MySQL 或 MariaDB 实例运行 `SHOW MASTER STATUS` 命令以确定二进制日志位置。

   您将收到类似于以下示例的输出。

```
File                        Position
------------------------------------
 mysql-bin-changelog.000031      107
------------------------------------
```

3. 使用 `mysqldump` 将数据库从外部实例复制到 Amazon RDS 数据库实例。对于非常大的数据库，您可能需要使用将数据导入 Amazon RDS for MySQL 数据库实例并减少停机时间中的过程。

   对于 Linux、macOS 或 Unix：

```
mysqldump --databases database_name \
    --single-transaction \
    --compress \
    --order-by-primary \
    -u local_user \
    -plocal_password | mysql \
        --host=hostname \
        --port=3306 \
        -u RDS_user_name \
        -pRDS_password
```

   对于 Windows：

```
mysqldump --databases database_name ^
    --single-transaction ^
    --compress ^
    --order-by-primary ^
    -u local_user ^
    -plocal_password | mysql ^
        --host=hostname ^
        --port=3306 ^
        -u RDS_user_name ^
        -pRDS_password
```

Note

确保 -p 选项和输入的密码之间没有空格。

要指定主机名、用户名、端口和密码以连接到 Amazon RDS 数据库实例，请在 `--host` 命令中使用 `--user (-u)`、`--port`、`-p` 和 `mysql` 选项。主机名是 Amazon RDS 数据库实例端点中的域名服务 (DNS) 名称，例如 `myinstance.123456789012.us-east-1.rds.amazonaws.com`。您可以在 Amazon Web Services Management Console上的实例详细信息中找到端点值。

4. 再次将源 MySQL 或 MariaDB 实例设置为可写。

```
mysql> SET GLOBAL read_only = OFF;
mysql> UNLOCK TABLES;
```

有关生成备份以用于复制的更多信息，请参阅 MySQL 文档。

5. 在 Amazon Web Services Management Console 中，将托管外部数据库的服务器的 IP 地址添加到 Amazon RDS 数据库实例的 Virtual Private Cloud (VPC) 安全组中。有关修改 VPC 安全组的更多信息，请参阅 Amazon Virtual Private Cloud 用户指南 中的您的 VPC 的安全组。

在满足以下条件时，IP 地址可能会发生更改：

- 您正在使用公有 IP 地址在外部源实例和数据库实例之间进行通信。
- 外部源实例已停止并重新启动。

如果满足这些条件，请在添加 IP 地址之前先对其进行验证。

您可能还需要配置本地网络，以允许来自 Amazon RDS 数据库实例的 IP 地址的连接。通过执行此操作，您的本地网络可以与外部 MySQL 或 MariaDB 实例进行通信。要查找 Amazon RDS 数据库实例的 IP 地址，请使用 host 命令。

```
host db_instance_endpoint
```

主机名是 Amazon RDS 数据库实例端点中的 DNS 名称。

6. 通过使用所选的客户端，连接到外部实例并创建要用于复制的用户。请仅将该账户用于复制，并将其限制为您的域以提高安全性。示例如下：

```
CREATE USER 'repl_user'@'mydomain.com' IDENTIFIED BY 'password';
```

> **Note**
>
> 作为安全最佳实践，请指定除此处所示提示以外的密码。

7. 对于外部 实例，向复制用户授予 REPLICATION CLIENT 和 REPLICATION SLAVE 权限。例如，要为您的域的“REPLICATION CLIENT”用户授予对所有数据库的 REPLICATION SLAVE 和 repl_user 权限，请发出以下命令。

```
GRANT REPLICATION CLIENT, REPLICATION SLAVE ON *.* TO 'repl_user'@'mydomain.com';
```

8. 将 Amazon RDS 数据库实例设置为副本。为此，请先以主用户身份连接到 Amazon RDS 数据库实例。然后使用 mysql.rds_set_external_source（RDS for MySQL 主要版本 8.4 及更高版本）或 mysql.rds_set_external_master（RDS for MariaDB 和 RDS for MySQL 主要版本 8.0 及更低版本）命令，将外部 MySQL 或 MariaDB 数据库标识为源实例。使用在步骤 2 中确定的主日志文件名和主日志位置。下面是一些命令示例。

MySQL 8.4

```
CALL mysql.rds_set_external_source ('mysourceserver.mydomain.com', 3306,
 'repl_user', 'password', 'mysql-bin-changelog.000031', 107, 1);
```

MariaDB 以及 MySQL 8.0 和 5.7

```
CALL mysql.rds_set_external_master ('mymasterserver.mydomain.com', 3306,
 'repl_user', 'password', 'mysql-bin-changelog.000031', 107, 1);
```

> **Note**
>
> 在 RDS for MySQL 上，可以选择通过运行 mysql.rds_set_external_source_with_delay（RDS for MySQL 主要版本 8.4 及更高版本） 或 mysql.rds_set_external_master_with_delay（RDS for MariaDB 和 RDS for MySQL 主要版本 8.0 及更低版本） 存储过程来使用延迟复制。在 RDS for MySQL 中，使用延迟复制的一个原因是，使用 mysql.rds_start_replication_until 存储过程打开灾难恢复。目前，RDS for MariaDB 支持延迟复制，但不支持 `mysql.rds_start_replication_until` 过程。

9. 在 Amazon RDS 数据库实例上，发出 mysql.rds_start_replication 命令以启动复制。

```
CALL mysql.rds_start_replication;
```

适用于 MariaDB 数据库引擎的选项

在下文中，您可以找到适用于运行 MariaDB 数据库引擎的 Amazon RDS 实例的选项或其他功能的说明。要开启这些选项，可以将它们添加到自定义选项组中，然后将选项组与数据库实例相关联。有关使用选项组的详细信息，请参阅使用选项组。

Amazon RDS 支持 MariaDB 的以下选项：

选项 ID	引擎版本
MARIADB_AUDIT_PLUGIN	MariaDB 10.3 及更高版本

MariaDB 审计插件支持

Amazon RDS 支持在 MariaDB 数据库实例上使用 MariaDB 审核插件。MariaDB 审核插件可记录数据库活动，如用户登录数据库、针对数据库运行查询等。数据库活动记录存储在日志文件中。

审计插件选项设置

Amazon RDS 支持 MariaDB 审计插件选项的以下设置。

> **Note**
>
> 如果您未在 RDS 控制台中配置选项设置，RDS 将使用原定设置。

选项设置	有效值	默认值	描述
SERVER_AUDIT_FILE_PATH	/rdsdbdata/log/audit/	/rdsdbdata/log/audit/	日志文件的位置。日志文件包含 SERVER_AUDIT_EVENTS 中指定的活动记录。有关更多信息，请参阅“查看和列出数据库日志文件”和“MariaDB 数据库日志文件”。
SERVER_AUDIT_FILE_ROTATE_SIZE	1–1000000000	1000000	达到该字节大小时，会导致文件轮换。有关更多信息，请参阅“MariaDB 的日志轮换和保留”。

选项设置	有效值	默认值	描述
SERVER_AUDIT_FILE_ROTATIONS	0–100	9	当 server_audit_output_type=file 时要保存的日志轮换数量。如果设置为 0，则日志文件从不轮换。有关更多信息，请参阅MariaDB 的日志轮换和保留 和下载数据库日志文件。
SERVER_AUDIT_EVENTS	CONNECT, QUERY, TABLE, QUERY_DDL ，QUERY_DML ，QUERY_DML_NO_SELECT ，QUERY_DCL	CONNECT, QUERY	要在日志中记录的活动类型。安装 MariaDB 审核插件可自行登录。 • CONNECT：记录成功和失败的数据库连接以及数据库断开连接。 • QUERY：记录针对数据库运行的所有查询文本。 • TABLE：记录针对数据库运行查询时受查询影响的表。 • QUERY_DDL ：类似于 QUERY 事件，不过仅返回数据定义语言 (DDL) 查询（CREATE、ALTER 等）。 • QUERY_DML ：类似于 QUERY 事件，不过仅返回数据操作语言 (DML) 查询（INSERT、UPDATE 等，以及 SELECT）。 • QUERY_DML_NO_SELECT ：与 QUERY_DML 事件类似，但不会记录 SELECT 查询。 • QUERY_DCL ：类似于 QUERY 事件，不过仅返回数据控制语言 (DCL) 查询（GRANT、REVOKE 等）。

选项设置	有效值	默认值	描述
SERVER_AUDIT_INCL_USERS	多个逗号分隔值	无	仅包括指定用户的活动。默认情况下，会记录所有用户的活动。SERVER_AUDIT_INCL_USERS 和 SERVER_AUDIT_EXCL_USERS 是相互排斥的。如果将值添加到 SERVER_AUDIT_INCL_USERS，请确保未向 SERVER_AUDIT_EXCL_USERS 中添加任何值。
SERVER_AUDIT_EXCL_USERS	多个逗号分隔值	无	排除指定用户的活动。默认情况下，会记录所有用户的活动。SERVER_AUDIT_INCL_USERS 和 SERVER_AUDIT_EXCL_USERS 是相互排斥的。如果将值添加到 SERVER_AUDIT_EXCL_USERS，请确保未向 SERVER_AUDIT_INCL_USERS 中添加任何值。 rdsadmin 用户会每秒查询一次数据库，以检查数据库的运行状况。根据您的其他设置，此活动可能导致您的日志文件大小非常快速地增长。如果您不需要记录此活动，请将 rdsadmin 用户添加到 SERVER_AUDIT_EXCL_USERS 列表。 **Note** CONNECT会始终为所有用户记录 活动，即使是该选项设置指定的用户。
SERVER_AUDIT_LOGGING	ON	ON	日志记录处于活动状态。唯一有效值为 ON。Amazon RDS 不支持停用日志记录。如果您要停用日志记录，请删除 MariaDB 审核插件。有关更多信息，请参阅“删除 MariaDB 审核插件”。

选项设置	有效值	默认值	描述
SERVER_AUDIT_QUERY_LOG_LIMIT	0–2147483647	1024	记录中查询字符串的长度限制。

添加 MariaDB 审核插件

向数据库实例添加 MariaDB 审核插件的一般过程如下：

1. 创建新的选项组，或者复制或修改现有选项组。
2. 将 选项添加到该选项组。
3. 将选项组与数据库实例相关联。

添加 MariaDB 审核插件后，您无需重新启动数据库实例。一旦选项组处于活动状态，审核便会立即开始。

添加 MariaDB 审核插件

1. 确定您想要使用的选项组。您可以创建新的选项组，或使用现有选项组。如果您想使用现有选项组，请跳到下一步。否则，请创建自定义数据库选项组。对于 Engine（引擎），请选择 mariadb；对于 Major engine version（主引擎版本），请选择 10.3 或更高版本。有关更多信息，请参阅 创建选项组。
2. 将 MARIADB_AUDIT_PLUGIN 选项添加到选项组中，然后配置选项设置。有关添加选项的更多信息，请参阅 将选项添加到选项组。有关各项设置的更多信息，请参阅审计插件选项设置。
3. 将选项组应用到新的或现有的数据库实例。
   - 对于新数据库实例，您可以在启动实例时应用选项组。有关更多信息，请参阅“创建 Amazon RDS 数据库实例”。
   - 对于现有数据库实例，您可以通过修改数据库实例并附加新的选项组来应用选项组。有关更多信息，请参阅“修改 Amazon RDS 数据库实例”。

查看和下载 MariaDB 审计插件日志

启用 MariaDB 审计插件后，您将以访问其他任何基于文本的日志文件的相同方式来访问日志文件中的结果。审计日志文件位于 `/rdsdbdata/log/audit/`。有关在控制台中查看日志文件的信息，请参阅 查看和列出数据库日志文件。有关下载日志文件的信息，请参阅 下载数据库日志文件。

修改 MariaDB 审计插件设置

启用 MariaDB 审核插件后，您可以修改插件设置。有关如何修改选项设置的更多信息，请参阅 修改选项设置。有关各项设置的更多信息，请参阅审计插件选项设置。

删除 MariaDB 审核插件

Amazon RDS 不支持关闭 MariaDB 审核插件中的日志记录。不过，您可以从数据库实例删除插件。在删除 MariaDB 审核插件时，将自动重新启动数据库实例以停止审核。

要从数据库实例中删除 MariaDB 审核插件，请执行以下操作之一：

- 从 MariaDB 审核插件选项所属的选项组中删除 MariaDB 审核插件选项。此更改会影响使用该选项组的所有数据库实例。有关更多信息，请参阅“从选项组中删除选项”
- 修改数据库实例，并指定不包含插件的其他选项组。此更改会影响单个数据库实例。您可以指定默认 (空) 选项组，或指定其他自定义选项组。有关更多信息，请参阅“修改 Amazon RDS 数据库实例”。

MariaDB 的参数

默认情况下，MariaDB 数据库实例使用特定于 MariaDB 数据库的数据库参数组。该参数组包含 MySQL 数据库引擎的 Amazon RDS 数据库参数组中所含的部分而非全部参数。它还包含一些新的特定于 MariaDB 的参数。有关使用参数组和设置参数的信息，请参阅Amazon RDS 的参数组。

查看 MariaDB 参数

RDS for MariaDB 参数设置为您已选择的存储引擎的默认值。有关 MariaDB 参数的更多信息，请参阅 MariaDB 文档。有关 MariaDB 存储引擎的更多信息，请参阅 Amazon RDS 上的 MariaDB 支持的存储引擎。

您可以使用 RDS 控制台或 Amazon CLI 查看特定 RDS for MariaDB 版本的可用参数。有关在 RDS 控制台中查看 MariaDB 参数组中参数的信息，请参阅 在 Amazon RDS 中查看数据库参数组的参数值。

使用 Amazon CLI，您可以通过运行 `describe-engine-default-parameters` 命令查看 RDS for MariaDB 版本的参数。为 `--db-parameter-group-family` 选项指定下列值之一：

- `mariadb10.11`
- `mariadb10.6`
- `mariadb10.5`
- `mariadb10.4`
- `mariadb10.3`

例如，要查看 RDS for MariaDB 版本 10.6 的参数，请运行以下命令。

```
aws rds describe-engine-default-parameters --db-parameter-group-family mariadb10.6
```

输出看上去类似于以下内容。

```
{
    "EngineDefaults": {
        "Parameters": [
            {
                "ParameterName": "alter_algorithm",
                "Description": "Specify the alter table algorithm.",
                "Source": "engine-default",
                "ApplyType": "dynamic",
                "DataType": "string",
```

```
                "AllowedValues": "DEFAULT,COPY,INPLACE,NOCOPY,INSTANT",
                "IsModifiable": true
            },
            {
                "ParameterName": "analyze_sample_percentage",
                "Description": "Percentage of rows from the table ANALYZE TABLE will
 sample to collect table statistics.",
                "Source": "engine-default",
                "ApplyType": "dynamic",
                "DataType": "float",
                "AllowedValues": "0-100",
                "IsModifiable": true
            },
            {
                "ParameterName": "aria_block_size",
                "Description": "Block size to be used for Aria index pages.",
                "Source": "engine-default",
                "ApplyType": "static",
                "DataType": "integer",
                "AllowedValues": "1024-32768",
                "IsModifiable": false
            },
            {
                "ParameterName": "aria_checkpoint_interval",
                "Description": "Interval in seconds between automatic checkpoints.",
                "Source": "engine-default",
                "ApplyType": "dynamic",
                "DataType": "integer",
                "AllowedValues": "0-4294967295",
                "IsModifiable": true
            },
        ...
```

要仅列出 RDS for MariaDB 版本 10.6 的可修改参数，请运行以下命令。

对于 Linux、macOS 或 Unix：

```
aws rds describe-engine-default-parameters --db-parameter-group-family mariadb10.6 \
   --query 'EngineDefaults.Parameters[?IsModifiable==`true`]'
```

对于 Windows：

```
aws rds describe-engine-default-parameters --db-parameter-group-family mariadb10.6 ^
```

```
    --query "EngineDefaults.Parameters[?IsModifiable==`true`]"
```

不提供的 MariaDB 参数

以下 MySQL 参数在 MariaDB 特定的数据库参数组中不可用：

- bind_address
- binlog_error_action
- binlog_gtid_simple_recovery
- binlog_max_flush_queue_time
- binlog_order_commits
- binlog_row_image
- binlog_rows_query_log_events
- binlogging_impossible_mode
- block_encryption_mode
- core_file
- default_tmp_storage_engine
- div_precision_increment
- end_markers_in_json
- enforce_gtid_consistency
- eq_range_index_dive_limit
- explicit_defaults_for_timestamp
- gtid_executed
- gtid-mode
- gtid_next
- gtid_owned
- gtid_purged
- log_bin_basename
- log_bin_index
- log_bin_use_v1_row_events
- log_slow_admin_statements
- log_slow_slave_statements

- log_throttle_queries_not_using_indexes
- master-info-repository
- optimizer_trace
- optimizer_trace_features
- optimizer_trace_limit
- optimizer_trace_max_mem_size
- optimizer_trace_offset
- relay_log_info_repository
- rpl_stop_slave_timeout
- slave_parallel_workers
- slave_pending_jobs_size_max
- slave_rows_search_algorithms
- storage_engine
- table_open_cache_instances
- timed_mutexes
- transaction_allow_batching
- validate-password
- validate_password_dictionary_file
- validate_password_length
- validate_password_mixed_case_count
- validate_password_number_count
- validate_password_policy
- validate_password_special_char_count

有关 MySQL 参数的更多信息，请参阅 MySQL 文档。

将数据从 MySQL 数据库快照迁移到 MariaDB 数据库实例

您可使用 Amazon Web Services Management Console、Amazon CLI 或 Amazon RDS API 将 RDS for MySQL 数据库快照迁移到运行 MariaDB 的新数据库实例。您必须使用从运行 MySQL 5.6 或 5.7 的 Amazon RDS 数据库实例创建的数据库快照。要了解如何创建 RDS for MySQL 数据库快照，请参阅为 Amazon RDS 的单可用区数据库实例创建数据库快照。

迁移快照不会影响从中拍摄快照的原始数据库实例。您可以先测试和验证新数据库实例，然后再将流量转移到该数据库实例以替代原始数据库实例。

在从 MySQL 迁移到 MariaDB 后，MariaDB 数据库实例将与默认数据库参数组和选项组关联。在还原数据库快照后，可以关联新数据库实例的自定义数据库参数组。不过，MariaDB 参数组具有一组不同的可配置系统变量。有关 MySQL 和 MariaDB 系统变量之间差异的信息，请参阅 MariaDB 和 MySQL 之间的系统变量差异。要了解数据库参数组，请参阅 Amazon RDS 的参数组。要了解选项组，请参阅使用选项组。

执行迁移

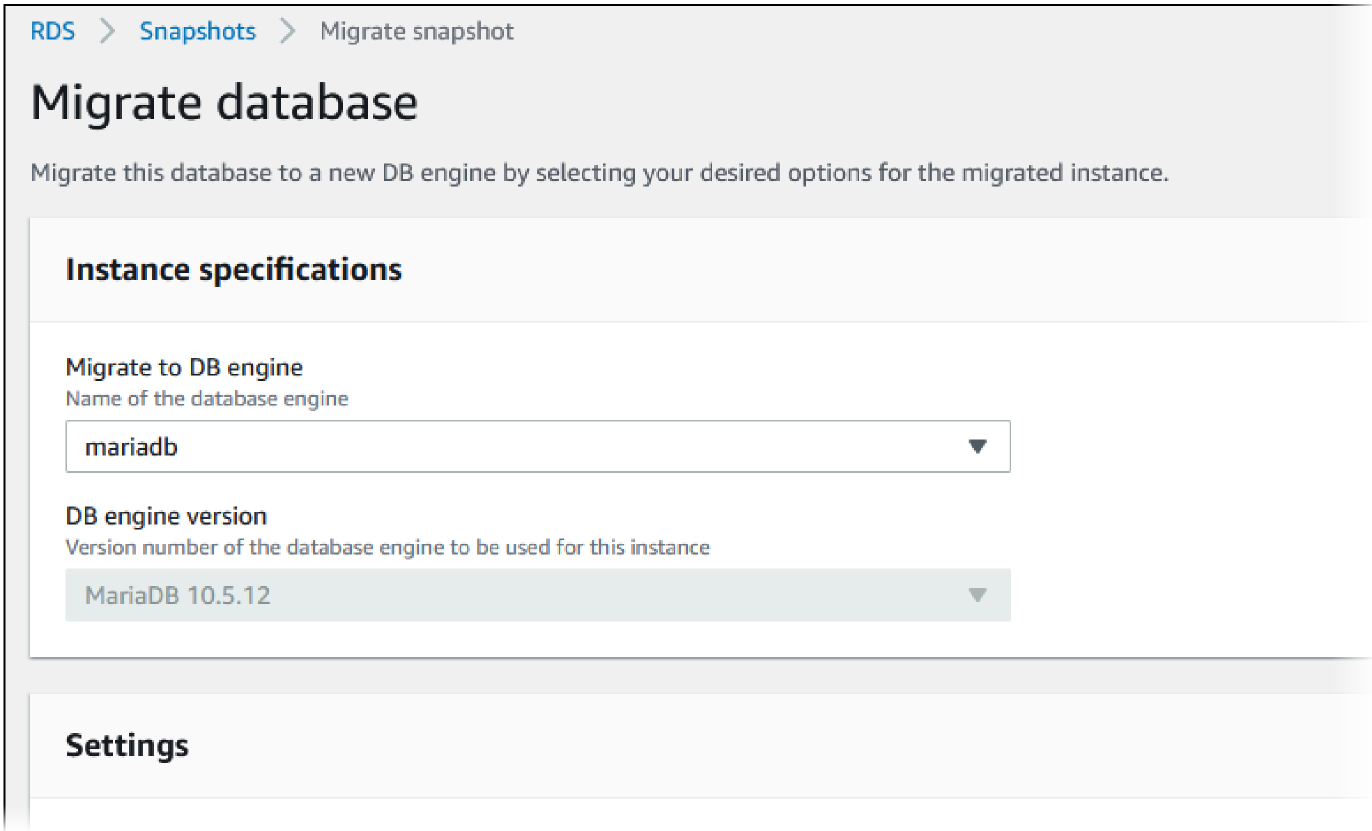
您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 将 RDS for MySQL 数据库快照迁移到新的 MariaDB 数据库实例。

控制台

将 MySQL 数据库快照迁移到 MariaDB 数据库实例

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择快照，然后选择您希望迁移的 MySQL 数据库快照。
3. 对于 Actions（操作），请选择 Migrate snapshot（迁移快照）。随后将显示 Migrate database（迁移数据库）页。
4. 对于迁移到数据库引擎，选择 mariadb。

   Amazon RDS 将自动选择 DB engine version（数据库引擎版本）。您不能更改数据库引擎版本。



5. 对于其余部分，请指定数据库实例设置。有关每项设置的信息，请参阅 数据库实例的设置。
6. 选择 Migrate。

Amazon CLI

要将数据从 MySQL 数据库快照迁移到 MariaDB 数据库实例，请结合以下选项运行 Amazon CLI `restore-db-instance-from-db-snapshot` 命令：

- --db-instance-identifier – 要从数据库快照创建的数据库实例的名称。
- --db-snapshot-identifier – 要从中还原的数据库快照的标识符。
- --engine – 要用于新实例的数据库引擎。

Example

对于 Linux、macOS 或 Unix：

```
aws rds restore-db-instance-from-db-snapshot \
    --db-instance-identifier newmariadbinstance \
    --db-snapshot-identifier mysqlsnapshot \
```

```
    --engine mariadb
```

对于 Windows：

```
aws rds restore-db-instance-from-db-snapshot ^
    --db-instance-identifier newmariadbinstance ^
    --db-snapshot-identifier mysqlsnapshot ^
    --engine mariadb
```

API

要将数据从 MySQL 数据库快照迁移到 MariaDB 数据库实例，请调用 Amazon RDS API 操作 `RestoreDBInstanceFromDBSnapshot`。

MariaDB 和 MySQL 之间的不兼容性

MySQL 和 MariaDB 之间的不兼容性包括：

- 您无法将使用 MySQL 8.0 创建的数据库快照迁移到 MariaDB。
- 如果源 MySQL 数据库使用 SHA256 密码哈希，请确保先重置经过 SHA256 哈希处理的用户密码，然后再连接到 MariaDB 数据库。以下代码说明如何重置经过 SHA256 哈希处理的密码。

```
SET old_passwords = 0;
UPDATE mysql.user SET plugin = 'mysql_native_password',
Password = PASSWORD('new_password')
WHERE (User, Host) = ('master_user_name', %);
FLUSH PRIVILEGES;
```

- 如果您的 RDS 主用户账户使用 SHA-256 密码哈希，请确保使用 Amazon Web Services Management Console、`modify-db-instance` Amazon CLI 命令或 ModifyDBInstance RDS API 操作重置密码。有关修改数据库实例的信息，请参阅修改 Amazon RDS 数据库实例。
- MariaDB 不支持 Memcached 插件。不过，Memcached 插件使用的数据将存储为 InnoDB 表。在迁移 MySQL 数据库快照后，您可以使用 SQL 访问由 Memcached 插件使用的数据。有关 innodb_memcache 数据库的更多信息，请参阅 InnoDB 内存缓存插件内部信息。

Amazon RDS SQL 上的 MariaDB 参考

在下文中，您可以找到对于运行 MariaDB 数据库引擎的 Amazon RDS 实例可用的系统存储过程的说明。

对于 MySQL 数据库实例和 MariaDB 数据库实例，您可以使用可用的系统存储过程。这些存储过程记录在RDS for MySQL 存储过程参考上。MariaDB 数据库实例支持除 `mysql.rds_start_replication_until` 和 `mysql.rds_start_replication_until_gtid` 以外的所有存储过程。

此外，只有运行 MariaDB 的 Amazon RDS 数据库实例支持以下系统存储过程：

- mysql.rds_replica_status
- mysql.rds_set_external_master_gtid
- mysql.rds_kill_query_id

mysql.rds_replica_status

显示 MariaDB 只读副本的复制状态。

在只读副本上调用此过程可显示有关副本线程基本参数的状态信息。

语法

```
CALL mysql.rds_replica_status;
```

使用说明

只有运行 MariaDB 版本 10.5 及更高版本的 MariaDB 数据库实例才支持此过程。

此过程等同于 `SHOW REPLICA STATUS` 命令。MariaDB 10.5 及更高版本的数据库实例不支持此命令。

在以前版本的 MariaDB 中，同等的 `SHOW SLAVE STATUS` 命令需要 `REPLICATION SLAVE` 特权。在 MariaDB 版本 10.5 及更高版本中，其需要 `REPLICATION REPLICA ADMIN` 特权。为了保护 MariaDB 10.5 及更高版本数据库实例的 RDS 管理，未向 RDS 主用户授予此新权限。

示例

以下示例显示了 MariaDB 只读副本的状态：

```
call mysql.rds_replica_status;
```

响应类似于以下内容：

```
*************************** 1. row ***************************
                Replica_IO_State: Waiting for master to send event
                     Source_Host: XX.XX.XX.XXX
                     Source_User: rdsrepladmin
                     Source_Port: 3306
                   Connect_Retry: 60
                 Source_Log_File: mysql-bin-changelog.003988
             Read_Source_Log_Pos: 405
                  Relay_Log_File: relaylog.011024
                   Relay_Log_Pos: 657
           Relay_Source_Log_File: mysql-bin-changelog.003988
              Replica_IO_Running: Yes
             Replica_SQL_Running: Yes
                 Replicate_Do_DB:
             Replicate_Ignore_DB:
              Replicate_Do_Table:
          Replicate_Ignore_Table:
 mysql.rds_sysinfo,mysql.rds_history,mysql.rds_replication_status
         Replicate_Wild_Do_Table:
     Replicate_Wild_Ignore_Table:
                      Last_Errno: 0
                      Last_Error:
                    Skip_Counter: 0
             Exec_Source_Log_Pos: 405
                 Relay_Log_Space: 1016
                 Until_Condition: None
                  Until_Log_File:
                   Until_Log_Pos: 0
              Source_SSL_Allowed: No
              Source_SSL_CA_File:
              Source_SSL_CA_Path:
                 Source_SSL_Cert:
               Source_SSL_Cipher:
                  Source_SSL_Key:
           Seconds_Behind_Master: 0
   Source_SSL_Verify_Server_Cert: No
                   Last_IO_Errno: 0
                   Last_IO_Error:
                  Last_SQL_Errno: 0
```

```
                  Last_SQL_Error:
      Replicate_Ignore_Server_Ids:
                Source_Server_Id: 807509301
                  Source_SSL_Crl:
              Source_SSL_Crlpath:
                      Using_Gtid: Slave_Pos
                     Gtid_IO_Pos: 0-807509301-3980
         Replicate_Do_Domain_Ids:
     Replicate_Ignore_Domain_Ids:
                   Parallel_Mode: optimistic
                       SQL_Delay: 0
             SQL_Remaining_Delay: NULL
       Replica_SQL_Running_State: Reading event from the relay log
              Replica_DDL_Groups: 15
Replica_Non_Transactional_Groups: 0
    Replica_Transactional_Groups: 3658
1 row in set (0.000 sec)

Query OK, 0 rows affected (0.000 sec)
```

mysql.rds_set_external_master_gtid

配置从在 Amazon RDS 外部运行的 MariaDB 实例到 MariaDB 数据库实例的基于 GTID 的复制。仅当外部 MariaDB 实例的版本为 10.0.24 或更高版本时，才支持此存储过程。在设置复制时，如果其中的一个或两个实例不支持 MariaDB 全局事务标识符 (GTID)，请使用 mysql.rds_set_external_master（RDS for MariaDB 和 RDS for MySQL 主要版本 8.0 及更低版本）。

使用 GTID 进行复制提供了二进制日志复制无法提供的崩溃安全功能，因此，我们建议在复制实例支持 GTID 的情况下使用它。

语法

```
CALL mysql.rds_set_external_master_gtid(
  host_name
  , host_port
  , replication_user_name
  , replication_user_password
  , gtid
  , ssl_encryption
);
```

参数

host_name

字符串。在 Amazon RDS 外部运行并将变为源实例的 MariaDB 实例的主机名或 IP 地址。

host_port

整数。在 Amazon RDS 外部运行的要配置为源实例的 MariaDB 实例使用的端口。如果网络配置包括转换端口号的 SSH 端口复制，则指定由 SSH 公开的端口号。

replication_user_name

字符串。在要配置为只读副本的 MariaDB 数据库实例中具有 `REPLICATION SLAVE` 权限的用户的 ID。

replication_user_password

字符串。在 `replication_user_name` 中指定的用户 ID 的密码。

gtid

字符串。应作为复制的起始点的源实例上的全局事务 ID。

如果源实例在您配置复制时已被锁定，则可以使用 `@@gtid_current_pos` 获取当前 GTID，以便让二进制日志在您获得 GTID 和复制开始之间的这一段时间内不会改变。

否则，如果您使用 `mysqldump` 版本 10.0.13 或更高版本在开始复制前填充副本实例，则可以使用 `--master-data` 或 `--dump-slave` 选项获取输出中的 GTID 位置。如果您不使用 `mysqldump` 10.0.13 版或更高版本，则可以运行 `SHOW MASTER STATUS` 或使用这些相同的 `mysqldump` 选项来获取二进制日志文件名称和位置，然后通过对外部 MariaDB 实例运行 `BINLOG_GTID_POS` 来将这些名称和位置转换为 GTID：

```
SELECT BINLOG_GTID_POS('<binary log file name>', <binary log file position>);
```

有关 GTID 的 MariaDB 实施的更多信息，请参阅 MariaDB 文档中的[全局事务 ID]。

ssl_encryption

指定是否在复制连接中使用安全套接字层（SSL）加密的值。1 表示使用 SSL 加密，0 表示不使用加密。默认值为 0。

> **Note**
>
> 不支持 MASTER_SSL_VERIFY_SERVER_CERT 选项。此选项设置为 0，这意味着连接已加密，但未验证证书。

使用说明

mysql.rds_set_external_master_gtid 过程必须由主用户运行。它必须在 MariaDB 数据库实例（由您配置为在 Amazon RDS 外部运行的 MariaDB 实例的副本）上运行。在运行 mysql.rds_set_external_master_gtid 之前，必须将在 Amazon RDS 外部运行的 MariaDB 实例配置为源实例。有关更多信息，请参阅“将数据导入到 Amazon RDS for MariaDB 数据库实例”。

> **Warning**
>
> 请勿使用 mysql.rds_set_external_master_gtid 管理两个 Amazon RDS 数据库实例之间的复制。仅当使用在 RDS 外部运行的 MariaDB 实例进行复制时，才能使用它。有关管理 Amazon RDS 数据库实例之间的复制的信息，请参阅使用数据库实例只读副本。

在调用 mysql.rds_set_external_master_gtid 将 Amazon RDS 数据库实例配置为只读副本后，可对该副本调用 mysql.rds_start_replication 开始复制过程。您可以调用 mysql.rds_reset_external_master（RDS for MariaDB 和 RDS for MySQL 主要版本 8.0 及更低版本）删除只读副本配置。

调用 mysql.rds_set_external_master_gtid 时，Amazon RDS 将时间、用户和“设置主体”的操作记录在 mysql.rds_history 和 mysql.rds_replication_status 表中。

示例

在 MariaDB 数据库实例上运行时，以下示例将该数据库实例配置为在 Amazon RDS 外部运行的某个 MariaDB 实例的副本。

```
call mysql.rds_set_external_master_gtid
 ('Sourcedb.some.com',3306,'ReplicationUser','SomePassW0rd','0-123-456',0);
```

mysql.rds_kill_query_id

结束针对 MariaDB 服务器运行的查询，以终止长时间运行的查询或有问题的查询。您可以识别查询 ID 并有效地停止特定查询，以解决性能问题并保持理想的数据库操作。

语法

```
CALL mysql.rds_kill_query_id(queryID);
```

参数

queryID

整数。将结束的查询的标识。

使用说明

要停止针对 MariaDB 服务器运行的查询，请使用 `mysql.rds_kill_query_id` 过程并传入该查询的 ID。要获取查询 ID，请查询 MariaDB 信息架构 PROCESSLIST 架构表，如下所示：

```
SELECT USER, HOST, COMMAND, TIME, STATE, INFO, QUERY_ID FROM
                INFORMATION_SCHEMA.PROCESSLIST WHERE USER = '<user name>';
```

与 MariaDB 服务器的连接将会保留。

示例

以下示例结束查询 ID 为 230040 的查询：

```
call mysql.rds_kill_query_id(230040);
```

MariaDB 数据库实例的本地时区

默认情况下，MariaDB 数据库实例的时区是协调世界时 (UTC)。您可以将数据库实例的时区设置为您的应用程序的本地时区。

要设置数据库实例的本地时区，请将数据库实例参数组中的 `time_zone` 参数设置为本节后面列出的受支持值之一。在设置参数组的 `time_zone` 参数时，所有使用该参数组的数据库实例和只读副本都会改用新的本地时区。有关如何设置参数组中的参数的信息，请参阅Amazon RDS 的参数组。

设置本地时区之后，所有新数据库连接都会反映更改。如果在更改本地时区时打开了任何数据库连接，则到关闭连接再打开新连接之后才会看到本地时区更新。

您可以为数据库实例及其一个或多个只读副本设置不同的本地时区。为此，请对数据库实例和副本使用不同的参数组，并将每个参数组中的 `time_zone` 参数设置为不同的本地时区。

如果要跨 Amazon Web Services 区域复制，则源数据库实例和只读副本使用不同的参数组（参数组对于 Amazon Web Services 区域是唯一的）。要对每个实例使用相同的本地时区，您必须在实例和只读副本的参数组中设置 `time_zone` 参数。

从数据库快照还原数据库实例时，本地时区设置为 UTC。还原完成之后，可以将时区更新为本地时区。如果将数据库实例还原到某个时间点，则还原的数据库实例的本地时区是来自还原的数据库实例的参数组的时区设置。

互联网编号分配机构（IANA）每年多次在 https://www.iana.org/time-zones 上发布新的时区。每当 RDS 发布 MariaDB 的新次要维护版本时，RDS 都会附带发布时的最新时区数据。当您使用最新的 RDS for MariaDB 版本时，您会获得来自 RDS 的最新时区数据。为确保您的数据库实例具有最新的时区数据，我们建议升级到更高的数据库引擎版本。或者，您可以手动修改 MariaDB 数据库实例中的时区表。为此，可以使用 SQL 命令或者在 SQL 客户端中运行 mysql_tzinfo_to_sql 工具。手动更新时区数据后，重启数据库实例以使更改生效。RDS 不会修改或重置正在运行的数据库实例的时区数据。只有在执行数据库引擎版本升级时，才会安装新的时区数据。

您可以将本地时区设置为以下值之一。

区	时区
非洲	非洲/开罗、非洲/卡萨布兰卡、非洲/哈拉雷、非洲/蒙罗维亚、非洲/内罗毕、非洲/的黎波里、非洲/温得和克
美洲	美洲/阿拉瓜尼亚、美洲/亚松森、美洲/波哥大、美洲/布宜诺斯艾利斯、美洲/加拉加斯、美洲/奇瓦瓦、美洲/库亚巴、美洲/丹佛、美洲/福塔莱萨、美洲/危地马

区	时区
	拉、美洲/哈利法克斯、美洲/玛瑙斯、美洲/马塔莫罗斯、美洲/蒙特雷、美洲/蒙得维的亚、美洲/凤凰城、美洲/圣地亚哥、美洲/蒂华纳
亚洲	亚洲/阿曼、亚洲/阿什哈巴德、亚洲/巴格达、亚洲/巴库、亚洲/曼谷、亚洲/贝鲁特、亚洲/加尔各答、亚洲/大马士革、亚洲/达卡、亚洲/伊尔库茨克、亚洲/耶路撒冷、亚洲/喀布尔、亚洲/卡拉奇、亚洲/加德满都、亚洲/克拉斯诺雅茨克、亚洲/马加丹、亚洲/马斯喀特、亚洲/新西伯利亚、亚洲/利雅得、亚洲/首尔、亚洲/上海、亚洲/新加坡、亚洲/台北、亚洲/德黑兰、亚洲/东京、亚洲/乌兰巴托、亚洲/符拉迪沃斯托克、亚洲/雅库茨克、亚洲/埃里温
大西洋	大西洋州/亚速尔群岛
澳大利亚	澳大利亚/阿德莱德、澳大利亚/布里斯班、澳大利亚/达尔文、澳大利亚/霍巴特、澳大利亚/珀斯、澳大利亚/悉尼
巴西	巴西/迪诺罗尼亚、巴西/东部
加拿大	加拿大/纽芬兰、加拿大/萨斯喀彻温省、加拿大/育空地区
欧洲	欧洲/阿姆斯特丹、欧洲/雅典、欧洲/都柏林、欧洲/赫尔辛基、欧洲/伊斯坦布尔、欧洲/加里宁格勒、欧洲/莫斯科、欧洲/巴黎、欧洲/布拉格、欧洲/萨拉热窝
太平洋	太平洋/奥克兰、太平洋/斐济、太平洋/关岛、太平洋/檀香山、太平洋/萨摩亚群岛
美国	美国/阿拉斯加、美国/中部、美国/印第安纳东部、美国/东部、美国/太平洋地区
UTC	UTC

RDS for MariaDB 的已知问题和限制

以下各项是使用 RDS for MariaDB 时的已知问题和限制。

> **Note**
>
> 此列表并不详尽。

主题

Amazon RDS 中的 MariaDB 文件大小限制

对于 MariaDB 数据库实例，在使用 InnoDB file-per-table 表空间时，表的最大大小为 16 TB。此限制也将系统表空间的最大大小约束为 16 TB。默认情况下，为 MariaDB 数据库实例设置 InnoDB file-per-table 表空间（每个表位于各自的表空间中）。此限制与 MariaDB 数据库实例的最大存储限制无关。有关存储限制的更多信息，请参阅 Amazon RDS 数据库实例存储。

根据您的应用程序，使用 InnoDB file-per-table 表空间有一些优缺点。要确定适用于您的应用程序的最佳方法，请参阅 MySQL 文档中的 File-Per-Table 表空间。

建议不要允许表增长到最大文件大小。一般而言，更好的方法是将数据分区到较小的表中，这可以提高性能并缩短恢复时间。

可用于将大型表划分为多个较小表的一个选项是分区。分区 可基于您指定的规则将大型表的各个部分分布到单独的文件中。例如，如果您按日期存储事务，则可以创建分区规则，以便使用分区将较旧的事务分布到单独的文件中。随后您可以定期存档无需随时可供应用程序使用的历史事务数据。有关更多信息，请参阅 MySQL 文档中的分区。

确定所有 InnoDB 表空间的大小

- 使用以下 SQL 命令可确定任何表是否太大以及是否适合进行分区。

> **Note**
>
> 对于 MariaDB 10.6 及更高版本，此查询还返回 InnoDB 系统表空间的大小。
> 对于 10.6 之前的 MariaDB 版本，您无法通过查询系统表来确定 InnoDB 系统表空间的大小。我们强烈建议您升级到更高的版本。

```
SELECT SPACE,NAME,ROUND((ALLOCATED_SIZE/1024/1024/1024), 2)
as "Tablespace Size (GB)"
FROM information_schema.INNODB_SYS_TABLESPACES ORDER BY 3 DESC;
```

确定非 InnoDB 用户表的大小

- 使用以下 SQL 命令来确定是否有任何非 InnoDB 用户表过大。

```
SELECT TABLE_SCHEMA, TABLE_NAME, round(((DATA_LENGTH + INDEX_LENGTH+DATA_FREE)
/ 1024 / 1024/ 1024), 2) As "Approximate size (GB)" FROM information_schema.TABLES
WHERE TABLE_SCHEMA NOT IN ('mysql', 'information_schema', 'performance_schema')
and ENGINE<>'InnoDB';
```

启用 InnoDB file-per-table 表空间

- 在数据库实例的参数组中将 `innodb_file_per_table` 参数设置为 `1`。

禁用 InnoDB file-per-table 表空间

- 在数据库实例的参数组中将 `innodb_file_per_table` 参数设置为 `0`。

有关更新参数组的信息，请参阅Amazon RDS 的参数组。

当您启用或禁用 InnoDB file-per-table 表空间后，可以发出 `ALTER TABLE` 命令。您可以使用此命令将表从全局表空间移动到其自己的表空间。或者，您可以将表从其自己的表空间移动到全局表空间。以下为示例。

```
ALTER TABLE table_name ENGINE=InnoDB, ALGORITHM=COPY;
```

InnoDB 保留字

`InnoDB` 是 RDS for MariaDB 的保留字。您不能将此名称用于 MariaDB 数据库。

自定义端口

Amazon RDS 禁止连接到 MariaDB 引擎的自定义端口 33060。为您的 MariaDB 引擎选择不同的端口。

性能详情

InnoDB 计数器在 RDS for MariaDB 版本 10.11 的性能详情中不可见，因为 MariaDB 社区不再支持它们。

Amazon RDS for Microsoft SQL Server

Amazon RDS 支持多个版本的 Microsoft SQL Server。下表显示了每个主要版本支持的最新次要版本。有关所支持版本和 RDS 引擎版本的完整列表，请参阅 Amazon RDS 上的 Microsoft SQL Server 版本。

主要版本	Service Pack/ GDR	累积更新	次要版本	知识库文章	发行日期
SQL Server 2022	不适用	CU18	16.0.4185.3	KB5050771	2025 年 3 月 13 日
SQL Server 2019	不适用	CU32	15.0.4430.1	KB5054833	2025 年 2 月 27 日
SQL Server 2017	不适用	CU31 GDR	14.0.3485.1	KB5046858	2024 年 11 月 12 日
SQL Server 2016	SP3 GDR	CU14	13.0.6455.2	KB5046855	2024 年 11 月 12 日

有关适用于 SQL Server 的许可的信息，请参阅授予 Amazon RDS 上 Microsoft SQL Server 的许可。有关 SQL Server 内部版本的信息，请参阅有关在何处查找有关最新 SQL Server 内部版本的信息的这篇 Microsoft 支持文章。

利用 Amazon RDS，您可以创建数据库实例和数据库快照、时间点还原以及自动或手动备份。可在 VPC 内使用运行 SQL Server 的数据库实例。此外，您还可使用安全套接字层（SSL）连接到运行 SQL Server 的数据库实例，并且可以使用透明数据加密（TDE）加密静态数据。Amazon RDS 当前支持对将 SQL Server 数据库镜像（DBM）或 Always On 可用性组（AG）用作高可用性失效转移解决方案的 SQL Server 执行多可用区部署。

为让用户获得托管式服务体验，Amazon RDS 未提供对数据库实例的 Shell 访问权限，并且限制对需要高级权限的某些系统程序和表的访问权限。Amazon RDS 支持使用任何标准 SQL 客户端应用程序（例如 Microsoft SQL Server Management Studio）访问数据库实例上的数据库。Amazon RDS 不允许通过 Telnet、安全外壳（SSH）或 Windows 远程桌面连接直接从主机访问数据库实例。在创建数据库实例时，主用户将分配给该实例上的所有用户数据库的 db_owner 角色，并具有用于备份的权限以外的所有数据库级权限。Amazon RDS 会为您管理备份。

在创建您的第一个数据库实例之前，您应完成本指南的设置部分中的步骤。有关更多信息，请参阅“设置 Amazon RDS 环境”。

主题

Amazon RDS 上 Microsoft SQL Server 的常见管理任务

以下是您使用 Amazon RDS for SQL Server 数据库实例执行的常见管理任务，以及有关每个任务的相关文档的链接。

任务区域	描述	相关文档
实例类、存储和 PIOPS	如果您出于生产目的而创建数据库实例，则应该了解 Amazon RDS 中的实例类、存储类型和配置的 IOPS 的工作原理。	Microsoft SQL Server 的数据库实例类支持 Amazon RDS 存储类型
多可用区部署	生产数据库实例应使用多可用区部署。多可用区部署为数据库实例提供了更高的可用性、数据持久性和容错能力。SQL Server 的多可用区部署使用 SQL Server 的本机 DBM 或 AG 技术实现。	配置和管理 Amazon RDS 的多可用区部署 使用 Microsoft SQL Server 数据库镜像或 Always On 可用性组的多可用区部署
Amazon Virtual Private Cloud (VPC)	如果您的 Amazon 账户具有默认 VPC，则自动在默认 VPC 中创建数据库实例。如果您的账户没有默认 VPC，并且您希望数据库实例位于 VPC 中，则必须在创建数据库实例之前创建 VPC 和子网组。	在 VPC 中使用数据库实例
安全组	默认情况下，数据库实例在创建时带有一个防火墙，这个防火墙会阻止对实例进行访问。因此，您必须使用正确的 IP 地址和网络配置创建安全组来访问数据库实例。	使用安全组控制访问权限
参数组	如果数据库实例需要特定数据库参数，则应在创建数据库实例前创建参数组。	Amazon RDS 的参数组

任务区域	描述	相关文档
选项组	如果数据库实例需要特定数据库选项，则应在创建数据库实例前创建选项组。	适用于 Microsoft SQL Server 数据库引擎的选项
连接到您的数据库实例	在创建安全组并将其与数据库实例关联后，您可以使用任何标准 SQL 客户端应用程序 (如 Microsoft SQL Server Management Studio) 连接到数据库实例。	连接到 Microsoft SQL Server 数据库实例
备份和还原	在创建数据库实例时，可以将它配置为使用自动备份。您还可以使用完整备份文件 (.bak 文件) 来手动备份和还原数据库。	备份简介 使用本机备份和还原导入和导出 SQL Server 数据库
监控	可使用 CloudWatch Amazon RDS 指标、事件和增强监测来监控 SQL Server 数据库实例。	在 Amazon RDS 控制台中查看指标 查看 Amazon RDS 事件
日志文件	您可以访问 SQL Server 数据库实例的日志文件。	监控 Amazon RDS 日志文件 Amazon RDS for Microsoft SQL Server 数据库日志文件

此外，还有使用 SQL Server 数据库实例的高级管理任务。有关更多信息，请参阅以下文档：

- Amazon RDS for Microsoft SQL Server 的常见 DBA 任务.
- 将 Amazon Managed Active Directory 用于 RDS for SQL Server
- 访问临时数据库

Microsoft SQL Server 数据库实例的限制

在数据库实例上进行 Microsoft SQL Server 的 Amazon RDS 实施时，应注意某些限制：

- 数据库实例上支持的最大数据库数取决于实例类的类型和可用性模式 — 单可用区、数据库镜像 (DBM) 多可用区或可用性组 (AG) 多可用区。Microsoft SQL Server 系统数据库不计入此限制中。

  下表显示了每种实例类类型和可用性模式支持的最大数据库数。使用此表帮助您确定是否可从一种实例类类型切换到另一种类型，或是否可从一种可用性模式切换到另一种模式。如果您的源数据库实例拥有的数据库数量超出目标实例类类型或可用性模式可以支持的数量，则修改数据库实例会失败。您可以在 Events (事件) 窗格中查看请求的状态。

实例类类型	单可用区	DBM 多可用区	Always On 可用性组多可用区
db.*.micro to db.*.medium	30	不适用	不适用
db.*.large	30	30	30
db.*.xlarge 至 db.*.16xlarge	100	50	75
db.*.24xlarge 至 db.*.32xlarge	100	50	100

  * 表示不同实例类的类型。

  例如，我们假定您的数据库实例在使用单可用区的 db.*.16xlarge 上运行，且拥有 76 个数据库。您修改数据库实例，以便升级为使用 Always On 可用性组多可用区。因为您的数据库实例包含的数据库数量超出您的目标配置支持的数据库数量，所以此次升级失败。如果您将实例类类型升级为 db.*.24xlarge，则修改会成功。

  如果升级失败，则您会看到类似于下面所示的事件和消息：

  - 无法修改数据库实例类。该实例有 76 个数据库，但在转换之后，它仅支持 75 个数据库。
  - 无法将数据库实例转换为多可用区：该实例有 76 个数据库，但在转换之后，它仅支持 75 个数据库。

如果时间点还原或快照还原失败，则您会看到类似于下面所示的事件和消息：

  - 数据库实例进入不兼容的还原。该实例有 76 个数据库，但在转换之后，它仅支持 75 个数据库。

- 以下端口预留供 Amazon RDS 使用，您不能在创建数据库实例时使用它们：`1234, 1434, 3260, 3343, 3389, 47001,` 和 `49152-49156`。
- 不允许来自 169.254.0.0/16 范围内的 IP 地址的客户端连接。这是自动私有 IP 寻址范围 (APIPA)，它用于本地链路寻址。
- 如果数据库实例的处理器数量超过软件限制（24 个内核、4 个插槽和 128GB RAM），则 SQL Server Standard 版将仅使用部分可用处理器。例如 db.m5.24xlarge 和 db.r5.24xlarge 实例类。

  有关详细信息，请参阅 Microsoft 文档中 [SQL Server 2019 (15.x) 的版本和支持的功能]下的扩展限制表。
- Amazon RDS for SQL Server 不支持将数据导入 msdb 数据库。
- 您无法重命名 SQL Server 多可用区部署中的数据库实例中的数据库。
- 在 RDS for SQL Server 上设置以下数据库参数时，请确保遵循以下准则：
  - `max server memory (mb)` >= 256 MB
  - `max worker threads` >=（逻辑 CPU 的数量 * 7）

  有关设置数据库参数的更多信息，请参阅 [Amazon RDS 的参数组]。
- SQL Server 数据库实例的最大存储大小如下所示：
  - 通用型 (SSD) 存储 – 16 TiB（针对所有版本）
  - 预调配 IOPS 存储 – 64 TiB（针对所有版本）
  - 磁性存储 – 1 TiB（针对所有版本）

  对于需要大量存储的情况，您可跨多个数据库实例使用分片绕过该限制。此方法需要连接到已分片系统的应用程序中的依赖数据的路由逻辑。可以使用现有分片框架，也可以编写自定义代码来启用分片。如果使用现有框架，则该框架无法在与数据库实例相同的服务器上安装任何组件。
- SQL Server 数据库实例的最小存储大小如下所示：
  - 通用 (SSD) 存储 – 20 GiB，用于 Enterprise、Standard、Web 和 Express 版本
  - 预置 IOPS 存储 – 20 GiB，用于 Enterprise、Standard、Web 和 Express 版本
  - 磁性存储 – 20 GiB，用于 Enterprise、Standard、Web 和 Express 版本
- Amazon RDS 不支持在与 RDS 数据库实例相同的服务器上运行这些服务：
  - 数据质量服务
  - 主数据服务

要使用这些功能，建议您在 Amazon EC2 实例上安装 SQL Server，或者使用本地 SQL Server 实例。在这些情况下，EC2 或 SQL Server 实例充当 Amazon RDS 上的 SQL Server 数据库实例的主数据服务服务器。根据 Microsoft 许可策略，您可以在带有 Amazon EBS 存储的 Amazon EC2 实例上安装 SQL Server。

- 由于 Microsoft SQL Server 中存在的限制，还原到成功运行 `DROP DATABASE` 前的时间点可能不会反映出数据库在该时间点的状态。例如，通常会将已删掉的数据库还原到距发出 `DROP DATABASE` 命令前还有长达 5 分钟时的状态。使用此类型的还原，意味着无法还原您的已删除数据库上在那几分钟进行的事务。要解决该问题，您可以在完成还原操作后重新发出 `DROP DATABASE` 的命令。删除数据库也会删除该数据库的事务日志。
- 对于 SQL Server，在创建数据库实例之后，需创建数据库。数据库名称遵循常用的 SQL Server 命名规则，但有以下区别：
  - 数据库名称不能以 `rdsadmin` 开头。
  - 它们不能以空格或制表符开头或结尾。
  - 它们不能包含任何创建新行的字符。
  - 它们不能包含单引号 (`'`)。
- SQL Server Web Edition 仅允许您在为 SQL Server 数据库实例创建新的 RDS 时使用开发/测试模板。

Microsoft SQL Server 的数据库实例类支持

数据库实例的计算和内存容量由数据库实例类决定。您需要的数据库实例类取决于您的处理能力和内存要求。有关更多信息，请参阅“数据库实例类”。

以下是 Microsoft SQL Server 支持的数据库实例类列表，此处为方便起见而提供。如需最新的列表，请参阅 RDS 控制台：https://console.aws.amazon.com/rds/。

并非所有受支持的 SQL Server 次要版本中均提供所有数据库实例类。例如，较旧的次要版本中不提供一些较新的数据库实例类（例如 db.r6i）。您可以使用 describe-orderable-db-instance-options Amazon CLI 命令来找出哪些数据库实例类可用于您的 SQL Server 版本。

SQL Server 版本	2022 支持范围	2019 支持范围	2017 和 2016 支持范围
企业版	db.t3.xlarge –db.t3.2xlarge db.r5.xlarge –db.r5.24xlarge db.r5b.xlarge –db.r5b.24xlarge db.r5d.xlarge –db.r5d.24xlarge db.r6i.xlarge –db.r6i.32xlarge db.m5.xlarge –db.m5.24xlarge db.m5d.xlarge –db.m5d.24xlarge db.m6i.xlarge –db.m6i.32xlarge db.x2iedn.xlarge –db.x2iedn.32xlarge	db.t3.xlarge –db.t3.2xlarge db.r5.xlarge –db.r5.24xlarge db.r5b.xlarge –db.r5b.24xlarge db.r5d.xlarge –db.r5d.24xlarge db.r6i.xlarge –db.r6i.32xlarge db.m5.xlarge –db.m5.24xlarge db.m5d.xlarge –db.m5d.24xlarge db.m6i.xlarge –db.m6i.32xlarge db.x1.16xlarge –db.x1.32xlarge	db.t3.xlarge –db.t3.2xlarge db.r5.xlarge –db.r5.24xlarge db.r5b.xlarge –db.r5b.24xlarge db.r5d.xlarge –db.r5d.24xlarge db.r6i.xlarge –db.r6i.32xlarge db.m5.xlarge –db.m5.24xlarge db.m5d.xlarge –db.m5d.24xlarge db.m6i.xlarge –db.m6i.32xlarge db.x1.16xlarge –db.x1.32xlarge

SQL Server 版本	2022 支持范围	2019 支持范围	2017 和 2016 支持范围
	db.z1d.xlarge –db.z1d.12xlarge	db.x1e.xlarge –db.x1e.32xlarge db.x2iedn.xlarge –db.x2iedn.32xlarge db.z1d.xlarge –db.z1d.12xlarge	db.x1e.xlarge –db.x1e.32xlarge db.x2iedn.xlarge –db.x2iedn.32xlarge db.z1d.xlarge –db.z1d.12xlarge

SQL Server 版本	2022 支持范围	2019 支持范围	2017 和 2016 支持范围
标准版	db.t3.xlarge –db.t3.2xlarge db.r5.large –db.r5.24xlarge db.r5b.large –db.r5b.8xlarge db.r5d.large –db.r5d.24xlarge db.r6i.large –db.r6i.8xlarge db.m5.large –db.m5.24xlarge db.m5d.large –db.m5d.24xlarge db.m6i.large –db.m6i.8xlarge db.x2iedn.xlarge –db.x2iedn.8xlarge db.z1d.large –db.z1d.12xlarge	db.t3.xlarge –db.t3.2xlarge db.r5.large –db.r5.24xlarge db.r5b.large –db.r5b.24xlarge db.r5d.large –db.r5d.24xlarge db.r6i.large –db.r6i.8xlarge db.m5.large –db.m5.24xlarge db.m5d.large –db.m5d.24xlarge db.m6i.large –db.m6i.8xlarge db.x1.16xlarge –db.x1.32xlarge db.x1e.xlarge –db.x1e.32xlarge	db.t3.xlarge –db.t3.2xlarge db.r5.large –db.r5.24xlarge db.r5b.large –db.r5b.24xlarge db.r5d.large –db.r5d.24xlarge db.r6i.large –db.r6i.8xlarge db.m5.large –db.m5.24xlarge db.m5d.large –db.m5d.24xlarge db.m6i.large –db.m6i.8xlarge db.x1.16xlarge –db.x1.32xlarge db.x1e.xlarge –db.x1e.32xlarge

SQL Server 版本	2022 支持范围	2019 支持范围	2017 和 2016 支持范围
		db.x2iedn.xlarge –db.x2iedn.32xlarge db.z1d.large –db.z1d.12xlarge	db.x2iedn.xlarge –db.x2iedn.32xlarge db.z1d.large –db.z1d.12xlarge
Web 版	db.t3.small –db.t3.xlarge db.r5.large –db.r5.4xlarge db.r5b.large –db.r5b.4xlarge db.r5d.large –db.r5d.4xlarge db.r6i.large –db.r6i.4xlarge db.m5.large –db.m5.4xlarge db.m5d.large –db.m5d.4xlarge db.m6i.large –db.m6i.4xlarge db.z1d.large –db.z1d.13xlarge	db.t3.small –db.t3.2xlarge db.r5.large –db.r5.4xlarge db.r5b.large –db.r5b.4xlarge db.r5d.large –db.r5d.4xlarge db.r6i.large –db.r6i.4xlarge db.m5.large –db.m5.4xlarge db.m5d.large –db.m5d.4xlarge db.m6i.large –db.m6i.4xlarge db.z1d.large –db.z1d.3xlarge	db.t3.small –db.t3.2xlarge db.r5.large –db.r5.4xlarge db.r5b.large –db.r5b.4xlarge db.r5d.large –db.r5d.4xlarge db.r6i.large –db.r6i.4xlarge db.m5.large –db.m5.4xlarge db.m5d.large –db.m5d.4xlarge db.m6i.large –db.m6i.4xlarge db.z1d.large –db.z1d.3xlarge

SQL Server 版本	2022 支持范围	2019 支持范围	2017 和 2016 支持范围
精简版	`db.t3.micro –db.t3.xlarge`	`db.t3.micro –db.t3.xlarge`	`db.t3.micro –db.t3.xlarge`

Microsoft SQL Server 安全性

Microsoft SQL Server 数据库引擎使用基于角色的安全性。您在创建数据库实例时指定的主用户名是 SQL Server 身份验证登录名，该登录名是 `processadmin`、`public` 和 `setupadmin` 固定服务器角色的成员。

任何创建数据库的用户都将被分配该数据库的 db_owner 角色，并将拥有除用于备份外的所有数据库级别的权限。Amazon RDS 会为您管理备份。

以下服务器级别的角色在 Amazon RDS for SQL Server 中不可用：

- bulkadmin
- dbcreator
- diskadmin
- securityadmin
- serveradmin
- sysadmin

以下服务器级别的权限在 RDS for SQL Server 数据库实例上不可用：

- 更改任何数据库
- ALTER ANY EVENT NOTIFICATION
- ALTER RESOURCES
- ALTER SETTINGS（您可以使用数据库参数组 API 操作来修改参数；有关更多信息，请参阅 Amazon RDS 的参数组）
- AUTHENTICATE SERVER
- CONTROL_SERVER
- CREATE DDL EVENT NOTIFICATION

- CREATE ENDPOINT
- CREATE SERVER ROLE
- CREATE TRACE EVENT NOTIFICATION
- 删除任何数据库
- EXTERNAL ACCESS ASSEMBLY
- SHUTDOWN（您可转而使用 RDS 重启选项）
- UNSAFE ASSEMBLY
- ALTER ANY AVAILABILITY GROUP
- CREATE ANY AVAILABILITY GROUP

Microsoft SQL Server 数据库实例的 SSL 支持

可使用 SSL 对应用程序和运行 Microsoft SQL Server 的 Amazon RDS 数据库实例之间的连接进行加密。您还可强制至数据库实例的所有连接使用 SSL。如果强制所有连接使用 SSL，则对客户端是透明的，客户端不需要为使用 SSL 而执行任何操作。

在所有 Amazon 区域以及所有支持的 SQL Server 版本中支持 SSL。有关更多信息，请参阅 将 SSL 与 Microsoft SQL Server 数据库实例结合使用。

将 SSL 与 Microsoft SQL Server 数据库实例结合使用

可使用安全套接字层 (SSL) 对客户端应用程序和运行 Microsoft SQL Server 的 Amazon RDS 数据库实例之间的连接进行加密。对于所有支持的 SQL Server 版本，所有 Amazon 区域提供了 SSL 支持。

在您创建 SQL Server 数据库实例时，Amazon RDS 会为其创建 SSL 证书。SSL 证书会将数据库实例终端节点作为 SSL 证书的公用名 (CN) 包含在内以防止欺诈攻击。

使用 SSL 连接到 SQL Server 数据库实例有两种方法：

- 强制所有连接使用 SSL — 这种方法对客户端是透明的，客户端不需要为使用 SSL 而执行任何操作。

> **Note**
>
> 当您将 `rds.force_ssl` 设置为 1 并使用 SSMS 版本 19.3、20.0 和 20.2 时，请检查以下各项：
>
> - 在 SSMS 中启用信任服务器证书。
> - 将证书导入您的系统。

- 加密特定连接 — 这种方法在特定客户端计算机上设置 SSL 连接，您必须在客户端上执行加密连接的操作。

有关 SQL Server 的传输层安全性 (TLS) 支持的信息，请参阅 [Microsoft SQL Server 的 TLS 1.2 支持](#)。

强制与数据库实例的连接使用 SSL

您还可强制与数据库实例的所有连接使用 SSL。如果强制所有连接使用 SSL，则对客户端是透明的，客户端不需要为使用 SSL 而执行任何操作。

如果要强制使用 SSL，请使用 `rds.force_ssl` 参数。默认情况下，`rds.force_ssl` 参数设置为 `0 (off)`。将 `rds.force_ssl` 参数设置为 `1 (on)` 可强制连接使用 SSL。由于 `rds.force_ssl` 参数是静态的，因此，在更改值后，您必须重启数据库实例才能使更改生效。

强制与数据库实例的所有连接都使用 SSL

1. 确定附加到数据库实例的参数组：

   a. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。

   b. 在 Amazon RDS 控制台的右上角，选择数据库实例的Amazon区域。
   c. 在导航窗格中，选择数据库，然后选择数据库实例的名称以显示其详细信息。
   d. 选择 Configuration 选项卡。在此部分中查找参数组。
2. 如果需要，可创建新的参数组。如果数据库实例使用默认参数组，则必须创建新的参数组。如果数据库实例使用非默认参数组，则可以选择编辑现有参数组或创建新的参数组。如果您编辑现有参数组，则更改将影响使用该参数组的所有数据库实例。

   要创建新的参数组，请按照在 Amazon RDS 中创建数据库参数组中的说明执行操作。
3. 编辑新的或现有的参数组以将 `rds.force_ssl` 参数设置为 `true`。要编辑参数组，请按照在 Amazon RDS 中修改数据库参数组中的参数中的说明执行操作。
4. 如果您创建了一个新的参数组，可修改数据库实例以附加该新参数组。修改数据库实例的 DB Parameter Group 设置。有关更多信息，请参阅“修改 Amazon RDS 数据库实例”。
5. 重启数据库实例。有关更多信息，请参阅“重启中的数据库实例”。

加密特定连接

您可以强制与数据库实例的所有连接使用 SSL，也可以仅加密来自特定客户端计算机的连接。要在特定客户端中使用 SSL，您必须获取客户端计算机的证书、在客户端计算机中导入证书，然后加密来自该客户端计算机的连接。

> **Note**
>
> 在 2014 年 8 月 5 日之后创建的所有 SQL Server 实例都在 SSL 证书的公用名 (CN) 字段中使用数据库实例终端节点。在 2014 年 8 月 5 日之前，SSL 证书验证不可用于基于 VPC 的 SQL Server 实例。如果您具有在 2014 年 8 月 5 日之前创建的基于 VPC 的 SQL Server 数据库实例，而且要使用 SSL 证书验证并确保实例终端节点作为该数据库实例 SSL 证书的 CN 包括在内，请重命名该实例。当您重命名数据库实例时，新证书会进行部署，并且实例会重启以启用新证书。

获取客户端计算机的证书

要对从客户端计算机到运行 Microsoft SQL Server 的 Amazon RDS 数据库实例的连接进行加密，您需要客户端计算机上的证书。

要获取该证书，请将该证书下载到客户端计算机。有关下载证书的更多信息，请参阅使用 SSL/TLS 加密与数据库实例或集群的连接。

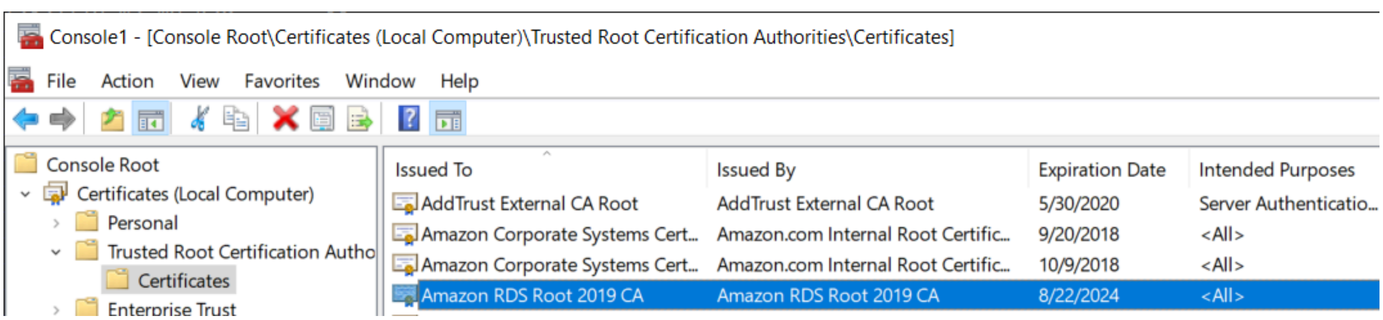
下载适当的证书后，使用以下部分中的过程将该证书导入 Microsoft Windows 操作系统。

将证书导入客户端计算机

您可以使用以下过程将证书导入客户端计算机上的 Microsoft Windows 操作系统。

将证书导入 Windows 操作系统：

1. 在 Start (开始) 菜单上的搜索框中，键入 **Run**，然后按 Enter。
2. 在 Open (打开) 框中，键入 **MMC**，然后选择 OK (确定)。
3. 在 MMC 控制台中的 File 菜单上，选择 Add/Remove Snap-in。
4. 在 Add or Remove Snap-ins (添加或删除管理单元) 对话框中，对于 Available snap-ins (可用管理单元)，选择 **Certificates**，然后选择 Add (添加)。
5. 在 Certificates snap-in 对话框中，选择 Computer account，然后选择 Next。
6. 在 Select computer 对话框中，选择 Finish。
7. 在 Add or Remove Snap-ins 对话框中，选择 OK。
8. 在 MMC 控制台中，展开 Certificates，打开 Trusted Root Certification Authorities 的上下文 (右键单击) 菜单，选择 All Tasks，然后选择 Import。
9. 在证书导入向导的第一页上，选择 Next。
10. 在证书导入向导的第二页上，选择 Browse。在浏览窗口中，将文件类型更改为 All files (*.*) (所有文件 (*.*))，因为 .pem 不是标准证书扩展名。找到您之前下载的 .pem 文件。
11. 选择 Open 以选择证书文件，然后选择 Next。
12. 在证书导入向导的第三页上，选择 Next。
13. 在证书导入向导的第四页上，选择 Finish。这将显示一个指示导入已成功的对话框。
14. 在 MMC 控制台中，展开 Certificates，再展开 Trusted Root Certification Authorities，然后选择 Certificates。找到证书以确认其存在，如此处所示。



对与运行 Microsoft SQL Server 的 Amazon RDS 数据库实例的连接进行加密

将证书导入客户端计算机后，您可以对从客户端计算机到运行 Microsoft SQL Server 的 Amazon RDS 数据库实例的连接进行加密。

对于 SQL Server Management Studio，使用以下过程。有关 SQL Server Management Studio 的更多信息，请参阅[使用 SQL Server Management Studio](#)。

对来自 SQL Server Management Studio 的连接进行加密

1. 启动 SQL Server Management Studio。
2. 为 Connect to server 键入服务器信息、登录用户名和密码。
3. 选择 Options。
4. 选择 Encrypt connection。
5. 选择 Connect。
6. 通过运行以下查询确认连接已加密。验证查询为 `true` 返回 `encrypt_option`。

```
select ENCRYPT_OPTION from SYS.DM_EXEC_CONNECTIONS where SESSION_ID = @@SPID
```

对于任何其他 SQL 客户端，请使用以下过程。

对来自其他 SQL 客户端的连接进行加密

1. 向连接字符串追加 `encrypt=true`。在 GUI 工具的连接页上，此字符串可能以选项或属性形式提供。

> **Note**
>
> 要为使用 JDBC 进行连接的客户端启用 SSL 加密，您可能需要将 Amazon RDS SQL 证书添加到 Java CA 证书 (cacerts) 存储。可通过使用 [keytool](#) 实用程序执行该操作。

2. 通过运行以下查询确认连接已加密。验证查询为 `true` 返回 `encrypt_option`。

```
select ENCRYPT_OPTION from SYS.DM_EXEC_CONNECTIONS where SESSION_ID = @@SPID
```

配置 SQL Server 安全协议和密码

您可以使用数据库参数打开和关闭某些安全协议和密码。您可以配置的安全参数（TLS 版本 1.2 除外）如下表所示。

数据库参数	允许的值（默认值以粗体显示）	描述
rds.tls10	默认值，启用，禁用	TLS 1.0。
rds.tls11	默认值，启用，禁用	TLS 1.1。
rds.tls12	默认值	TLS 1.2。您无法修改此值。
rds.fips	0, 1	将参数设置为 1 时，RDS 会强制使用符合美国联邦信息处理标准 (FIPS) 140-2 标准的模块。 有关详细信息，请参阅 Microsoft 文档中的 Use SQL Server 2016 in FIPS 140-2-compliant mode（在 FIPS 140-2 合规模式下使用 SQL Server 2016）。
rds.rc4	默认值，启用，禁用	RC4 流密码。
rds.diffie-hellman	默认值，启用，禁用	Diffie-Hellman key-exchange 加密。
rds.diffie-hellman-min-key-bit-length	默认值，1024，2048，3072，4096	Diffie-Hellman 密钥的最小位长度。
rds.curve25519	默认值，启用，禁用	Curve25519 elliptic-curve 加密密码。并非所有引擎版本都支持此参数。

数据库参数	允许的值（默认值以粗体显示）	描述
rds.3des168	默认值，启用，禁用	具有 168 位密钥长度的三重数据加密标准 (DES) 加密密码。

> **Note**
>
> 对于 16.00.4120.1、15.00.4365.2、14.00.3465.1、13.00.6435.1 和 12.00.6449.1 之后的次要引擎版本，数据库参数 `rds.tls10`、`rds.tls11`、`rds.rc4`、`rds.curve25519` 和 `rds.3des168` 的默认设置处于禁用状态。否则，默认设置为启用。
> 对于 16.00.4120.1、15.00.4365.2、14.00.3465.1、13.00.6435.1 和 12.00.6449.1 之后的次要引擎版本，`rds.diffie-hellman-min-key-bit-length` 的默认设置为 3072。否则，默认设置为 2048。

使用以下过程配置安全协议和密码：

1. 创建自定义数据库参数组。
2. 修改参数组中的参数。
3. 将数据库参数组与数据库实例相关联。

有关数据库参数组的更多信息，请参阅 [Amazon RDS 的参数组](#)。

创建安全相关的参数组

为与 SQL Server 版本和数据库实例版本相对应的安全相关参数创建参数组。

控制台

以下过程为 SQL Server 标准版 2016 创建一个参数组。

创建参数组

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择参数组。

3. 选择创建参数组。
4. 在创建参数组窗格中，执行以下操作：

    a. 对于参数组系列，选择 sqlserver-se-13.0。
    b. 对于组名称，输入参数组的标识符，如 **sqlserver-ciphers-se-13**。
    c. 对于描述，输入 **Parameter group for security protocols and ciphers**。
5. 选择创建。

CLI

以下过程为 SQL Server 标准版 2016 创建一个参数组。

创建参数组

- 运行以下命令之一。

    Example

    对于 Linux、macOS 或 Unix：

```
aws rds create-db-parameter-group \
    --db-parameter-group-name sqlserver-ciphers-se-13 \
    --db-parameter-group-family "sqlserver-se-13.0" \
    --description "Parameter group for security protocols and ciphers"
```

    对于 Windows：

```
aws rds create-db-parameter-group ^
    --db-parameter-group-name sqlserver-ciphers-se-13 ^
    --db-parameter-group-family "sqlserver-se-13.0" ^
    --description "Parameter group for security protocols and ciphers"
```

修改安全相关参数

修改与 SQL Server 版本和数据库实例版本对应的参数组中的安全相关参数。

控制台

以下过程修改您为 SQL Server 标准版 2016 创建的参数组。此示例关闭 TLS 版本 1.0。

修改参数组

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择参数组。
3. 选择参数组，例如 sqlserver-ciphers-se-13。
4. 在参数下，从参数列表中筛选 **rds**。
5. 选择编辑参数。
6. 选择 rds.tls10。
7. 对于值，选择已禁用。
8. 选择 Save changes（保存更改）。

CLI

以下过程修改您为 SQL Server 标准版 2016 创建的参数组。此示例关闭 TLS 版本 1.0。

修改参数组

- 运行以下命令之一。

  Example

  对于 Linux、macOS 或 Unix：

  ```
  aws rds modify-db-parameter-group \
      --db-parameter-group-name sqlserver-ciphers-se-13 \
      --parameters
   "ParameterName='rds.tls10',ParameterValue='disabled',ApplyMethod=pending-reboot"
  ```

  对于 Windows：

  ```
  aws rds modify-db-parameter-group ^
      --db-parameter-group-name sqlserver-ciphers-se-13 ^
      --parameters
   "ParameterName='rds.tls10',ParameterValue='disabled',ApplyMethod=pending-reboot"
  ```

将安全相关参数组与您的数据库实例关联

要将参数组与数据库实例关联，请使用 Amazon Web Services Management Console或 Amazon CLI。

控制台

您可以将参数组与新的或现有的数据库实例关联：

- 对于新的数据库实例，请在启动实例时将其关联。有关更多信息，请参阅“创建 Amazon RDS 数据库实例”。
- 对于现有数据库实例，请通过修改实例将其关联。有关更多信息，请参阅 修改 Amazon RDS 数据库实例。

CLI

您可以将参数组与新的或现有的数据库实例关联。

使用参数组创建数据库实例

- 指定在创建参数组时使用的相同数据库引擎类型和主要版本。

  Example

  对于 Linux、macOS 或 Unix：

```
aws rds create-db-instance \
    --db-instance-identifier mydbinstance \
    --db-instance-class db.m5.2xlarge \
    --engine sqlserver-se \
    --engine-version 13.00.5426.0.v1 \
    --allocated-storage 100 \
    --master-user-password secret123 \
    --master-username admin \
    --storage-type gp2 \
    --license-model li \
    --db-parameter-group-name sqlserver-ciphers-se-13
```

  对于 Windows：

```
aws rds create-db-instance ^
```

```
    --db-instance-identifier mydbinstance ^
    --db-instance-class db.m5.2xlarge ^
    --engine sqlserver-se ^
    --engine-version 13.00.5426.0.v1 ^
    --allocated-storage 100 ^
    --master-user-password secret123 ^
    --master-username admin ^
    --storage-type gp2 ^
    --license-model li ^
    --db-parameter-group-name sqlserver-ciphers-se-13
```

> **Note**
>
> 作为安全最佳实践，请指定除此处所示提示以外的密码。

修改数据库实例并关联参数组

- 运行以下命令之一。

  Example

  对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier mydbinstance \
    --db-parameter-group-name sqlserver-ciphers-se-13 \
    --apply-immediately
```

  对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier mydbinstance ^
    --db-parameter-group-name sqlserver-ciphers-se-13 ^
    --apply-immediately
```

更新应用程序以使用新的 SSL/TLS 证书连接到 Microsoft SQL Server 数据库实例

自 2023 年 1 月 13 日起，Amazon RDS 发布了新的证书颁发机构（CA）证书，以便使用安全套接字层或传输层安全性协议（SSL/TLS）连接到 RDS 数据库实例。接下来，您可以找到有关更新应用程序以使用新证书的信息。

本主题可帮助您确定是否有任何客户端应用程序使用 SSL/TLS 连接到您的数据库实例。如果是这样，您可以进一步检查这些应用程序是否需要证书验证才能连接。

> **Note**
>
> 某些应用程序配置为仅在它们可以成功验证服务器上的证书时才连接到 SQL Server 数据库实例。
> 对于此类应用程序，您必须更新客户端应用程序信任存储，以包括新的 CA 证书。

更新客户端应用程序信任存储中的 CA 证书后，可以在数据库实例上轮换这些证书。强烈建议在生产环境中实现这些过程之前，先在开发或测试环境中测试它们。

有关证书轮换的更多信息，请参阅轮换 SSL/TLS 证书。有关下载证书的更多信息，请参阅使用 SSL/TLS 加密与数据库实例或集群的连接。有关对 Microsoft SQL Server 数据库实例使用 SSL/TLS 的信息，请参阅将 SSL 与 Microsoft SQL Server 数据库实例结合使用。

主题

确定是否有任何应用程序使用 SSL 连接到 Microsoft SQL Server 数据库实例

检查数据库实例配置中 `rds.force_ssl` 参数的值。默认情况下，`rds.force_ssl` 参数设置为 0 (off)。如果 `rds.force_ssl` 参数设置为 1（开），则客户端需要使用 SSL/TLS 进行连接。有关参数组的更多信息，请参阅 Amazon RDS 的参数组。

运行以下查询来获取到数据库实例的所有打开的连接的当前加密选项。如果连接已加密，则列 `ENCRYPT_OPTION` 将返回 `TRUE`。

```
select SESSION_ID,
    ENCRYPT_OPTION,
    NET_TRANSPORT,
    AUTH_SCHEME
    from SYS.DM_EXEC_CONNECTIONS
```

此查询仅显示当前连接。它没有显示过去已连接并断开连接的应用程序是否使用了 SSL。

确定客户端是否需要证书验证才能连接

您可以检查不同类型的客户端是否需要证书验证才能连接。

> **Note**
> 如果您使用的连接器不是所列的连接器，请参阅特定连接器的文档以获取有关它如何执行加密连接的信息。有关更多信息，请参阅 Microsoft SQL Server 文档中的 Microsoft SQL 数据库的连接模块。

SQL Server Management Studio

检查是否对 SQL Server Management Studio 连接进行了加密：

1. 启动 SQL Server Management Studio。
2. 为连接到服务器输入服务器信息、登录用户名和密码。
3. 选择 Options。
4. 在连接页面中检查是否选择了加密连接。

有关 SQL Server Management Studio 的更多信息，请参阅使用 SQL Server Management Studio。

sqlcmd

以下使用 `sqlcmd` 客户端的示例显示了如何检查脚本的 SQL Server 连接以确定成功的连接是否需要有效证书。有关更多信息，请参阅 Microsoft SQL Server 文档中的使用 sqlcmd 进行连接。

当使用 `sqlcmd` 时，如果您使用 `-N` 命令参数对连接进行加密，则 SSL 连接需要对服务器证书进行验证，如以下示例所示。

```
$ sqlcmd -N -S dbinstance.rds.amazon.com -d ExampleDB
```

> **Note**
> 如果 sqlcmd 是使用 -C 选项调用的，则它将信任服务器证书，即使该证书与客户端信任存储不匹配也是如此。

ADO.NET

在以下示例中，应用程序使用 SSL 进行连接，并且必须验证服务器证书。

```
using SQLC = Microsoft.Data.SqlClient;

...

    static public void Main()
    {
        using (var connection = new SQLC.SqlConnection(
            "Server=tcp:dbinstance.rds.amazon.com;" +
            "Database=ExampleDB;User ID=LOGIN_NAME;" +
            "Password=YOUR_PASSWORD;" +
            "Encrypt=True;TrustServerCertificate=False;"
            ))
        {
            connection.Open();
            ...
        }
```

Java

在以下示例中，应用程序使用 SSL 进行连接，并且必须验证服务器证书。

```
String connectionUrl =
    "jdbc:sqlserver://dbinstance.rds.amazon.com;" +
    "databaseName=ExampleDB;integratedSecurity=true;" +
```

```
"encrypt=true;trustServerCertificate=false";
```

要为使用 JDBC 进行连接的客户端启用 SSL 加密，您可能需要将 Amazon RDS 证书添加到 Java CA 证书存储。有关说明，请参阅 Microsoft SQL Server 文档中的配置客户端加密。您还可以通过将 `trustStore=`*`path-to-certificate-trust-store-file`* 附加到连接字符串来直接提供受信任的 CA 证书文件名。

> **Note**
>
> 如果在连接字符串中使用 `TrustServerCertificate=true`（或其等效项），则连接过程将跳过信任链验证。在这种情况下，即使无法验证证书，应用程序也会连接。使用 `TrustServerCertificate=false` 执行证书验证，这是最佳实践。

更新应用程序信任存储

您可以更新使用 Microsoft SQL Server 的应用程序的信任存储。有关说明，请参阅 加密特定连接。另请参阅 Microsoft SQL Server 文档中的配置客户端加密。

如果您使用的操作系统不是 Microsoft Windows，请参阅 SSL/TLS 实现的软件分发文档，以获取有关添加新的根 CA 证书的信息。例如，OpenSSL 和 GnuTLS 是常用的选项。使用实现方法将信任添加到 RDS 根 CA 证书。Microsoft 提供了有关在某些系统上配置证书的说明。

有关下载根证书的信息，请参阅 使用 SSL/TLS 加密与数据库实例或集群的连接。

有关导入证书的示例脚本，请参阅 将证书导入信任存储的示例脚本。

> **Note**
>
> 更新信任存储时，除了添加新证书外，还可以保留较旧证书。

Microsoft SQL Server 数据库实例的合规性计划支持

Amazon范围内的服务已由第三方审核员完全评估，由此获得了证书、合规性证明或操作授权书 (ATO)。有关更多信息，请参阅合规性计划范围内的Amazon服务。

Microsoft SQL Server 数据库实例的 HIPAA 支持

您可以使用 Amazon RDS for Microsoft SQL Server 数据库来构建符合 HIPAA 标准的应用程序。您可以在与 Amazon 签订的业务合作协议 (BAA) 下存储与医疗保健相关的信息，包括受保护医疗信息 (PHI)。有关更多信息，请参阅 HIPAA 合规性。

Amazon RDS for SQL Server 支持在以下版本中使用 HIPAA：

- SQL Server 2022 企业版、标准版和 Web 版本
- SQL Server 2019 Enterprise、Standard 和 Web 版本
- SQL Server 2017 Enterprise、Standard 和 Web 版本
- SQL Server 2016 Enterprise、Standard 和 Web 版本

要启用对数据库实例的 HIPAA 支持，请设置以下三个组件。

组件	详细信息
审核	要设置审核，请将参数 `rds.sqlserver_audit` 的值设置为 `fedramp_hipaa`。如果数据库实例尚未使用自定义数据库参数组，您必须创建一个自定义参数组并将它附加到数据库实例，然后才能修改 `rds.sqlserver_audit` 参数。有关更多信息，请参阅"Amazon RDS 的参数组"。
传输加密	要设置传输加密，请强制与数据库实例的所有连接使用安全套接字层 (SSL)。有关更多信息，请参阅"强制与数据库实例的连接使用 SSL"。
静态加密	要设置静态加密，您有两种选择： 1. 如果您运行的是 SQL Server 2016–2022 企业版或 2022 标准版，则可以使用透明数据加密（TDE）来实现静态加密。有关更多信息，请参阅 SQL Server 中的透明数据加密支持。 2. 您可以使用 Amazon Key Management Service (Amazon KMS) 加密密钥设置静态加密。有关更多信息，请参阅 加密 Amazon RDS 资源。

Amazon RDS 上的 Microsoft SQL Server 版本

在创建新数据库实例时，您可以指定目前支持的任何 Microsoft SQL Server 版本。您可以指定 Microsoft SQL Server 主要版本（例如 Microsoft SQL Server 14.00），以及所指定主要版本的任何受支持的次要版本。如果没有指定版本，Amazon RDS 会默认使用支持的版本，通常是最新的版本。如果指定了主要版本但未指定次要版本，Amazon RDS 将默认使用您指定的主要版本的最新版本。

下表显示所有版本和所有 Amazon 区域支持的 SQL Server 版本，除非另有说明。

> **Note**
>
> 您也可以使用 [describe-db-engine-versions](https://docs.aws.amazon.com/cli/latest/reference/rds/describe-db-engine-versions.html) Amazon CLI 命令查看受支持版本的列表以及新建数据库实例的默认值。您可以运行 [describe-db-major-engine-versions](https://docs.aws.amazon.com/cli/latest/reference/rds/describe-db-major-engine-versions.html) Amazon CLI 命令或者使用 [DescribeDBMajorEngineVersions](https://docs.aws.amazon.com/AmazonRDS/latest/APIReference/API_DescribeDBMajorEngineVersions.html) RDS API 操作，来查看 SQL Server 数据库的主要版本。

主要版本	次要版本	RDS API **`EngineVersion`** 和 CLI **`engine-version`**
SQL Server 2022	16.00.4185.3（CU18）	`16.00.4185.3.v1`
	16.00.4175.1（CU17）	`16.00.4175.1.v1`
	16.00.4165.4（CU16）	`16.00.4165.4.v1`
	16.00.4150.1（CU15）	`16.00.4150.1.v1`
	16.00.4140.3（CU14 GDR）	`16.00.4140.3.v1`
	16.00.4135.4（CU14）	`16.00.4135.4.v1`
	16.00.4131.2（CU13）	`16.00.4131.2.v1`
	16.00.4125.3（CU13）	`16.00.4125.3.v1`
	16.00.4120.1（CU12 GDR）	`16.00.4120.1.v1`
	16.00.4115.5（CU12）	`16.00.4115.5.v1`
	16.00.4105.2（CU11）	

主要版本	次要版本	RDS API **EngineVersion** 和 CLI **engine-version**

	16.00.4095.4（CU10）
	16.00.4085.2（CU9）
		`16.00.4105.2.v1`
		`16.00.4095.4.v1`
		`16.00.4085.2.v1`

主要版本	次要版本	RDS API **EngineVersion** 和 CLI **engine-version**
SQL Server 2019	15.00.4430.1（CU32）	15.00.4430.1.v1
	15.00.4420.2（CU31）	15.00.4420.2.v1
	15.00.4415.2（CU30）	15.00.4415.2.v1
	15.00.4410.1（CU29 GDR）	15.00.4410.1.v1
	15.00.4395.2（CU28）	15.00.4395.2.v1
	15.00.4390.2（CU28）	15.00.4390.2.v1
	15.00.4385.2（CU28）	15.00.4385.2.v1
	15.00.4382.1（CU27）	15.00.4382.1.v1
	15.00.4375.4（CU27）	15.00.4375.4.v1
	15.00.4365.2（CU26）	15.00.4365.2.v1
	15.00.4355.3（CU25）	15.00.4355.3.v1
	15.00.4345.5（CU24）	15.00.4345.5.v1
	15.00.4335.1（CU23）	15.00.4335.1.v1
	15.00.4322.2（CU22）	15.00.4322.2.v1
	15.00.4316.3（CU21）	15.00.4316.3.v1
	15.00.4312.2（CU20）	15.00.4312.2.v1
	15.00.4236.7 (CU16)	15.00.4236.7.v1
	15.00.4198.2 (CU15)	15.00.4198.2.v1
	15.00.4153.1 (CU12)	15.00.4153.1.v1
	15.00.4073.23 (CU8)	15.00.4073.23.v1

主要版本	次要版本	RDS API **EngineVersion** 和 CLI **engine-version**
	15.00.4043.16 (CU5)	`15.00.4043.16.v1`
SQL Server 2017	14.00.3485.1（CU31 GDR）	`14.00.3485.1.v1`
	14.00.3480.1（CU31）	`14.00.3480.1.v1`
	14.00.3475.1（CU31）	`14.00.3475.1.v1`
	14.00.3471.2（CU31）	`14.00.3471.2.v1`
	14.00.3465.1（CU31）	`14.00.3465.1.v1`
	14.00.3460.9（CU31）	`14.00.3460.9.v1`
	14.00.3451.2 (CU30)	`14.00.3451.2.v1`
	14.00.3421.10 (CU27)	`14.00.3421.10.v1`
	14.00.3401.7 (CU25)	`14.00.3401.7.v1`
	14.00.3381.3 (CU23)	`14.00.3381.3.v1`
	14.00.3356.20 (CU22)	`14.00.3356.20.v1`
	14.00.3294.2 (CU20)	`14.00.3294.2.v1`
	14.00.3281.6 (CU19)	`14.00.3281.6.v1`

主要版本	次要版本	RDS API **EngineVersion** 和 CLI **engine-version**
SQL Server 2016	13.00.6455.2（GDR）	`13.00.6455.2.v1`
	13.00.6450.1（GDR）	`13.00.6450.1.v1`
	13.00.6445.1（GDR）	`13.00.6445.1.v1`
	13.00.6441.1（GDR）	`13.00.6441.1.v1`
	13.00.6435.1（GDR）	`13.00.6435.1.v1`
	13.00.6430.49（GDR）	`13.00.6430.49.v1`
	13.00.6419.1（SP3 + 补丁）	`13.00.6419.1.v1`
	13.00.6300.2 (SP3)	`13.00.6300.2.v1`

Amazon RDS 中的版本管理

Amazon RDS 包括灵活的版本管理功能，让您可以控制何时以及如何修补或升级您的数据库实例。这让您可对数据库引擎执行以下操作：

- 维持与数据库引擎修补程序版本的兼容性。
- 在将新修补程序版本部署到生产环境之前，先测试它们，以确定它们可与您的应用程序一起使用。
- 规划和执行版本升级以满足您的服务水平协议和计时要求。

Amazon RDS 中的 Microsoft SQL Server 引擎修补

Amazon RDS 会定期将官方 Microsoft SQL Server 数据库修补程序聚合到 Amazon RDS 特定的数据库实例引擎版本。有关每个引擎版本中的 Microsoft SQL Server 修补程序的更多信息，请参阅 Amazon RDS 上的版本和功能支持。

目前，您对数据库实例手动执行所有引擎升级。有关更多信息，请参阅"升级 Microsoft SQL Server 数据库引擎"。

Amazon RDS 上的 Microsoft SQL Server 主要引擎版本的弃用时间表

下表显示了 Microsoft SQL Server 主要引擎版本的计划弃用时间表。

日期	信息
2024 年 7 月 9 日	Microsoft 将停止 SQL Server 2014 的关键补丁更新。有关更多信息，请参阅 Microsof SQL Server 2014。
2024 年 6 月 1 日	Amazon RDS 计划在 RDS for SQL Server 上终止对 Microsoft SQL Server 2014 的支 实例都将被安排迁移到 SQL Server 2016（最新可用的次要版本）。有关更多信息，请 RDS for SQL Server 终止对 SQL Server 2014 主要版本的支持。 为避免从 Microsoft SQL Server 2014 自动升级，您可以在方便的时候进行升级。有关 级数据库实例引擎版本。
2022 年 7 月 12 日	Microsoft 将停止 SQL Server 2012 的关键修补程序更新。有关详细信息，请参阅 Mic Microsoft SQL Server 2012。
2022 年 6 月 1 日	Amazon RDS 计划在 RDS for SQL Server 上终止对 Microsoft SQL Server 2012 的支 实例都将被安排迁移到 SQL Server 2014（最新可用的次要版本）。有关更多信息，请 RDS for SQL Server 终止对 SQL Server 2012 主要版本的支持。 为避免从 Microsoft SQL Server 2012 自动升级，您可以在方便的时候进行升级。有关 级数据库实例引擎版本。
2021 年 9 月 1 日	Amazon RDS 开始禁止使用 Microsoft SQL Server 2012 为 SQL Server 数据库实例创 信息，请参阅公告：Amazon RDS for SQL Server 终止对 SQL Server 2012 主要版本
2019 年 7 月 12 日	Amazon RDS 团队已在 2019 年 6 月停止为 Microsoft SQL Server 2008 R2 提供支持 Server 2008 R2 实例正在迁移到 SQL Server 2012（最新可用的次要版本）。 为避免从 Microsoft SQL Server 2008 R2 自动升级，您可以在方便的时候进行升级。 阅“升级数据库实例引擎版本”。

日期	信息
2019 年 4 月 25 日	在 2019 年 4 月底之前，您将不能再使用 Microsoft SQL Server 2008R2 创建新的 Am Server 数据库实例。

Amazon RDS 上的 Microsoft SQL Server 功能

Amazon RDS 上支持的 SQL Server 版本包括以下功能。一般来说，除非 Microsoft 文档中另有说明，否则版本中也包括以前版本的功能。

主题

Microsoft SQL Server 2022 功能

SQL Server 2022 包括很多新功能，例如以下各项：

- 参数敏感计划优化 – 允许单个参数化语句有多个缓存计划，从而可能减少有关参数嗅探的问题。
- SQL Server Ledger – 提供以加密方式证明您的数据未经授权而未被更改的功能。
- 事务日志文件增长事件的即时文件初始化 – 可以更快地执行高达 64MB 的日志增长事件，包括对于启用了 TDE 的数据库。
- 系统页面锁存并发增强 – 在分配和取消分配数据页和范围时减少页面锁存争用，为 `tempdb` 繁重的工作负载提供显著的性能增强。

有关 SQL Server 2022 功能的完整列表，请参阅 Microsoft 文档中的 SQL Server 2022（16.x）中的新增功能。

有关不支持特征的列表，请参阅 不支持的功能以及有限支持的功能。

Microsoft SQL Server 2019 功能

SQL Server 2019 包括很多新功能，例如以下内容：

- 加速数据库恢复 (ADR) – 减少重新启动或长时间运行的事务回滚后的崩溃恢复时间。
- 智能查询处理 (IQP)：
  - 行模式内存授予反馈 – 自动更正多余的授权，否则会导致内存浪费并降低并发性。
  - 行存储上的批处理模式 – 为分析工作负载启用批处理模式执行，而不需要列存储索引。
  - 表变量延迟编译 – 提高了引用表变量的查询的计划质量和整体性能。
- 智能性能：
  - `OPTIMIZE_FOR_SEQUENTIAL_KEY` 索引选项 – 提高向索引进行高并发性插入的吞吐量。
  - 改进了间接检查点可扩展性 – 帮助具有繁重 DML 工作负载的数据库。
  - 并发页可用空间 (PFS) 更新 – 允许作为共享闩锁而非独占闩锁进行处理。
- 监控改进：
  - `WAIT_ON_SYNC_STATISTICS_REFRESH` 等待类型 – 显示在同步统计信息刷新操作上花费的累积实例级时间。
  - 数据库范围内的配置 – 包括 `LIGHTWEIGHT_QUERY_PROFILING` 和 `LAST_QUERY_PLAN_STATS`。
  - 动态管理功能 (DMF) – 包括 `sys.dm_exec_query_plan_stats` 和 `sys.dm_db_page_info`。
- 详细截断警告 – 数据截断错误消息默认包括表名和列名以及截断的值。
- 可恢复的联机索引创建 – 在 SQL Server 2017 中，只支持可恢复的联机索引重建。

有关 SQL Server 2019 功能的完整列表，请参阅 Microsoft 文档中的 SQL Server 2019（15.x）中的新增功能。

有关不支持功能的列表，请参阅 不支持的功能以及有限支持的功能。

Microsoft SQL Server 2017 功能

SQL Server 2017 包括很多新功能，例如以下内容：

- 自适应查询处理
- 自动更正计划（自动优化功能）
- GraphDB
- 可恢复的索引重建

有关 SQL Server 2017 功能的完整列表，请参阅 Microsoft 文档中的 SQL Server 2017 中的新增功能。

有关不支持功能的列表，请参阅 不支持的功能以及有限支持的功能。

Microsoft SQL Server 2016 功能

Amazon RDS 支持 SQL Server 2016 的以下功能：

- 始终加密
- JSON 支持
- 运行分析
- 查询存储
- 临时表

有关 SQL Server 2016 功能的完整列表，请参阅 Microsoft 文档中的 SQL Server 2016 中的新增功能。

Amazon RDS 上的 Microsoft SQL Server 2014 支持终止

SQL Server 2014 在 Amazon RDS 上的支持已经终止。

RDS 正在将仍在使用 SQL Server 2014 的所有现有数据库实例升级到 SQL Server 2016 的最新次要版本。有关更多信息，请参阅 Amazon RDS 中的版本管理。

Amazon RDS 上的 Microsoft SQL Server 2012 支持终止

SQL Server 2012 在 Amazon RDS 上的支持已经终止。

RDS 正在将仍在使用 SQL Server 2012 的所有现有数据库实例升级到 SQL Server 2016 的最新次要版本。有关更多信息，请参阅 Amazon RDS 中的版本管理。

Amazon RDS 上的 Microsoft Server 2008 R2 支持终止

SQL Server 2008 R2 在 Amazon RDS 上的支持已经终止。

RDS 正在将仍在使用 SQL Server 2008 R2 的所有现有实例升级到 SQL Server 2012 的最新次要版本。有关更多信息，请参阅 Amazon RDS 中的版本管理。

适用于 Microsoft SQL Server 数据库实例的更改数据捕获支持

Amazon RDS 支持运行 Microsoft SQL Server 的数据库实例的更改数据捕获 (CDC)。CDC 捕获对表中数据所做的更改，并存储有关您稍后可访问的每项更改的元数据。有关更多信息，请参阅 Microsoft 文档中的更改数据捕获。

Amazon RDS 支持在以下 SQL Server 版本中使用 CDC：

- Microsoft SQL Server 企业版（所有版本）
- Microsoft SQL Server 标准版：
  - 2022
  - 2019
  - 2017
  - 2016 版本 13.00.4422.0 SP1 CU2 及更高版本

要将 CDC 用于 Amazon RDS 数据库实例，请先使用 RDS 提供的存储过程在数据库级别启用或禁用 CDC。之后，担任该数据库的 `db_owner` 角色的任何用户均可使用本机 Microsoft 存储过程来控制该数据库上的 CDC。有关更多信息，请参阅“在 Amazon RDS for SQL Server 上使用变更数据捕获”。

您可以使用 CDC 和 Amazon Database Migration Service 启用来自 SQL Server 数据库实例的持续复制。

不支持的功能以及有限支持的功能

Amazon RDS 中不支持以下 Microsoft SQL Server 功能：

- Always On 可用性组 (AG)
- 备份到 Microsoft Azure Blob 存储
- 缓冲池扩展
- 自定义密码策略

- 数据质量服务
- 数据库日志传输
- 数据库快照（Amazon RDS 仅支持数据库实例快照）
- 扩展存储过程，包括 xp_cmdshell
- FILESTREAM 支持
- 文件表
- Machine Learning 和 R 服务（需要操作系统访问权限才能安装）
- 维护计划
- 性能数据收集器
- 基于策略的管理
- PolyBase
- 复制
- 资源管理器

  在多租户环境中，建议您了解性能要求和注意事项，以显著减少由于工作负载争夺资源而导致的问题。
- 服务器级别触发器
- 服务代理端点
- 拉伸数据库
- TRUSTWORTHY 数据库属性（需要 sysadmin 角色）
- T-SQL 端点 (使用 CREATE ENDPOINT 的所有操作不可用)
- WCF Data Services

Amazon RDS 上对以下 Microsoft SQL Server 功能提供有限支持：

- 分布式查询/链接服务器。有关更多信息，请参阅通过 Amazon RDS for Microsoft SQL Server 实施链接服务器。
- 通用运行时语言 (CLR)。在适用于 SQL Server 2016 和更低版本的 RDS 上，在 SAFE 模式中支持 CLR 并仅使用汇编位。在适用于 SQL Server 2017 和更高版本的 RDS 上不支持 CLR。有关更多信息，请参阅 Microsoft 文档中的通用运行时语言集成。
- Amazon RDS for SQL Server 中带有 Oracle OLEDB 的链接服务器。有关更多信息，请参阅 在 Amazon RDS for SQL Server 中支持带有 Oracle OLEDB 的链接服务器。

对于 SQL Server 2022，Amazon RDS 中不支持以下功能：

- 暂停数据库以获取快照
- 外部数据来源
- 备份和还原到 S3 兼容的对象存储
- 对象存储集成
- TLS 1.3 和 MS-TDS 8.0
- 使用 QAT 分载备份压缩
- SQL Server Analysis Services (SSAS)
- 使用多可用区部署进行数据库镜像。SQL Server Always On 是多可用区部署中唯一受支持的方法。

使用 Microsoft SQL Server 数据库镜像或 Always On 可用性组的多可用区部署

Amazon RDS 通过使用 SQL Server 数据库镜像 (DBM) 或 Always On 可用性组 (AG) 支持为运行 Microsoft SQL Server 的数据库实例进行多可用区部署。多可用区部署为数据库实例提供了更高的可用性、数据持久性和容错能力。在进行计划的数据维护或发生未计划的服务中断时，Amazon RDS 将自动失效转移到最新辅助副本，以便数据库操作能够在不进行手动干预的情况下快速恢复。主实例和辅助实例使用相同的端点，其物理网络地址作为失效转移过程的一部分转换到被动辅助副本。在失效转移时，您不必重新配置应用程序。

Amazon RDS 管理失效转移的方式是：主动监控多可用区部署并在主实例发生问题时主动启动失效转移。除非备用实例和主要实例完全同步，否则失效转移不会发生。Amazon RDS 自动修复运行状况不佳的数据库实例并重新建立同步复制，主动维护您的多可用区部署。您不必管理任何内容。Amazon RDS 负责处理主实例、见证以及备用实例。当您设置 SQL Server 多可用区时，RDS 为实例上的所有数据库配置被动辅助实例。

有关更多信息，请参阅“Amazon RDS for Microsoft SQL Server 多可用区部署”。

使用透明数据加密来加密静态数据

Amazon RDS 支持 Microsoft SQL Server 透明数据加密（TDE），此方法以透明方式对存储数据进行加密。Amazon RDS 利用选项组启用和配置这些功能。有关 TDE 选项的更多信息，请参阅 SQL Server 中的透明数据加密支持。

适用于 Amazon RDS for Microsoft SQL Server 的函数和存储过程

在下面，您可以找到有助于自动执行 SQL Server 任务的 Amazon RDS 函数和存储过程的列表。

任务类型	过程或函数	其使用位置
管理任务	rds_drop_database	删除 Amazon RDS for Microsoft SQL Server 数据库实例中的数据库
	rds_failover_time	确定 Amazon RDS for SQL Server 的上次失效转移时间
	rds_manage_view_db_permission	拒绝或支持查看 Amazon RDS for SQL Server 的数据库名称
	rds_modify_db_name	在多可用区部署中重命名 Amazon RDS for Microsoft SQL Server 数据库
	rds_read_error_log	查看错误和代理日志
	rds_set_configuration	此操作用于设置各种数据库实例配置： • 多可用区实例的更改数据捕获 • 设置跟踪文件和转储文件的保留期 • 压缩备份文件
	rds_set_database_online	将 Amazon RDS for SQL Server 数据库从脱机状态转换为联机状态
	rds_set_system_database_syn	启用 SQL Server Agent 作业复制

任务类型	过程或函数	其使用位置
	c_objects	
	rds_fn_get_system_database_sync_objects	
	rds_fn_server_object_last_sync_time	
	rds_show_configuration	要查看使用 rds_set_configuration 设置的值，请参阅这些主题： • 多可用区实例的更改数据捕获 • 设置跟踪文件和转储文件的保留期
	rds_shrink_tempdbfile	缩减临时数据库
更改数据捕获 (CDC)	rds_cdc_disable_db	禁用 CDC
	rds_cdc_enable_db	启用 CDC
数据库邮件	rds_fn_sysmail_allitems	查看消息、日志和附件

任务类型	过程或函数	其使用位置
	rds_fn_sysmail_event_log	查看消息、日志和附件
	rds_fn_sysmail_mailattachments	查看消息、日志和附件
	rds_sysmail_control	此操作用于启动和停止邮件队列： • 启动邮件队列 • 停止邮件队列
	rds_sysmail_delete_mailitems_sp	删除消息
本机备份和恢复	rds_backup_database	备份数据库
	rds_cancel_task	取消任务
	rds_finish_restore	完成数据库还原
	rds_restore_database	还原数据库
	rds_restore_log	还原日志

任务类型	过程或函数	其使用位置
Amazon S3 文件传输	rds_delete_from_filesystem	删除 RDS 数据库实例上的文件
	rds_download_from_s3	将文件从 Amazon S3 存储桶下载到 SQL Server 数据库实例
	rds_gather_file_details	列出 RDS 数据库实例上的文件
	rds_upload_to_s3	将文件从 SQL Server 数据库实例上传到 Amazon S3 存储桶
Microsoft 分布式事务处理协调器 (MSDTC)	rds_msdtc_transaction_tracing	使用事务跟踪
SQL Server Audit	rds_fn_get_audit_file	查看审核日志

任务类型	过程或函数	其使用位置
透明数据加密	rds_backup_tde_certificate rds_drop_tde_certificate rds_restore_tde_certificate rds_fn_list_user_tde_certificates	SQL Server 中的透明数据加密支持

任务类型	过程或函数	其使用位置
Microsoft 商业智能 (MSBI)	rds_msbi_task	此操作与 SQL Server Analysis Services (SSAS) 一起使用： • 在 Amazon RDS 上部署 SSAS 项目 • 添加域用户作为数据库管理员 • 备份 SSAS 数据库 • 还原 SSAS 数据库 此操作也与 SQL Server Integration Services (SSIS) 一起使用： • SSISDB 上的管理权限 • 部署 SSIS 项目 此操作也与 SQL Server Reporting Services (SSRS) 一起使用： • 将访问权限授予域用户 • 撤销系统级权限
	rds_fn_task_status	此操作显示 MSBI 任务的状态： • SSAS：监控部署任务的状态 • SSIS：监控部署任务的状态 • SSRS：监控任务的状态
SSIS	rds_drop_ssis_database	删除 SSISDB 数据库
	rds_sqlagent_proxy	创建 SSIS 代理

任务类型	过程或函数	其使用位置
SSRS	rds_drop_ssrs_databases	删除 SSRS 数据库

Microsoft SQL Server 数据库实例的本地时区

默认情况下，已设置运行 Microsoft SQL Server 的 Amazon RDS 数据库实例的时区。当前默认值为协调世界时 (UTC)。您可改为将数据库实例的时区设置为本地时区以匹配应用程序的时区。

首次创建数据库实例时将设置时区。您可以使用 Amazon Web Services Management Console、Amazon RDS API CreateDBInstance 操作或 Amazon CLI create-db-instance 命令创建数据库实例。

如果数据库实例是多可用区部署（使用 SQL Server DBM 或 AG）的一部分，则当您进行故障转移时，您的时区将保留您设置的本地时区。有关更多信息，请参阅“使用 Microsoft SQL Server 数据库镜像或 Always On 可用性组的多可用区部署”。

在请求时间点还原时，请指定要还原到的时间。时间以您的当地时区显示。有关更多信息，请参阅“将 Amazon RDS 的数据库实例还原到指定时间”。

下面是在数据库实例上设置本地时区时的限制：

- 您无法修改现有 SQL Server 数据库实例的时区。
- 您无法将快照从一个时区的数据库实例还原至另一个时区的数据库实例。
- 强烈建议您不要将备份文件从一个时区还原到另一个时区。如果您将备份文件从一个时区还原至为另一个时区，则必须审查您的查询和应用程序来查看时区更改的效果。有关更多信息，请参阅“使用本机备份和还原导入和导出 SQL Server 数据库”。

支持的时区

您可以将本地时区设置为下表中列出的值之一。

时区	标准时间偏差	描述	备注
阿富汗标准时间	(UTC+04:30)	喀布尔	此时区不遵守夏令时。
阿拉斯加标准时间	(UTC–09:00)	阿拉斯加州	
阿留申群岛标准时间	(UTC–10:00)	阿留申群岛	
阿尔泰标准时间	(UTC+07:00)	巴尔瑙尔，戈尔诺-阿尔泰斯克	
阿拉伯标准时间	(UTC+03:00)	科威特，利雅得	此时区不遵守夏令时。
阿拉伯标准时间	(UTC+04:00)	阿布扎比、马斯喀特	
阿拉伯标准时间	(UTC+03:00)	巴格达	此时区不遵守夏令时。
阿根廷标准时间	(UTC–03:00)	布宜诺斯艾利斯市	此时区不遵守夏令时。
阿斯特拉罕标准时间	(UTC+04:00)	阿斯特拉罕，乌里扬诺夫斯克	
大西洋标准时间	(UTC–04:00)	大西洋时间 (加拿大)	
澳大利亚中部标准时间	(UTC+09:30)	达尔文	此时区不遵守夏令时。
澳大利亚中西部标准时间	(UTC+08:45)	尤克拉	
澳大利亚东部标准时间	(UTC+10:00)	堪培拉、墨尔本、悉尼	
阿塞拜疆标准时间	(UTC+04:00)	巴库	
亚速尔群岛标准时间	(UTC–01:00)	亚速尔群岛	

时区	标准时间偏差	描述	备注
巴伊亚标准时间	(UTC–03:00)	萨尔瓦多	
孟加拉国标准时间	(UTC+06:00)	达卡	此时区不遵守夏令时。
白俄罗斯标准时间	(UTC+03:00)	明斯克	此时区不遵守夏令时。
布干维尔岛标准时间	(UTC+11:00)	布干维尔岛	
加拿大中部标准时间	(UTC–06:00)	萨斯喀彻温省	此时区不遵守夏令时。
佛得角标准时间	(UTC–01:00)	佛得角群岛	此时区不遵守夏令时。
高加索标准时间	(UTC+04:00)	埃里温	
澳大 利亚中部标准时间	(UTC+09:30)	阿德莱德	
中美洲标准时间	(UTC–06:00)	中美洲	此时区不遵守夏令时。
中亚标准时间	(UTC+06:00)	阿斯塔纳	此时区不遵守夏令时。
巴西中部标准时间	(UTC–04:00)	库亚巴	
中欧标准时间	(UTC+01:00)	贝尔格莱德、布拉迪斯拉发、布达佩斯、卢布尔雅那、布拉格	
中欧标准时间	(UTC+01:00)	萨拉热窝、斯科普里、华沙、萨格勒布	
太平洋中部标准时间	(UTC+11:00)	索罗门群岛、新喀里多尼亚	此时区不遵守夏令时。

时区	标准时间偏差	描述	备注
中部标准时间	(UTC–06:00)	中部时间 (美国和加拿大)	
中部标准时间（墨西哥）	(UTC–06:00)	瓜达拉哈拉、墨西哥城、蒙特雷	
查塔姆群岛标准时间	(UTC+12:45)	查塔姆群岛	
中国标准时间	(UTC+08:00)	北京、重庆、香港特别行政区、乌鲁木齐	此时区不遵守夏令时。
古巴标准时间	(UTC–05:00)	哈瓦那	
国际日期变更线标准时间	(UTC–12:00)	国际日期变更线西	此时区不遵守夏令时。
东非标准时间	(UTC+03:00)	内罗毕	此时区不遵守夏令时。
澳大利亚东部标准时间	(UTC+10:00)	布里斯班	此时区不遵守夏令时。
东欧标准时间	(UTC+02:00)	基希讷乌	
东部南美洲标准时间	(UTC–03:00)	巴西利亚	
复活节岛标准时间	(UTC–06:00)	复活节岛	
东部标准时间	(UTC–05:00)	东部时间 (美国和加拿大)	
东部标准时间（墨西哥）	(UTC–05:00)	切图马尔	
埃及标准时间	(UTC+02:00)	开罗	
叶卡捷琳堡标准时间	(UTC+05:00)	叶卡捷琳堡	
斐济标准时间	(UTC+12:00)	斐济	

时区	标准时间偏差	描述	备注
FLE 标准时间	(UTC+02:00)	赫尔辛基，基辅，里加，索非亚，塔林，维尔纽斯	
格鲁吉亚标准时间	(UTC+04:00)	第比利斯	此时区不遵守夏令时。
GMT 标准时间	(UTC)	都柏林、爱丁堡、里斯本、伦敦	此时区与格林威治标准时间不同。此时区遵守夏令时。
格陵兰标准时间	(UTC–03:00)	格陵兰	
格林威治标准时间	(UTC)	蒙罗维亚、雷克雅未克	此时区不遵守夏令时。
GTB 标准时间	(UTC+02:00)	雅典、布加勒斯特	
海地标准时间	(UTC–05:00)	海地	
夏威夷标准时间	(UTC–10:00)	夏威夷州	
印度标准时间	(UTC+05:30)	钦奈、加尔各答、孟买、新德里	此时区不遵守夏令时。
伊朗标准时间	(UTC+03:30)	德黑兰	
以色列标准时间	(UTC+02:00)	耶路撒冷	
约旦标准时间	(UTC+02:00)	阿曼	
加里宁格勒标准时间	(UTC+02:00)	加里宁格勒	
堪察加标准时间	(UTC+12:00)	彼得罗巴甫洛夫斯克-堪察加 – 旧	
韩国标准时间	(UTC+09:00)	首尔	此时区不遵守夏令时。

时区	标准时间偏差	描述	备注
利比亚标准时间	(UTC+02:00)	的黎波里	
莱恩群岛标准时间	(UTC+14:00)	圣诞岛	
豪勋爵岛标准时间	(UTC+10:30)	豪勋爵岛	
马加丹标准时间	(UTC+11:00)	马加丹	此时区不遵守夏令时。
麦哲伦标准时间	(UTC–03:00)	蓬塔阿雷纳斯	
马克萨斯群岛标准时间	(UTC–09:30)	马克萨斯群岛	
毛里求斯标准时间	(UTC+04:00)	路易港	此时区不遵守夏令时。
中东标准时间	(UTC+02:00)	贝鲁特	
蒙得维的亚标准时间	(UTC–03:00)	蒙得维的亚	
摩洛哥标准时间	(UTC+01:00)	卡萨布兰卡	
山地标准时间	(UTC–07:00)	山区时间 (美国和加拿大)	
山地标准时间（墨西哥）	(UTC–07:00)	奇瓦瓦、拉巴斯、马萨特兰	
缅甸标准时间	(UTC+06:30)	仰光	此时区不遵守夏令时。
中亚北部标准时间	(UTC+07:00)	新西伯利亚	
纳米比亚标准时间	(UTC+02:00)	温得和克	
尼泊尔标准时间	(UTC+05:45)	加德满都	此时区不遵守夏令时。
新西兰标准时间	(UTC+12:00)	奥克兰、惠灵顿	

时区	标准时间偏差	描述	备注
纽芬兰标准时间	(UTC–03:30)	纽芬兰省	
诺福克岛标准时间	(UTC+11:00)	诺福克岛	
北亚东部标准时间	(UTC+08:00)	伊尔库茨克	
北亚标准时间	(UTC+07:00)	克拉斯诺亚尔斯克	
朝鲜标准时间	(UTC+09:00)	平壤	
鄂木斯克标准时间	(UTC+06:00)	鄂木斯克	
太平洋 SA 标准时间	(UTC–03:00)	圣地亚哥	
太平洋标准时间	(UTC–08:00)	太平洋时间 (美国和加拿大)	
太平洋标准时间（墨西哥）	(UTC–08:00)	下加利福尼亚州	
巴基斯坦标准时间	(UTC+05:00)	伊斯兰堡，卡拉奇	此时区不遵守夏令时。
巴拉圭标准时间	(UTC–04:00)	亚松森	
罗马标准时间	(UTC+01:00)	布鲁塞尔，哥本哈根，马德里，巴黎	
俄罗斯时区 10	(UTC+11:00)	乔库尔达赫	
俄罗斯时区 11	(UTC+12:00)	阿纳德尔，堪察加彼得罗巴甫洛夫斯克	
俄罗斯时区 3	(UTC+04:00)	伊热夫斯克，萨马拉	
俄罗斯标准时间	(UTC+03:00)	莫斯科、圣彼德堡、伏尔加格勒	此时区不遵守夏令时。
南美洲东部标准时间	(UTC–03:00)	卡宴，福塔雷萨	此时区不遵守夏令时。

时区	标准时间偏差	描述	备注
SA 太平洋标准时间	(UTC–05:00)	波哥大、利马、基多、里约布兰科	此时区不遵守夏令时。
南美洲西部标准时间	(UTC–04:00)	乔治敦，拉巴斯，马瑙斯，圣胡安	此时区不遵守夏令时。
圣皮埃尔标准时间	(UTC–03:00)	圣皮埃尔和密克隆群岛	
萨哈林标准时间	(UTC+11:00)	萨哈林	
萨摩亚群岛标准时间	(UTC+13:00)	萨摩亚群岛	
圣多美标准时间	(UTC+01:00)	圣多美	
萨拉托夫标准时间	(UTC+04:00)	萨拉托夫	
东南亚标准时间	(UTC+07:00)	曼谷、河内、雅加达	此时区不遵守夏令时。
新加坡标准时间	(UTC+08:00)	吉隆坡、新加坡	此时区不遵守夏令时。
南非标准时间	(UTC+02:00)	哈拉雷，比勒陀利亚	此时区不遵守夏令时。
斯里兰卡标准时间	(UTC+05:30)	斯里哈亚华登尼普拉	此时区不遵守夏令时。
苏丹标准时间	(UTC+02:00)	喀土穆	
叙利亚标准时间	(UTC+02:00)	大马士革	
台北标准时间	(UTC+08:00)	台北	此时区不遵守夏令时。
塔斯马尼亚岛标准时间	(UTC+10:00)	霍巴特	
托坎廷斯标准时间	(UTC–03:00)	阿拉瓜伊纳	

时区	标准时间偏差	描述	备注
东京标准时间	(UTC+09:00)	大坂、札幌、东京	此时区不遵守夏令时。
托木斯克标准时间	(UTC+07:00)	托木斯克	
汤加标准时间	(UTC+13:00)	努库阿洛法	此时区不遵守夏令时。
外贝加尔标准时间	(UTC+09:00)	赤塔市	
土耳其标准时间	(UTC+03:00)	伊斯坦布尔	
特克斯和凯科斯群岛标准时间	(UTC–05:00)	特克斯和凯科斯群岛	
乌兰巴托标准时间	(UTC+08:00)	乌兰巴托	此时区不遵守夏令时。
美国东部标准时间	(UTC–05:00)	印第安纳 (东部)	
美国山地标准时间	(UTC–07:00)	亚利桑那州	此时区不遵守夏令时。
UTC	UTC	协调世界时	此时区不遵守夏令时。
UTC–02	(UTC–02:00)	协调世界时–02	此时区不遵守夏令时。
UTC–08	(UTC–08:00)	协调世界时–08	
UTC–09	(UTC–09:00)	协调世界时–09	
UTC–11	(UTC–11:00)	协调世界时–11	此时区不遵守夏令时。
UTC+12	(UTC+12:00)	协调世界时+12	此时区不遵守夏令时。
UTC+13	(UTC+13:00)	协调世界时+13	

时区	标准时间偏差	描述	备注
委内瑞拉标准时间	(UTC–04:00)	加拉加斯	此时区不遵守夏令时。
符拉迪沃斯托克标准时间	(UTC+10:00)	符拉迪沃斯托克	
伏尔加格勒标准时间	(UTC+04:00)	伏尔加格勒	
澳大利亚西部标准时间	(UTC+08:00)	珀斯	此时区不遵守夏令时。
中非西部标准时间	(UTC+01:00)	中非西部	此时区不遵守夏令时。
西欧标准时间	(UTC+01:00)	阿姆斯特丹、柏林、伯尔尼、罗马、斯德哥尔摩、维也纳	
西蒙古标准时间	(UTC+07:00)	科布多	
西亚标准时间	(UTC+05:00)	阿什哈巴德，塔什干	此时区不遵守夏令时。
西岸标准时间	(UTC+02:00)	加沙，希伯伦	
西太平洋标准时间	(UTC+10:00)	关岛，莫尔兹比港	此时区不遵守夏令时。
雅库茨克标准时间	(UTC+09:00)	雅库茨克	

授予 Amazon RDS 上 Microsoft SQL Server 的许可

在您设置 Microsoft SQL Server 的 Amazon RDS 数据库实例时，已包含软件许可证。

这意味着您无需单独购买 SQL Server 许可证。Amazon拥有 SQL Server 数据库软件的许可证。Amazon RDS 定价包含软件许可证、底层硬件资源以及 Amazon RDS 管理功能。

Amazon RDS 支持以下 Microsoft SQL Server 版本：

- 企业
- 标准
- Web
- Express

> **Note**
>
> SQL Server Web Edition 的许可仅支持公用且可通过 Internet 访问的网页、网站、Web 应用程序和 Web 服务。此级别的支持是符合 Microsoft 的使用权限所必需的。有关更多信息，请参阅 Amazon 服务条款。

Amazon RDS 通过使用 SQL Server 数据库镜像 (DBM) 或 Always On 可用性组 (AG) 支持为运行 Microsoft SQL Server 的数据库实例进行多可用区部署。多可用区部署没有额外的许可要求。有关更多信息，请参阅"Amazon RDS for Microsoft SQL Server 多可用区部署"。

还原许可证终止的数据库实例

Amazon RDS 拍摄许可证终止的数据库实例的快照。如果实例因许可问题而终止，可以从快照将它还原为新的数据库实例。新的数据库实例已包含许可证。

有关更多信息，请参阅"还原 Amazon RDS for SQL Server 的许可证终止的数据库实例"。

开发和测试

由于许可要求，我们无法在 Amazon RDS 上提供 SQL Server Developer 版本。您可以使用 Express 版本来满足很多开发、测试及其他非生产需求。但是，如果您需要 SQL Server 企业级安装的全部功能以用于开发，则可以使用带有 BYOM 的 CEV 在 RDS Custom for SQL Server 上下载并安装 SQL Server Developer 版本。有关更多信息，请参阅 使用自带媒体（BYOM）准备 CEV。Developer 版本

无需专用基础设施。通过使用您自己的主机，您还可以访问在 Amazon RDS 上无法访问的其他可编程性功能。有关 SQL Server 版本之间的差异的更多信息，请参阅 Microsoft 文档中的 SQL Server 2019 的版本和支持的功能。

连接到 Microsoft SQL Server 数据库实例

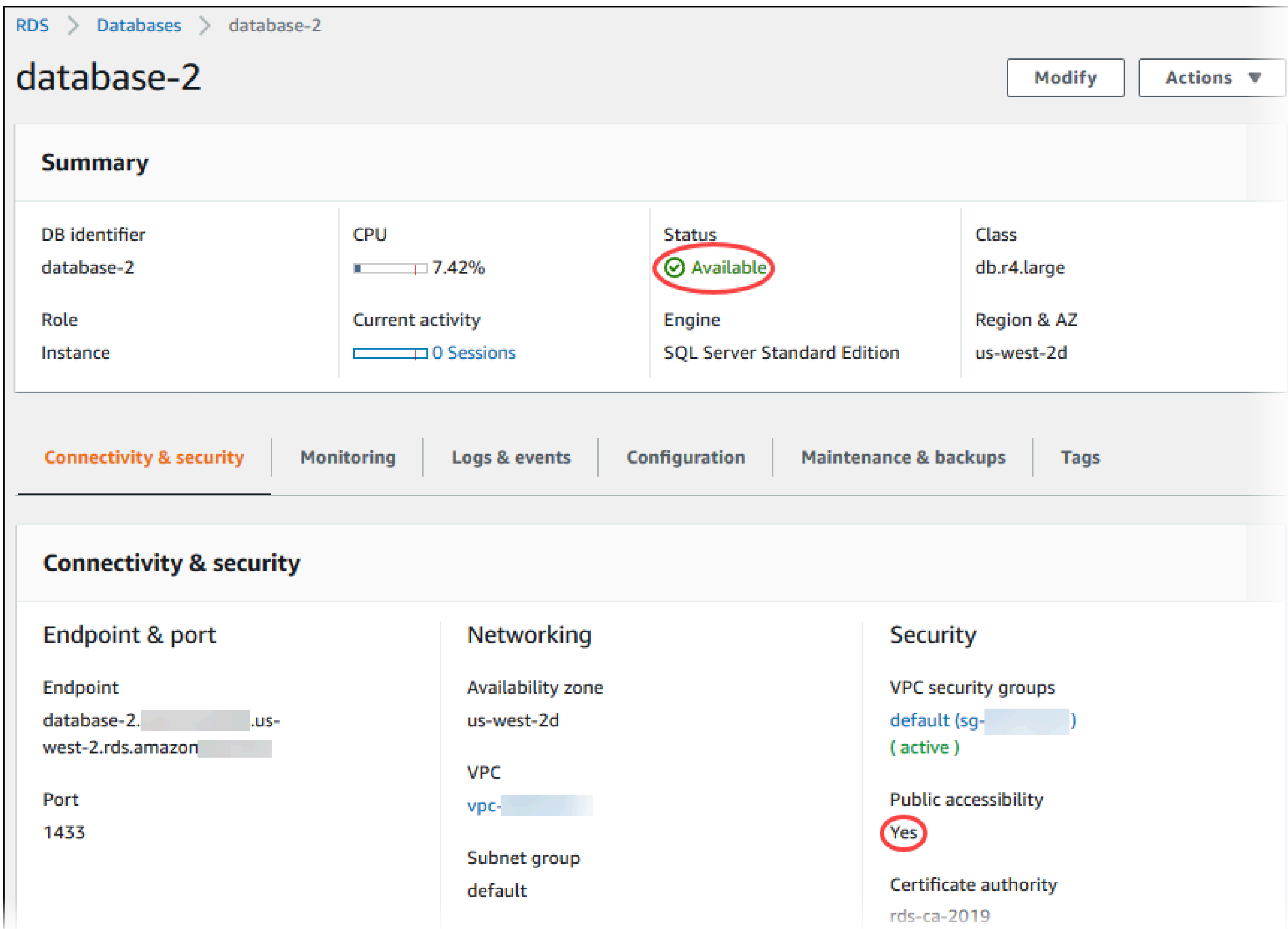
在 Amazon RDS 配置数据库实例后，您可以使用任何标准 SQL 客户端应用程序连接到该数据库实例。在本主题中，您将使用 Microsoft SQL Server Management Studio (SSMS) 或 SQL Workbench/J 连接到数据库实例。

有关指导您完成创建和连接到示例数据库实例的示例，请参阅 创建 Microsoft SQL Server 数据库实例并连接到该实例。

连接之前

在连接到数据库实例之前，它必须是可用且可访问的。

1. 确保其状态为`available`。您可以通过 Amazon Web Services Management Console 或使用 describe-db-instances Amazon CLI 命令查看实例详细信息页面上的相关内容。



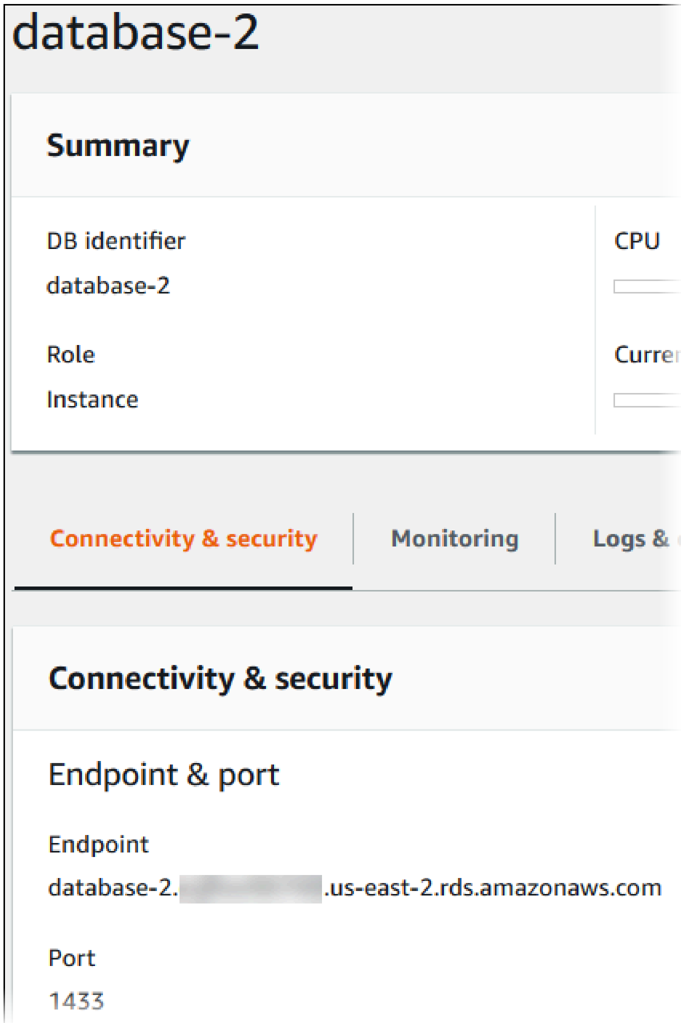
2. 请确保您的源可以访问该实例。根据您的情况，该实例可能不需要能够公开访问。有关更多信息，请参阅 Amazon VPC 和 Amazon RDS。
3. 确保 VPC 安全组的入站规则允许访问您的数据库实例。有关更多信息，请参阅“无法连接到 Amazon RDS 数据库实例”。

查找数据库实例端点和端口号

您需要端点和端口号才能连接到数据库实例。

如需查找端点和端口

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在 Amazon RDS 控制台的右上角，选择数据库实例的Amazon区域。
3. 找到数据库实例的域名系统（DNS）名称（端点）和端口号：
   a. 打开 RDS 控制台，然后选择 Databases (数据库)，以显示数据库实例的列表。
   b. 选择 SQL Server 数据库实例名称以显示其详细信息。
   c. 在连接和安全性选项卡上，复制端点。



d. 请注意端口号。

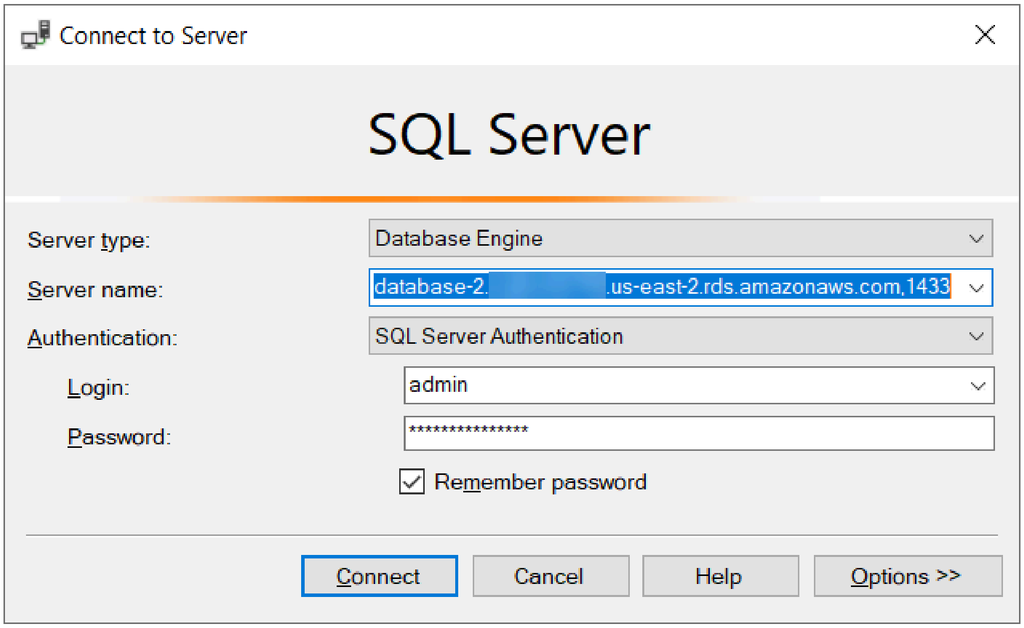
使用 Microsoft SQL Server Management Studio 连接到数据库实例

在此过程中，您将使用 Microsoft SQL Server Management Studio (SSMS) 连接示例数据库实例。要下载此实用工具的独立版本，请参阅 Microsoft 文档了解[下载中心 SQL Server Management Studio (SSMS)]。

使用 SSMS 连接数据库实例

1. 启动 SQL Server Management Studio。

   显示连接至服务器对话框。



2. 为数据库实例提供信息：

   a. 对于 Server type，选择 Database Engine。

   b. 对于 Server name (服务器名称)，输入数据库实例的 DNS 名称（端点）和端口号，并用逗号隔开。

   > **Important**
   >
   > 将端点和端口号之间的冒号更改为逗号。

   您的服务器名称应类似于以下示例：

   ```
   database-2.cg034itsfake.us-east-1.rds.amazonaws.com,1433
   ```

   c. 对于 Authentication，选择 SQL Server Authentication。

   d. 对于 Login (登录名)，输入数据库实例的主用户名。

   e. 对于 Password (密码)，输入数据库实例的密码。

3. 选择 Connect。

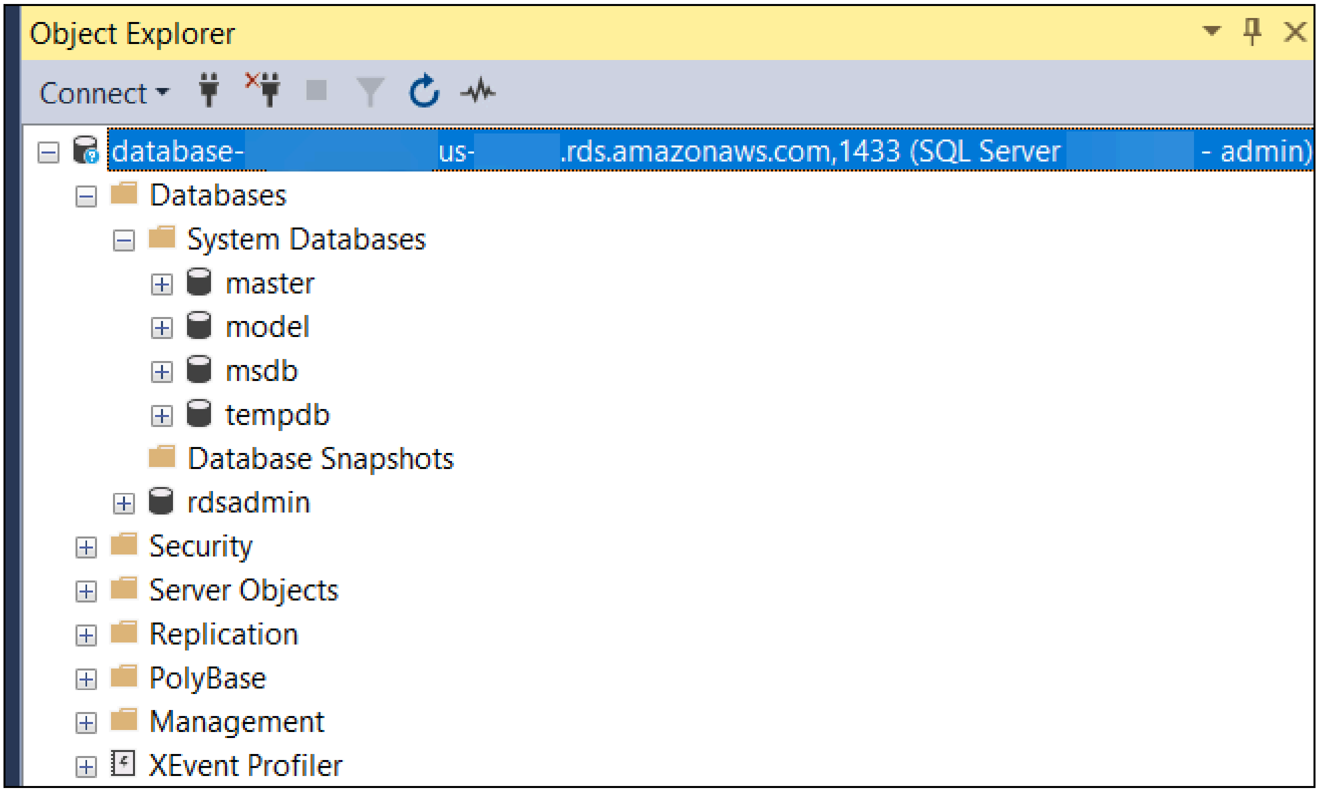
   稍等几分钟后，SSMS 将连接到数据库实例。

   如果您无法连接到数据库实例，请参阅 安全组注意事项 和 排除与 SQL Server 数据库实例的连接故障。

4. 您的 SQL Server 数据库实例附带 SQL Server 的标准内置系统数据库（`master`、`model`、`msdb` 和 `tempdb`）。要了解系统数据库，请执行以下操作：

   a. 在 SSMS 的 View 菜单上，选择 Object Explorer。

   b. 依次展开数据库实例、数据库和系统数据库。



5. 您的 SQL Server 数据库实例还附带一个名为 `rdsadmin` 的数据库。Amazon RDS 使用此数据库存储它用于管理您的数据库的对象。`rdsadmin` 数据库还包括存储过程，您可以运行它们来执行高级任务。有关更多信息，请参阅“Amazon RDS for Microsoft SQL Server 的常见 DBA 任务”。

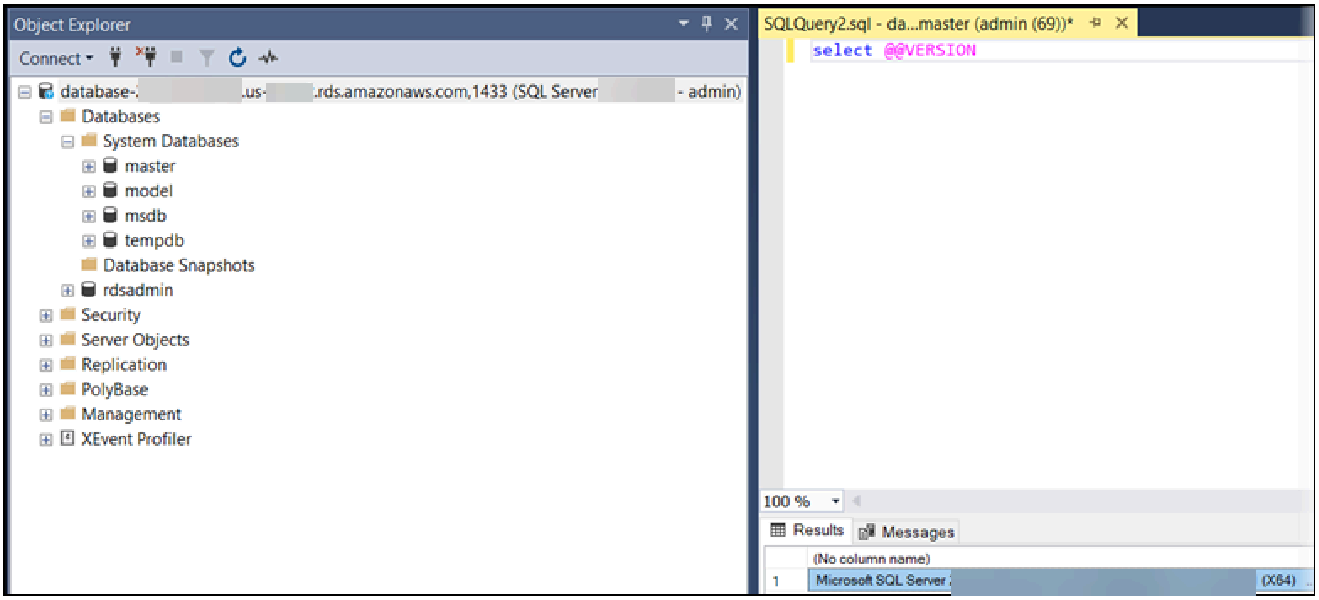
6. 您现在可以开始创建您自己的数据库并像往常一样对数据库实例和数据库运行查询。要对数据库实例运行测试查询，请执行以下操作：

   a. 在 SSMS 中，在 File 菜单上，指向 New，然后选择 Query with Current Connection。

   b. 输入以下 SQL 查询。

   ```
   select @@VERSION
   ```

   c. 运行查询。SSMS 会返回 SQL Server 版的 Amazon RDS 数据库实例。



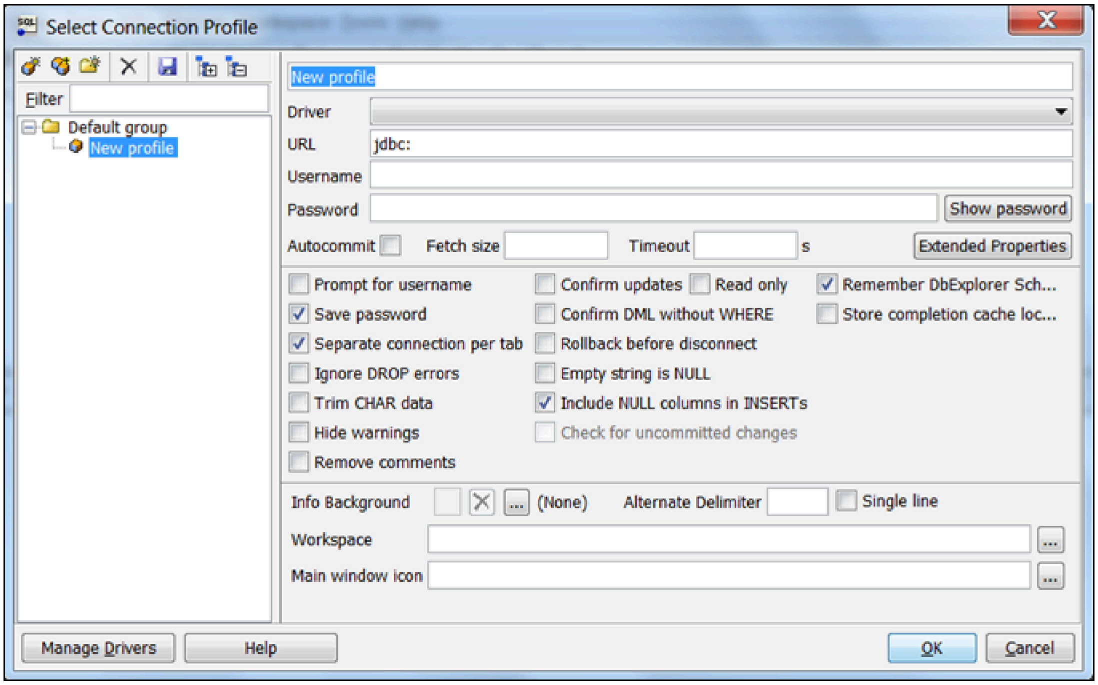
使用 SQL Workbench/J 连接到数据库实例

此示例说明如何使用 SQL Workbench/J 数据库工具连接到运行 Microsoft SQL Server 数据库引擎的数据库实例。要下载 SQL Workbench/J，请参阅 SQL Workbench/J。

SQL Workbench/J 使用 JDBC 连接到数据库实例。您还需要 JDBC driver for SQL Server。要下载此驱动程序，请参阅下载 Microsoft SQL Server JDBC 驱动程序。

使用 SQL Workbench/J 连接到数据库实例

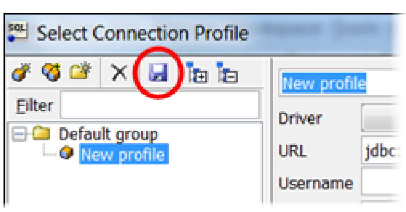
1. 打开 SQL Workbench/J。系统将显示 Select Connection Profile (选择连接配置文件) 对话框，如下所示。



2. 在该对话框顶部的第一个框中，输入配置文件的名称。
3. 对于 Driver (驱动程序)，请选择 **SQL JDBC 4.0**。
4. 对于 URL，输入 **jdbc:sqlserver://**，然后输入您的数据库实例的端点。例如，可以使用以下 URL 值。

```
jdbc:sqlserver://sqlsvr-pdz.abcd12340.us-west-2.rds.amazonaws.com:1433
```

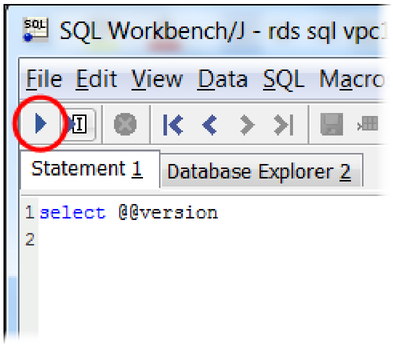
5. 对于 Username (用户名)，输入或粘贴数据库实例的主用户名。
6. 对于 Password (密码)，输入主用户的密码。
7. 选择对话框工具栏中的保存图标，如下所示。



8. 选择 OK。稍等几分钟后，SQL Workbench/J 将连接到数据库实例。如果您无法连接到数据库实例，请参阅 安全组注意事项 和 排除与 SQL Server 数据库实例的连接故障。
9. 在查询窗格中，输入以下 SQL 查询。

```
select @@VERSION
```

10. 选择工具栏中的 Execute 图标，如下所示。



查询将返回数据库实例的版本信息，类似于以下内容。

```
Microsoft SQL Server 2017 (RTM-CU22) (KB4577467) - 14.0.3356.20 (X64)
```

安全组注意事项

要连接到数据库实例，您的数据库实例必须与安全组关联。此安全组包含用于访问数据库实例的 IP 地址和网络配置。您可以在创建数据库实例时，将其与适当的安全组相关联。如果在创建数据库实例时分配了一个默认的未配置安全组，则数据库实例防火墙将阻止连接。

在某些情况下，您可能需要创建新的安全组以实现访问。有关创建新安全组的说明，请参阅 使用安全组控制访问权限。有关指导您完成 VPC 安全组规则设置过程的主题，请参阅教程：创建 VPC 以用于数据库实例（仅限 IPv4）。

在创建新安全组后，需要修改数据库实例以使其与该安全组关联。有关更多信息，请参阅“修改 Amazon RDS 数据库实例”。

可通过使用 SSL 加密与数据库实例的连接来增强安全性。有关更多信息，请参阅 将 SSL 与 Microsoft SQL Server 数据库实例结合使用。

排除与 SQL Server 数据库实例的连接故障

下表显示了尝试连接到 SQL Server 数据库实例时可能会遇到的错误消息。

问题	故障排除建议
Could not open a connection to SQL Server – Microsoft SQL Server,	确保指定了正确的服务器名称。对于 Server name (服务器名称)，输入示例数据库实例的 DNS 名称和端口号，用逗号隔开。

问题	故障排除建议
Error: 53 (无法建立与 SQL Server 的连接 - Microsoft SQL Server，错误：53)	**Important** 如果 DNS 名称和端口号之间有冒号，请将冒号更改为逗号。 您的服务器名称应类似于以下示例： `sample-instance.cg034itsfake.us-east-1.rds.amazonaws.com,1433`
No connection could be made because the target machine actively refused it – Microsoft SQL Server, Error: 10061 (无法连接，因为目标计算机有效拒绝连接 - Microsoft SQL Server，错误：10061)	您可以访问数据库实例，但连接已被拒绝。此问题通常是因为指定的用户名或密码不正确。请验证用户名和密码，然后重试。
A network-related or instance-specific error occurred while establishing a connection to SQL Server. The server was not found or was not accessible... The wait operation timed out – Microsoft SQL Server, Error: 258 (与 SQL Server 建立连接时发生网络相关的错误或特定于实例的错误。未找到服务器或无法访问服务器... 等待操作超时 - Microsoft SQL Server，错误：258)	由本地防火墙强制执行的访问规则与您授权用于访问数据库实例的 IP 地址可能不匹配。该问题很可能是由安全组中的入站规则所致。有关更多信息，请参阅 Amazon RDS 中的安全性。
	您的数据库实例必须可公开访问。要从 VPC 外部连接到该实例，必须向该实例分配一个公有 IP 地址。

> **Note**
>
> 有关连接问题的更多信息，请参阅无法连接到 Amazon RDS 数据库实例。

将 Active Directory 用于 RDS for SQL Server

您可以将 RDS for SQL Server 数据库实例加入 Microsoft Active Directory（AD）域。您的 AD 域可以托管在 Amazon 内的 Amazon Managed AD 上，也可以托管在您所选位置（包括您的企业数据中心、Amazon EC2 或其他云提供商）的自托管式 AD 上。

您可以结合使用 NTLM 身份验证和自托管式 Active Directory 来对域用户进行身份验证。您可以将 Kerberos 和 NTLM 身份验证与 Amazon Managed Active Directory 结合使用。

在以下各节中，您可以找到有关将自托管式 Active Directory 和 Amazon Managed Active Directory 用于 Amazon RDS 上的 Microsoft SQL Server 的信息。

主题

将自托管式 Active Directory 用于 Amazon RDS for SQL Server 数据库实例

无论您的 AD 托管在企业数据中心、Amazon EC2 上还是其他云提供商处，您都可以将 RDS for SQL Server 数据库实例直接加入到您的自托管式 Active Directory（AD）域。借助自托管式 AD，您可以使用 NTLM 身份验证直接控制 RDS for SQL Server 数据库实例上的用户和服务的身份验证，而无需使用中间域和林信任。当用户在加入您自托管式 AD 域的 RDS for SQL Server 数据库实例上进行身份验证时，身份验证请求将会转发到您指定的自托管式 AD 域中。

主题

- 区域和版本可用性
- 要求
- 限制
- 设置自托管式 Active Directory 概述
- 设置自托管式 Active Directory
- 在自托管式 Active Directory 域中管理数据库实例
- 了解自托管式 Active Directory 域成员资格
- 自托管式 Active Directory 问题排查
- 还原 SQL Server 数据库实例，然后将其添加到自托管式 Active Directory 域

区域和版本可用性

Amazon RDS 在所有 Amazon Web Services 区域都支持通过自托管式 AD 使用 NTLM 控制 SQL Server 的身份验证。

要求

在将 RDS for SQL Server 数据库实例加入自托管式 AD 域之前，请确保您已满足以下要求。

主题

- 配置您的本地 AD
- 配置您的网络连接
- 配置您的 AD 域服务账户

配置您的本地 AD

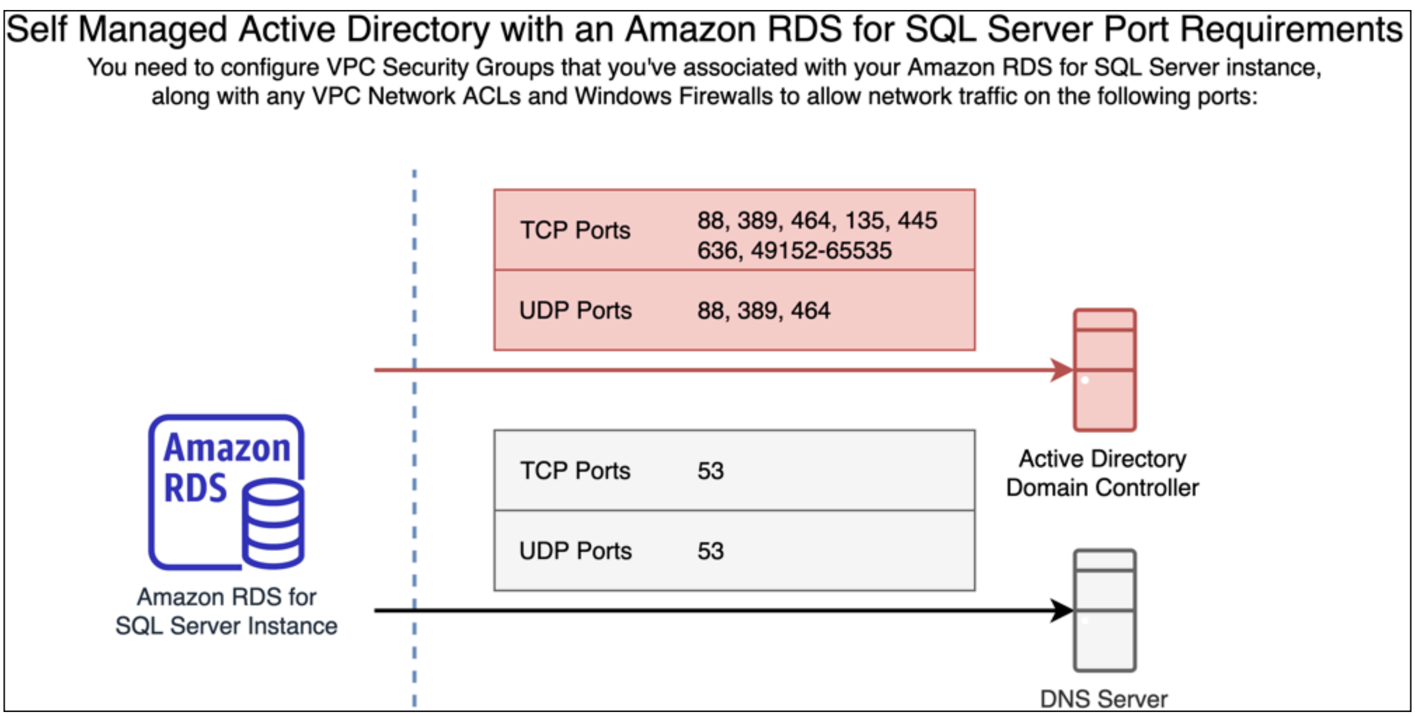
确保您有一个本地或其他自托管式 Microsoft AD，您可以在其中加入 Amazon RDS for SQL Server 实例。您的本地 AD 应具有以下配置：

- 如果您定义了 Active Directory 站点，请确保在 Active Directory 站点中定义与 RDS for SQL Server 数据库实例关联的 VPC 中的子网。确认您的 VPC 中的子网与您其他 AD 站点中的子网之间没有任何冲突。
- 您 AD 域控制器的域功能级别为 Windows Server 2008 R2 或更高版本。
- 您的 AD 域名不能采用单标签域（SLD）格式。RDS for SQL Server 不支持 SLD 域。
- AD 的完全限定域名（FQDN）不能超过 47 个字符。

配置您的网络连接

确保您满足以下网络配置：

- 在您要在其中创建 RDS for SQL Server 数据库实例的 Amazon VPC 与自托管式 Active Directory 之间配置了连接。您可以使用 Amazon Direct Connect、Amazon VPN、VPC 对等或 Amazon Transit Gateway 设置连接。
- 对于 VPC 安全组，原定设置 Amazon VPC 的原定设置安全组已添加到控制台中的 RDS for SQL Server 数据库实例。确保要在其中创建 RDS for SQL Server 数据库实例的子网的安全组和 VPC 网络 ACL 在端口上允许有下图所示方向的流量。



下表确定了每个端口的作用。

协议	端口	角色
TCP/UDP	53	域名系统（DNS）
TCP/UDP	88	Kerberos 身份验证
TCP/UDP	464	更改/设置密码
TCP/UDP	389	轻型目录访问协议（LDAP）
TCP	135	分布式计算环境/端点映射器（DCE/EPMAP）
TCP	445	目录服务 SMB 文件共享
TCP	636	基于 TLS/SSL 的轻型目录访问协议（LDAPS）
TCP	49152 - 65535	RPC 的临时端口

- 通常，域 DNS 服务器位于 AD 域控制器中。您无需配置 VPC DHCP 选项集即可使用此功能。有关更多信息，请参阅《Amazon VPC 用户指南》中的 DHCP 选项集。

Important

如果您使用的是 VPC 网络 ACL，则还必须允许动态端口（49152-65535）上有来自 RDS for SQL Server 数据库实例的出站流量。确保这些流量规则也镜像到适用于每个 AD 域控制器、DNS 服务器和 RDS for SQL Server 数据库实例的防火墙上。

虽然 VPC 安全组要求仅在发起网络流量的方向打开端口，但大多数 Windows 防火墙和 VPC 网络 ACL 要求双向打开端口。

配置您的 AD 域服务账户

确保您的 AD 域服务账户满足以下要求：

- 确保您在自托管式 AD 域中有一个服务账户，该账户具有将计算机加入该域的委派权限。域服务账户是您的自托管式 AD 中的一个用户账户，该账户已被委派执行某些任务的权限。
- 需要在您要加入 RDS for SQL Server 数据库实例的组织单位（OU）中向域服务账户委派以下权限：
  - 验证写入 DNS 主机名的能力
  - 验证写入服务主体名称的能力
  - 创建和删除计算机对象

  这些权限代表将计算机对象加入到您自托管式 Active Directory 至少需要具备的一组权限。有关更多信息，请参阅 Microsoft Windows Server 文档中的尝试将计算机加入域时出现错误。

> **Important**
>
> 创建数据库实例后，请勿移动 RDS for SQL Server 在组织单位中创建的计算机对象。移动关联对象将导致您的 RDS for SQL Server 数据库实例出现配置错误。如果您需要移动 Amazon RDS 创建的计算机对象，请通过 ModifyDBInstance RDS API 操作用计算机对象所需位置修改域参数。

限制

以下限制适用于面向 SQL Server 的自托管式 AD。

- NTLM 是唯一支持的身份验证类型。不支持 Kerberos 身份验证。如果您需要使用 kerberos 身份验证，则可以使用 Amazon Managed AD，而不是自托管式 AD。
- 不支持 Microsoft 分布式事务协调器（MSDTC）服务，因为此服务需要 Kerberos 身份验证。
- 您的 RDS for SQL Server 数据库实例不使用自托管式 AD 域的网络时间协议（NTP）服务器。而是使用 Amazon NTP 服务。
- SQL Server 链接服务器必须使用 SQL 身份验证连接加入自托管式 AD 域的其他 RDS for SQL Server 数据库实例。
- 来自自托管式 AD 域的 Microsoft 组策略对象（GPO）设置不适用于 RDS for SQL Server 数据库实例。

设置自托管式 Active Directory 概述

要为 RDS for SQL Server 数据库实例设置自托管式 AD，请执行以下步骤，详情请参阅设置自托管式 Active Directory：

在您的 AD 域中：

- 创建组织单位（OU)。
- 创建 AD 域用户。
- 将控制权委派给 AD 域用户。

从 Amazon Web Services Management Console或 API：

- 创建 Amazon KMS 密钥。
- 使用 Amazon Secrets Manager 创建密钥。
- 创建或修改 RDS for SQL Server 数据库实例，然后将其加入自托管式 AD 域。

设置自托管式 Active Directory

要设置自托管式 AD，请执行以下步骤。

主题

步骤 1：在 AD 中创建组织单位

> **Important**
>
> 我们建议为任何拥有已加入您自托管式 AD 域的 RDS for SQL Server 数据库实例的 Amazon 账户创建专用 OU 以及作用域为该 OU 的服务凭证。通过专用 OU 和服务凭证，您可以避免权限冲突并遵循最低权限原则。

在 AD 中创建 OU

1. 以域管理员身份连接到您的 AD 域。
2. 打开 Active Directory 用户和计算机，然后选择要在其中创建 OU 的域。
3. 右键单击该域并选择新建，然后选择组织单位。
4. 为 OU 输入名称。
5. 选中保护容器免遭意外删除复选框。
6. 单击确定。您的新 OU 将出现在您的域下。

步骤 2：在您的 AD 中创建 AD 域用户

域用户凭证将用于 Amazon Secrets Manager 中的密钥。

在您的 AD 中创建 AD 域用户

1. 打开 Active Directory 用户和计算机，然后选择要在其中创建用户的域和 OU。
2. 右键单击用户对象并选择新建，然后选择用户。
3. 输入用户的名字、姓氏和登录名。单击下一步。
4. 输入用户的密码。请勿选择“用户下次登录时必须更改密码”。请勿选择“账户已禁用”。单击下一步。
5. 单击确定。您的新用户将出现在您的域下。

步骤 3：将控制权委派给 AD 用户

将控制权委派给域中的 AD 域用户

1. 打开 Active Directory 用户和计算机 MMC 管理单元，然后选择要在其中创建用户的域。

2. 右键单击您之前创建的 OU，然后选择委派控制权。
3. 在委派控制权向导页面上，单击下一步。
4. 在用户或组部分，单击添加。
5. 在选择用户、计算机或组部分，输入您创建的 AD 用户，然后单击检查名称。如果您的 AD 用户检查成功，请单击确定。
6. 在用户或组部分，确认您的 AD 用户已添加，然后单击下一步。
7. 在要委派的任务部分，选择创建要委派的自定义任务，然后选择下一步。
8. 在 Active Directory 对象类型部分：

    a. 选择仅文件夹中的以下对象。
    b. 选择计算机对象。
    c. 选择在此文件夹中创建选定对象。
    d. 选择删除此文件夹中的选定对象，然后单击下一步。
9. 在权限部分：

    a. 将常规保持为选中状态。
    b. 选择已验证写入 DNS 主机名。
    c. 选择已验证写入服务主体名称，然后单击下一步。
10. 对于完成控制权委派向导，请检查并确认您的设置，然后单击完成。

步骤 4：创建 Amazon KMS 密钥

KMS 密钥用于加密您的 Amazon 密钥。

要创建 Amazon KMS 密钥

> **Note**
>
> 对于加密密钥，请勿使用 Amazon 原定设置 KMS 密钥。确保在包含要加入自托管式 AD 的 RDS for SQL Server 数据库实例的 Amazon 账户中创建 Amazon KMS 密钥。

1. 在 Amazon KMS 控制台中，选择创建密钥。
2. 对于密钥类型，选择对称。
3. 对于密钥用法，选择加密和解密。

4. 对于 Advanced options (高级选项)：

   a. 对于密钥材料源，选择 KMS。
   b. 对于区域性，选择单区域密钥，然后单击下一步。
5. 对于别名，提供 KMS 密钥的名称。
6. (可选）对于描述，提供 KMS 密钥的描述。
7. （可选）对于标签，提供 KMS 密钥的标签，然后单击下一步。
8. 对于密钥管理员，提供 IAM 用户的名称并将其选中。
9. 对于密钥删除，确保选中允许密钥管理员删除此密钥复选框，然后单击下一步。
10. 对于密钥用户，请提供上一步中的 IAM 用户并将其选中。单击下一步。
11. 审核配置。
12. 对于密钥策略，请在策略声明中包括以下内容：

```
{
    "Sid": "Allow use of the KMS key on behalf of RDS",
    "Effect": "Allow",
    "Principal": {
        "Service": [
            "rds.amazonaws.com"
        ]
    },
    "Action": "kms:Decrypt",
    "Resource": "*"
}
```
13. 单击 Finish (完成)。

步骤 5：创建 Amazon 密钥

创建密钥

> **Note**
> 确保在包含要加入自托管式 AD 的 RDS for SQL Server 数据库实例的 Amazon 账户中创建密钥。

1. 在 Amazon Secrets Manager 下，选择存储新密钥。

2. 对于密钥类型，请选择其他密钥类型。
3. 对于键/值对，请添加您的两个密钥：

    a. 对于第一个密钥，请输入 CUSTOMER_MANAGED_ACTIVE_DIRECTORY_USERNAME。
    b. 对于第一个密钥的值，请仅输入 AD 用户的用户名（不带域前缀）。请勿包含域名，因为这会导致实例创建失败。
    c. 对于第二个密钥，请输入 CUSTOMER_MANAGED_ACTIVE_DIRECTORY_PASSWORD。
    d. 对于第二个密钥的值，请输入您在域中为 AD 用户创建的密码。
4. 对于加密密钥，输入您在上一步中创建的 KMS 密钥，然后单击下一步。
5. 在密钥名称中，输入一个描述性名称，以便您稍后查找自己的密钥。
6. （可选）对于描述，输入密钥名称的描述。
7. 对于资源权限，请单击编辑。
8. 将以下策略添加到权限策略中：

> **Note**
>
> 我们建议您在策略中使用 aws:sourceAccount 和 aws:sourceArn 条件，以避免出现混淆代理人问题。将您的 Amazon Web Services 账户用于 aws:sourceAccount，并将 RDS for SQL Server 数据库实例 ARN 用于 aws:sourceArn。有关更多信息，请参阅[防范跨服务混淆代理问题](#)。

```
{
    "Version": "2012-10-17",
    "Statement":
    [
        {
            "Effect": "Allow",
            "Principal":
            {
                "Service": "rds.amazonaws.com"
            },
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "*",
            "Condition":
            {
                "StringEquals":
                {
```

```
                        "aws:sourceAccount": "123456789012"
                    },
                    "ArnLike":
                    {
                        "aws:sourceArn": "arn:aws:rds:us-west-2:123456789012:db:*"
                    }
                }
            }
        ]
}
```

9. 单击保存，然后单击下一步。
10. 对于配置轮换设置，保留原定设置值并选择下一步。
11. 查看密钥的设置，然后单击存储。
12. 选择您创建的密钥，然后复制密钥 ARN 的值。下一步将使用它来设置自托管式 Active Directory。

步骤 6：创建或修改 SQL Server 数据库实例

您可以使用控制台、CLI 或 RDS API 将 RDS for SQL Server 数据库实例与自托管式 AD 域关联。您可以通过下列方式之一来执行该操作：

- 使用控制台、create-db-instance CLI 命令或 CreateDBInstance RDS API 操作创建新的 SQL Server 数据库实例。

  有关说明，请参阅创建 Amazon RDS 数据库实例。
- 使用控制台、modify-db-instance CLI 命令或 ModifyDBInstance RDS API 操作修改现有 SQL Server 数据库实例。

  有关说明，请参阅修改 Amazon RDS 数据库实例。
- 使用控制台、restore-db-instance-from-db-snapshot CLI 命令或 RestoreDBInstanceFromDBSnapshot RDS API 操作，从数据库快照还原 SQL Server 数据库实例。

  有关说明，请参阅还原到数据库实例。
- 使用控制台、restore-db-instance-to-point-in-time CLI 命令或 RestoreDBInstanceToPointInTime RDS API 操作，将 SQL Server 数据库实例还原到某个时间点。

  有关说明，请参阅将 Amazon RDS 的数据库实例还原到指定时间。

使用 Amazon CLI 时，数据库实例需要以下参数才能使用您创建的自托管式 Active Directory 域：

- 对于 `--domain-fqdn` 参数，请使用自托管式 Active Directory 的完全限定域名（FQDN)。
- 对于 `--domain-ou` 参数，请使用您在自托管式 AD 中创建的 OU。
- 对于 `--domain-auth-secret-arn` 参数，使用您在上一步中创建的密钥 ARN 的值。
- 对于 `--domain-dns-ips` 参数，请使用自托管式 AD 的 DNS 服务器的主要 IPv4 地址和辅助 IPv4 地址。如果您没有辅助 DNS 服务器 IP 地址，请输入主 IP 地址两次。

以下示例 CLI 命令显示了如何创建、修改和删除已加入自托管式 AD 域的 RDS for SQL Server 数据库实例。

> **Important**
>
> 如果您修改数据库实例，将其加入自托管式 AD 域或将其从自托管式 AD 域中删除，则需要重启数据库实例才能使修改生效。您可以选择立即应用更改，也可以等到下一个维护时段。选择立即应用选项将导致单可用区数据库实例停机。多可用区数据库实例将在完成重启之前执行失效转移。有关更多信息，请参阅 使用计划修改设置。

以下 CLI 命令创建一个新的 RDS for SQL Server 数据库实例并将其加入自托管式 AD 域。

对于 Linux、macOS 或 Unix：

```
aws rds create-db-instance \
    --db-instance-identifier my-DB-instance \
    --db-instance-class db.m5.xlarge \
    --allocated-storage 50 \
    --engine sqlserver-se \
    --engine-version 15.00.4043.16.v1 \
    --license-model license-included \
    --master-username my-master-username \
    --master-user-password my-master-password \
    --domain-fqdn my_AD_domain.my_AD.my_domain \
    --domain-ou OU=my-AD-test-OU,DC=my-AD-test,DC=my-AD,DC=my-domain \
    --domain-auth-secret-arn "arn:aws:secretsmanager:region:account-number:secret:my-
AD-test-secret-123456" \
    --domain-dns-ips "10.11.12.13" "10.11.12.14"
```

对于 Windows：

```
aws rds create-db-instance ^
```

```
    --db-instance-identifier my-DB-instance ^
    --db-instance-class db.m5.xlarge ^
    --allocated-storage 50 ^
    --engine sqlserver-se ^
    --engine-version 15.00.4043.16.v1 ^
    --license-model license-included ^
    --master-username my-master-username ^
    --master-user-password my-master-password ^
    --domain-fqdn my-AD-test.my-AD.mydomain ^
    --domain-ou OU=my-AD-test-OU,DC=my-AD-test,DC=my-AD,DC=my-domain ^
    --domain-auth-secret-arn "arn:aws:secretsmanager:region:account-number:secret:my-
AD-test-secret-123456" \ ^
    --domain-dns-ips "10.11.12.13" "10.11.12.14"
```

以下 CLI 命令将现有 RDS for SQL Server 数据库实例修改为使用自托管式 Active Directory 域。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier my-DB-instance \
    --domain-fqdn my_AD_domain.my_AD.my_domain \
    --domain-ou OU=my-AD-test-OU,DC=my-AD-test,DC=my-AD,DC=my-domain \
    --domain-auth-secret-arn "arn:aws:secretsmanager:region:account-number:secret:my-
AD-test-secret-123456" \
    --domain-dns-ips "10.11.12.13" "10.11.12.14"
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier my-DBinstance ^
    --domain-fqdn my_AD_domain.my_AD.my_domain ^
    --domain-ou OU=my-AD-test-OU,DC=my-AD-test,DC=my-AD,DC=my-domain ^
    --domain-auth-secret-arn "arn:aws:secretsmanager:region:account-number:secret:my-
AD-test-secret-123456" ^
    --domain-dns-ips "10.11.12.13" "10.11.12.14"
```

以下 CLI 命令从自托管式 Active Directory 域中删除 RDS for SQL Server 数据库实例。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier my-DB-instance \
```

```
    --disable-domain
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier my-DB-instance ^
    --disable-domain
```

步骤 7：创建 Windows 身份验证 SQL Server 登录名

使用 Amazon RDS 主用户凭证连接 SQL Server 数据库实例，就像连接任何其他数据库实例一样。由于数据库实例已加入自托管式 AD 域，因此您可以预调配 SQL Server 登录名和用户。您可以通过自托管式 AD 域中的 AD 用户和组实用程序执行此操作。通过对这些 Windows 登录名授予和撤销标准 SQL Server 权限来管理数据库权限。

为了使自托管式 AD 用户能够在 SQL Server 上进行身份验证，自托管式 AD 用户或该用户所属的自托管式 Active Directory 组必须存在 SQL Server Windows 登录名。访问权限的精细控制是通过对这些 SQL Server 登录名授予和撤销权限来处理的。自托管式 AD 用户没有 SQL Server 登录名，或者用户所属自托管式 AD 组中的登录名无法访问 SQL Server 数据库实例。

创建自托管式 AD SQL Server 登录名需要 ALTER ANY LOGIN 权限。如果您尚未使用此权限创建任何登录名，请使用 SQL Server 身份验证以数据库实例的主用户身份进行连接，并在主用户的上下文下创建自托管式 AD SQL Server 登录名。

您可以运行数据定义语言（DDL）命令（如以下示例），为自托管式 AD 用户或组创建 SQL Server 登录名。

> **Note**
>
> 使用 Windows 2000 之前的登录名称指定用户和组，格式为 *my_AD_domain**my_AD_domain_user*。您不能使用 *my_AD_domain_user*@*my_AD_domain* 格式的用户主体名 (UPN)。

```
USE [master]
GO
CREATE LOGIN [my_AD_domain\my_AD_domain_user] FROM WINDOWS WITH DEFAULT_DATABASE =
 [master], DEFAULT_LANGUAGE = [us_english];
GO
```

有关更多信息，请参阅 Microsoft 开发人员网络文档中的 CREATE LOGIN (Transact-SQL)。

域中用户（人和应用程序）现在均可从加入自托管式 AD 域的客户端计算机使用 Windows 身份验证连接到 RDS for SQL Server 实例。

在自托管式 Active Directory 域中管理数据库实例

您可以使用控制台、Amazon CLI 或 Amazon RDS API 来管理数据库实例及其与自托管式 AD 域的关系。例如，您可以将数据库实例移入域、移出域或在域之间移动。

例如，使用 Amazon RDS API，您可以执行下列操作：

- 要重新尝试以失败的成员身份加入自托管式域，请使用 ModifyDBInstance API 操作并指定相同的参数集：
  - `--domain-fqdn`
  - `--domain-dns-ips`
  - `--domain-ou`
  - `--domain-auth-secret-arn`
- 要从自托管式域中删除数据库实例，请使用 `ModifyDBInstance` API 操作并指定 `--disable-domain` 作为域参数。
- 要将数据库实例从一个自托管式域移至另一个自托管式域，请使用 `ModifyDBInstance` API 操作并指定新域的域参数。
  - `--domain-fqdn`
  - `--domain-dns-ips`
  - `--domain-ou`
  - `--domain-auth-secret-arn`
- 要列出每个数据库实例的自托管式 AD 域成员资格，请使用 DescribeDBInstances API 操作。

了解自托管式 Active Directory 域成员资格

在创建或修改数据库实例后，实例将成为自托管式 AD 域的成员。Amazon 控制台将指示数据库实例的自托管式 Active Directory 域成员资格的状态。数据库实例的状态可以是以下状态之一：

- 已加入 – 实例是 AD 域的成员。
- 正在加入 – 实例正处于成为 AD 域成员的过程中。
- 待联接 – 实例成员资格待定。

- pending-maintenance-join – Amazon 将在下一计划维护时段期间尝试使实例成为 AD 域成员。
- 待删除 – 等待从 AD 域中删除实例。
- pending-maintenance-removal – Amazon将在下一计划维护时段期间尝试从 AD 域中删除实例。
- 失败 – 配置问题阻碍实例加入 AD 域。在重新发出实例修改命令之前检查并修复配置。
- 正在删除 – 正从自托管式 AD 域中删除实例。

成为自托管式 AD 域成员的请求可能因网络连接问题而失败。例如，您可能会创建数据库实例或修改现有实例，并且尝试使数据库实例成为某个自托管式 AD 域的成员会失败。在这种情况下，您重新发出命令来创建或修改数据库实例，或者修改新创建的实例来加入自托管式 AD 域。

自托管式 Active Directory 问题排查

以下是您在设置或修改自托管式 AD 时可能遇到的问题。

错误代码	描述	常见原因	故障排除建议
错误 2/0x2	系统找不到指定的文件。	使用 `–domain-ou` 参数指定的组织单位（OU）的格式或位置无效。通过 Amazon Secrets Manager 指定的域服务账户缺少加入 OU 所需的权限。	查看 `–domain-ou` 参数。确保域服务账户拥有相应的 OU 权限。有关更多信息，请参阅 配置您的 AD 域服务账户。
错误 5/0x5	访问被拒绝。	域服务账户的权限配置不正确，或者域中已经存在该计算机账户。	查看域中的域服务账户权限，并验证 RDS 计算机账户在域中是否重复。您可以通过在 RDS for SQL Server 数据库实例上运行 `SELECT @@SERVERNAME` 来验证 RDS 计算机账户的名称。如果您使用的是多可用区，请尝试通过失效转移重启，然后再次验证 RDS 计算机账户。有关更多信息，请参阅 重启中的数据库实例。

错误代码	描述	常见原因	故障排除建议
错误 87/0x57	参数不正确。	通过 Amazon Secrets Manager 指定的域服务账户不具备相应的权限。用户配置文件也可能已损坏。	查看域服务账户的要求。有关更多信息，请参阅 配置您的 AD 域服务账户。
错误 234/0xEA	指定的组织单位（OU）不存在。	使用 –domain-ou 参数指定的 OU 在您的自托管式 AD 中不存在。	查看 –domain-ou 参数并确保指定的 OU 在您的自托管式 AD 中存在。
错误 1326/0x52E	用户名或密码不正确。	Amazon Secrets Manager 中提供的域服务账户凭证包含未知的用户名或错误的密码。您的自托管式 AD 中也可能已禁用域账户。	确保 Amazon Secrets Manager 中提供的凭证正确无误，并且在您的自托管式 Active Directory 中启用了域账户。
错误 1355/0x54B	指定的域不存在或无法访问。	域已关闭、指定的 DNS IP 集无法访问，或指定的 FQDN 无法访问。	查看 –domain-dns-ips 和 –domain-fqdn 参数，确保正确。查看您的 RDS for SQL Server 数据库实例的网络配置，并确保您的自托管式 AD 可以访问。有关更多信息，请参阅 配置您的网络连接。
错误 1722/0x6BA	RPC 服务器不可用。	连接您的 AD 域的 RPC 服务时出现问题。这可能由于服务或网络问题导致。	验证 RPC 服务是否正在您的域控制器上运行，以及是否可以在您的域上从 RDS for SQL Server 数据库实例访问 TCP 端口 135 和 49152-65535 。

错误代码	描述	常见原因	故障排除建议
错误 2224/0x8b0	用户账户已存在。	正在尝试添加到您自托管式 AD 的计算机账户已经存在。	通过在您的 RDS for SQL Server 数据库实例上运行 `SELECT @@SERVERNAME` 来识别计算机账户，然后小心地将其从您的自托管式 AD 中删除。
错误 2242/0x8c2	此用户的密码已过期。	通过 Amazon Secrets Manager 指定的域服务账户的密码已过期。	更新用于将 RDS for SQL Server 数据库实例加入自托管式 AD 的域服务账户的密码。

还原 SQL Server 数据库实例，然后将其添加到自托管式 Active Directory 域

您可以还原数据库快照或对 SQL Server 数据库实例执行时间点恢复（PITR），然后将其添加到自托管式 Active Directory 域。还原数据库实例后，使用步骤 6：创建或修改 SQL Server 数据库实例中介绍的过程修改此实例，以将数据库实例添加到自托管式 AD 域。

将 Amazon Managed Active Directory 用于 RDS for SQL Server

当用户连接您的 RDS for SQL Server 数据库实例时，您可以使用 Amazon Managed Microsoft AD 通过 Windows 身份验证对用户进行身份验证。数据库实例使用 Amazon Directory Service for Microsoft Active Directory（也称为 Amazon Managed Microsoft AD）来启用 Windows 身份验证。当用户对联接到信任域的 SQL Server 数据库实例进行身份验证时，身份验证请求将转发至您使用 Amazon Directory Service 创建的域目录。

区域和版本可用性

Amazon RDS 支持仅使用 Amazon Managed Microsoft AD 进行 Windows 身份验证。RDS 不支持使用 AD Connector。有关更多信息，请参阅下列内容：

- Amazon Managed Microsoft AD 的应用程序兼容性策略
- AD Connector 的应用程序兼容性策略

有关版本和区域可用性的信息，请参阅 RDS for SQL Server 的 Kerberos 身份验证。

设置 Windows 身份验证概述

Amazon RDS 使用混合模式进行 Windows 身份验证。此方式意味着，主用户（用于创建 SQL Server 数据库实例的名称和密码）使用 SQL 身份验证。由于主用户账户是特权凭证，您应限制对此账户的访问。

要使用本地或自托管式 Microsoft Active Directory 获取 Windows 身份验证，请创建林信任。信任可以是单向或双向的。有关使用 Amazon Directory Service 设置林信任的更多信息，请参阅 Amazon Directory Service 管理指南 中的何时创建信任关系。

要为 SQL Server 数据库实例设置 Windows 身份验证，请执行下列步骤（为 SQL Server 数据库实例设置 Windows 身份验证 中更为详细地介绍）：

1. 从 Amazon Managed Microsoft AD或 Amazon Web Services Management Console API 使用 Amazon Directory Service 创建 Amazon Managed Microsoft AD 目录。
2. 如果使用 Amazon CLI 或 Amazon RDS API 创建 SQL Server 数据库实例，请创建 Amazon Identity and Access Management（IAM）角色。此角色使用托管 IAM 策略 `AmazonRDSDirectoryServiceAccess` 并允许 Amazon RDS 调用您的目录。如果您使用控制台创建 SQL Server 数据库实例，则 Amazon 将为您创建 IAM 角色。

为了让角色允许访问，Amazon Security Token Service (Amazon STS) 终端节点必须在您的 Amazon 账户的 Amazon 区域中激活。Amazon STS 终端节点默认在所有 Amazon 区域中保持活跃，且您无需任何进一步动作即可使用它们。有关更多信息，请参阅《IAM 用户指南》中的在 Amazon Web Services 区域中管理 Amazon STS。

3. 使用 Microsoft Active Directory 工具在 Amazon Managed Microsoft AD 目录中创建和配置用户与组。有关在 Microsoft Active Directory 中创建用户的更多信息，请参阅 Amazon Directory Service管理指南中的管理 Amazon Managed Microsoft AD 中的用户和组。
4. 如果您计划在不同 VPC 中查找目录和数据库实例，请启用跨 VPC 流量。
5. 使用 Amazon RDS 从控制台、Amazon CLI 或 Amazon RDS API 新建 SQL Server 数据库实例。在创建请求中，提供在创建目录时生成的域标识符（“d-*”标识符）和您创建的角色的名称。通过为数据库实例设置域和 IAM 角色参数，您还可将现有 SQL Server 数据库实例修改为使用 Windows 身份验证。
6. 使用 Amazon RDS 主用户凭证连接到 SQL Server 数据库实例，就像连接到任何其他数据库实例一样。由于数据库实例已联接到 Amazon Managed Microsoft AD 域，您可从域中的 Active Directory 用户和组中预配置 SQL Server 登录名和用户。（这称为 SQL Server“Windows”登录。）通过对这些 Windows 登录名授予和撤销标准 SQL Server 权限来管理数据库权限。

为 Kerberos 身份验证创建终端节点

基于 Kerberos 的身份验证需要终端节点是客户指定的主机名、句点，然后是完全限定的域名 (FQDN)。例如，以下是您可以用于基于 Kerberos 身份验证的终端节点的示例。在此示例中，SQL Server 数据库实例主机名为 `ad-test`，域名为 `corp-ad.company.com`。

```
ad-test.corp-ad.company.com
```

如果您要确保连接使用的是 Kerberos，请运行以下查询：

```
SELECT net_transport, auth_scheme
  FROM sys.dm_exec_connections
 WHERE session_id = @@SPID;
```

为 SQL Server 数据库实例设置 Windows 身份验证

使用 Amazon Directory Service for Microsoft Active Directory（也称为 Amazon Managed Microsoft AD）为 SQL Server 数据库实例设置 Windows 身份验证。要设置 Windows 身份验证，请执行下列步骤。

步骤 1：使用 Amazon Directory Service for Microsoft Active Directory 创建目录

Amazon Directory Service 将在 Amazon 云中创建完全托管的 Microsoft Active Directory。创建 Amazon Managed Microsoft AD 目录时，Amazon Directory Service 将代表您创建两个域控制器和域名服务 (DNS) 服务器。目录服务器在 VPC 内的两个不同可用区中的两个子网中创建。这种冗余有助于确保即使发生故障，目录仍可访问。

创建 Amazon Managed Microsoft AD 目录时，Amazon Directory Service 代表您执行以下任务：

- 在 VPC 中设置 Microsoft Active Directory。
- 使用用户名 Admin 和指定密码创建目录管理员账户。您可以使用此账户管理您的目录。
- 为目录控制器创建安全组。

在启动 Amazon Directory Service for Microsoft Active Directory 时，Amazon 创建一个组织单位 (OU)，其中包含目录的所有对象。此 OU (具有您在创建目录时键入的 NetBIOS 名称) 位于域根目录中。此域根目录由 Amazon 拥有和管理。

使用您的 Amazon Managed Microsoft AD 目录创建的 admin 账户对您的 OU 的最常见管理活动具有以下权限：

- 创建、更新或删除用户、组和计算机。
- 将资源添加到域 (如文件或打印服务器)，然后为 OU 中的用户和组分配这些资源的权限。
- 创建额外的 OU 和容器。
- 委托授权。
- 创建和链接组策略。
- 从 Active Directory 回收站还原删除的对象。
- 在 Active Directory Web 服务上运行 AD 和 DNS Windows PowerShell 模块。

admin 账户还具有执行下列域范围活动的权限：

- 管理 DNS 配置（添加、删除或更新记录、区域和转发器）。
- 查看 DNS 事件日志。
- 查看安全事件日志。

使用 Amazon Managed Microsoft AD 创建目录

1. 在 Amazon Directory Service 控制台导航窗格中，选择目录，然后选择设置目录。
2. 选择 Amazon Managed Microsoft AD。这是当前唯一支持用于 Amazon RDS 的选项。
3. 选择下一步。
4. 在输入目录信息页面上，提供以下信息：

   版本

   选择符合您要求的版本。

   目录 DNS 名称

   目录的完全限定名称，例如 `corp.example.com`。SQL Server 不支持超过 47 个字符的名称。

   目录 NetBIOS 名称

   可选的目录短名称，如 `CORP`。

   目录描述

   目录的可选描述。

   管理员密码

   目录管理员的密码。目录创建过程将使用用户名 Admin 和此密码创建一个管理员账户。

   目录管理员密码不能包含单词 `admin`。此密码区分大小写，且长度必须介于 8 – 64 个字符之间。至少，它还必须包含下列四种类别中三种类别的一个字符：

   - 小写字母 (a-z)
   - 大写字母 (A-Z)
   - 数字 (0-9)
   - 非字母数字字符 (~!@#$%^&*_-+=`|\(){}[]:;"'<>,.?/)

   确认密码

   重新键入管理员密码。
5. 选择下一步。
6. 在选择 VPC 和子网页面上，提供以下信息：

VPC

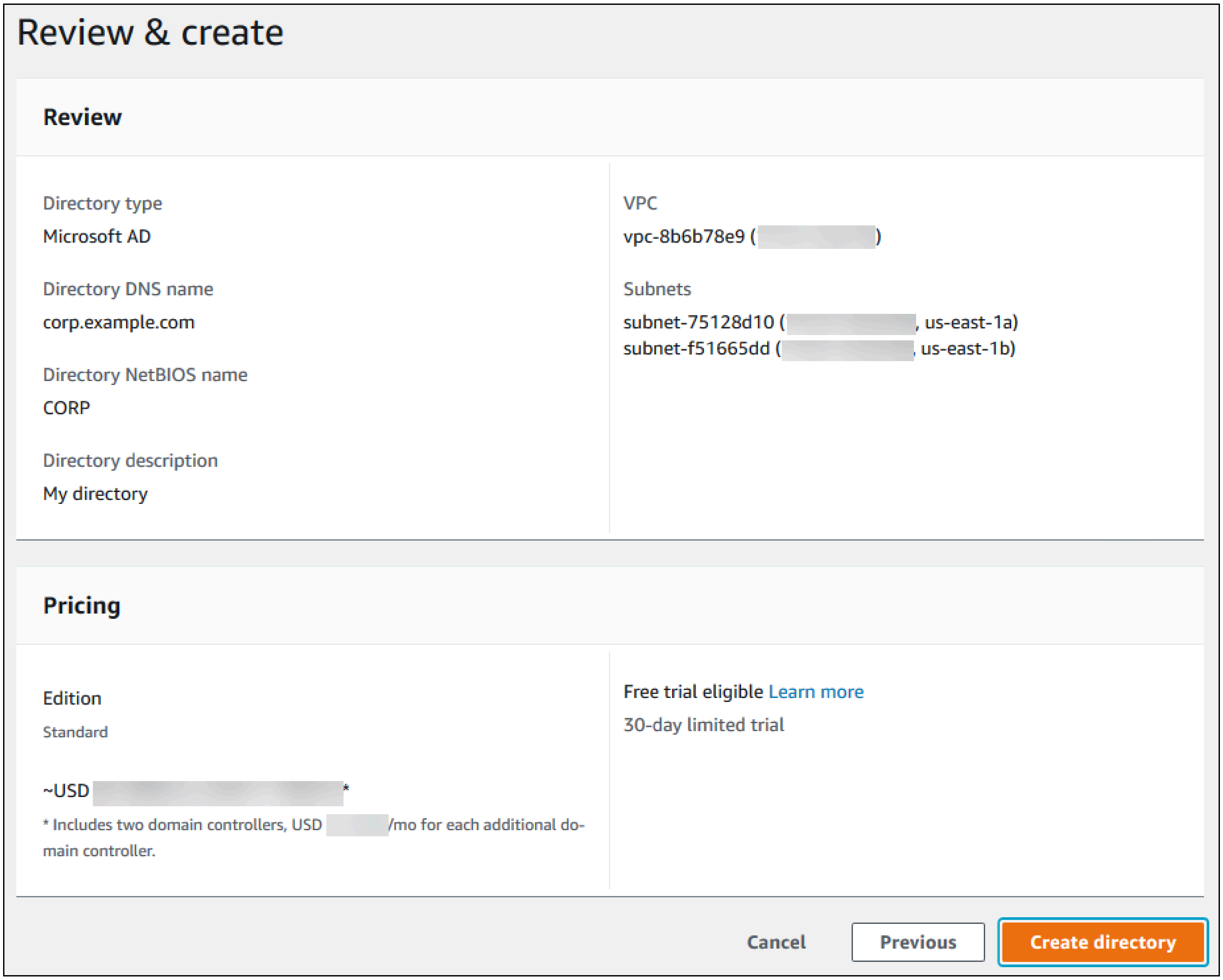
为目录选择 VPC。

> **Note**
>
> 您可以在不同 VPC 中查找目录和数据库实例，但如果这样做，请确保启用跨 VPC 流量。有关更多信息，请参阅“步骤 4：在目录和数据库实例之间启用跨 VPC 流量”。

子网

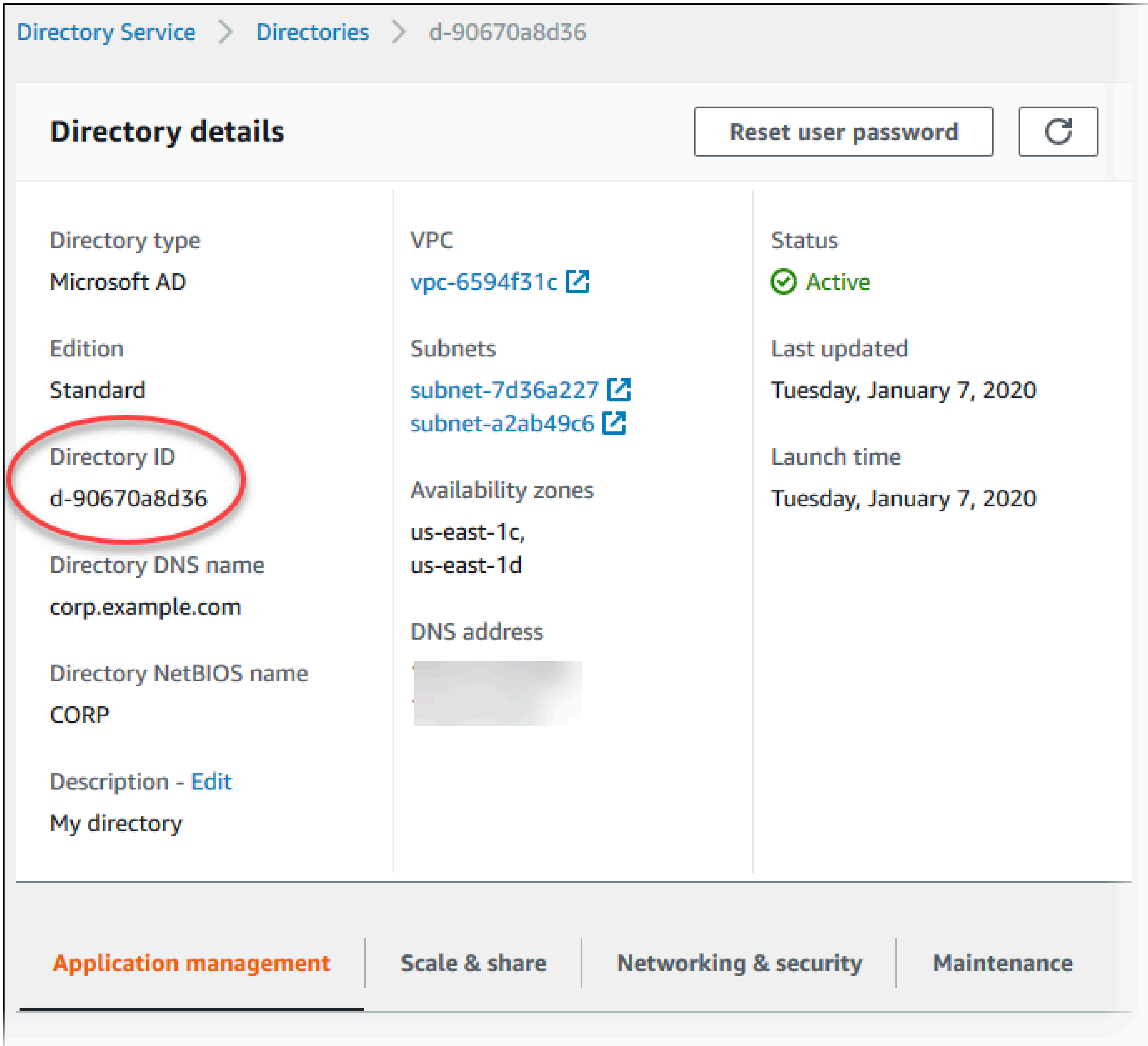
为目录服务器选择子网。两个子网必须位于不同的可用区。

7. 选择下一步。
8. 检查目录信息。如果需要进行更改，请选择上一步。如果信息正确，请选择 Create directory (创建目录)。



目录创建需要几分钟时间。创建成功后，Status (状态) 值将更改为 Active (活动)。

要查看有关您的目录的信息，请在目录列表中选择目录。记下目录 ID。在创建或修改 SQL Server 数据库实例时，您将需要此值。



步骤 2：创建 Amazon RDS 使用的 IAM 角色

如果您使用控制台创建您的 SQL Server 数据库实例，则可跳过此步。如果您已使用 CLI 或 RDS API 创建 SQL Server 数据库实例，则必须创建使用 `AmazonRDSDirectoryServiceAccess` 托管 IAM 策略的 IAM 角色。此角色允许 Amazon RDS 为您调用 Amazon Directory Service。

如果您使用自定义策略来加入域，而不是使用 Amazon 托管的 `AmazonRDSDirectoryServiceAccess` 策略，请确保允许 `ds:GetAuthorizedApplicationDetails` 操作。由于 Amazon Directory Service API 发生变化，这一要求自 2019 年 7 月起生效。

以下 IAM 策略 `AmazonRDSDirectoryServiceAccess` 提供对 Amazon Directory Service 的访问权限。

Example 用于提供 Amazon Directory Service 访问权限的 IAM 策略

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
            "ds:DescribeDirectories",
            "ds:AuthorizeApplication",
            "ds:UnauthorizeApplication",
            "ds:GetAuthorizedApplicationDetails"
        ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

我们建议在基于资源的信任关系中使用 [`aws:SourceArn`] 和 [`aws:SourceAccount`] 全局条件上下文键，以此限制服务对特定资源的权限。这是防范[混淆代理问题]最有效的方法。

您可以使用这两个全局条件上下文键并让 `aws:SourceArn` 值包含账户 ID。在这种情况下，当 `aws:SourceAccount` 值和 `aws:SourceArn` 值中的账户使用相同策略语句时，确保二者使用相同的账户 ID。

- 如果您想对单个资源进行跨服务访问，请使用 `aws:SourceArn`。
- 如果您想允许该账户中的任何资源与跨服务使用操作相关联，请使用 `aws:SourceAccount`。

在信任关系中，请务必使用 `aws:SourceArn` 全局条件上下文键和访问角色资源的完整 Amazon Resource Name (ARN)。对于 Windows 身份验证，请确保包含数据库实例，如以下示例所示。

Example 与 Windows 身份验证的全局条件上下文键的信任关系

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Effect": "Allow",
            "Principal": {
                "Service": "rds.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceArn": [
                        "arn:aws:rds:Region:my_account_ID:db:db_instance_identifier"
                    ]
                }
            }
        }
    ]
}
```

使用此 IAM 策略和信任关系创建 IAM 角色。有关创建 IAM 角色的更多信息，请参阅 IAM 用户指南 中的创建客户托管策略。

步骤 3：创建和配置用户和组

您可以使用“Active Directory 用户和计算机”工具创建用户和组。该工具是 Active Directory Domain Services 和 Active Directory Lightweight Directory Services 工具之一。用户表示有权访问您的目录的独立个人或实体。对于针对用户组授予或拒绝权限非常有用，从而不必将这些权限应用于每个独立用户。

要在 Amazon Directory Service 目录中创建用户和组，您必须连接到属于 Amazon Directory Service 目录成员的 Windows EC2 实例。您还必须以具有创建用户和组权限的用户身份登录。有关更多信息，请参阅 Amazon Directory Service管理指南中的添加用户和组（Simple AD 和 Amazon Managed Microsoft AD）。

步骤 4：在目录和数据库实例之间启用跨 VPC 流量

如果您打算将目录和数据库实例放在同一 VPC 中，请跳过该步骤，然后转到步骤 5：创建或修改 SQL Server 数据库实例。

如果您计划在不同 VPC 中查找目录和数据库实例，请使用 VPC 对等连接或 Amazon Transit Gateway 配置跨 VPC 流量。

以下过程使用 VPC 对等连接启用 VPC 之间的流量。请按照 Amazon Virtual Private Cloud 对等连接指南中的什么是 VPC 对等连接？操作。

使用 VPC 对等连接启用跨 VPC 流量

1. 设置适合的 VPC 路由规则，以便确保网络流量可以双向流动。
2. 确保数据库实例的安全组可以从目录的安全组接收入站流量。
3. 确保没有任何网络访问控制列表 (ACL) 规则会阻止流量。

如果该目录由不同的 Amazon 账户拥有，则您必须共享该目录。

在 Amazon 账户之间共享目录

1. 按照 Amazon 管理指南中的教程：共享 Amazon Managed Microsoft AD 目录以实现无缝 EC2 域加入中的说明，开始与将要在其中创建数据库实例的 Amazon Directory Service 账户共享目录。
2. 使用数据库实例的账户登录到 Amazon Directory Service 控制台，并确保在处理之前域具有 `SHARED` 状态。
3. 使用数据库实例的账户登录 Amazon Directory Service 控制台时，请记录目录 ID 值。您可以使用此目录 ID 将数据库实例加入域。

步骤 5：创建或修改 SQL Server 数据库实例

创建或修改 SQL Server 数据库实例，以便与您的目录一起使用。您可以使用控制台、CLI 或 RDS API 将数据库实例与目录关联。您可以通过下列方式之一来执行该操作：

- 使用控制台、create-db-instance CLI 命令或 CreateDBInstance RDS API 操作创建新的 SQL Server 数据库实例。

  有关说明，请参阅创建 Amazon RDS 数据库实例。
- 使用控制台、modify-db-instance CLI 命令或 ModifyDBInstance RDS API 操作修改现有 SQL Server 数据库实例。

  有关说明，请参阅修改 Amazon RDS 数据库实例。
- 使用控制台、restore-db-instance-from-db-snapshot CLI 命令或 RestoreDBInstanceFromDBSnapshot RDS API 操作，从数据库快照还原 SQL Server 数据库实例。

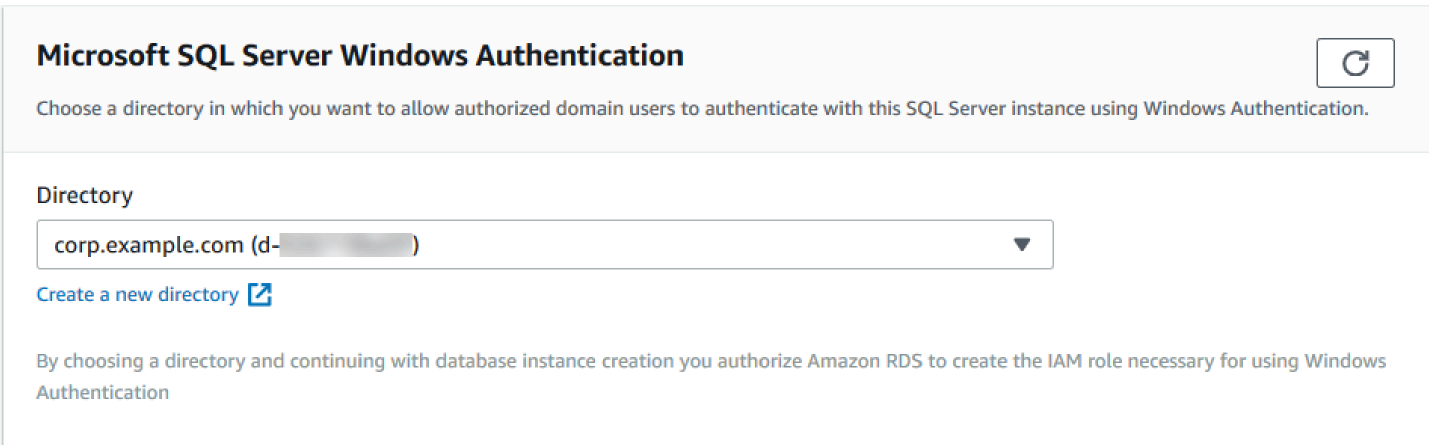
  有关说明，请参阅还原到数据库实例。
- 使用控制台、restore-db-instance-to-point-in-time CLI 命令或 RestoreDBInstanceToPointInTime RDS API 操作，将 SQL Server 数据库实例还原到某个时间点。

  有关说明，请参阅将 Amazon RDS 的数据库实例还原到指定时间。

只有 VPC 中的 SQL Server 数据库实例才支持 Windows 身份验证。

要使数据库实例能够使用您创建的域目录，需要满足以下条件：

- 对于目录，您必须选择创建目录时生成的域标识符 (d-*ID*)。
- 确保 VPC 安全组具有允许数据库实例与目录通信的出站规则。



使用 Amazon CLI 时，数据库实例需要以下参数才能使用您创建的目录：

- 对于 `--domain` 参数，请使用创建目录时生成的域标识符 (d-*ID*)。
- 对于 `--domain-iam-role-name` 参数，请使用您通过托管 IAM 策略 `AmazonRDSDirectoryServiceAccess` 创建的角色。

例如，以下 CLI 命令会修改数据库实例以使用目录。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier mydbinstance \
    --domain d-ID \
    --domain-iam-role-name role-name
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier mydbinstance ^
    --domain d-ID ^
```

```
    --domain-iam-role-name role-name
```

> **Important**
>
> 如果您修改数据库实例以启用 Kerberos 身份验证，请在进行更改之后重启数据库实例。

步骤 6：创建 Windows 身份验证 SQL Server 登录名

使用 Amazon RDS 主用户凭证连接到 SQL Server 数据库实例，就像连接到任何其他数据库实例一样。由于数据库实例已加入 Amazon Managed Microsoft AD 域，因此您可以预配置 SQL Server 登录名和用户。您可以从域中的 Active Directory 用户和组执行此操作。通过对这些 Windows 登录名授予和撤销标准 SQL Server 权限来管理数据库权限。

对于 Active Directory 用户，要对 SQL Server 进行身份验证，此用户或其所属的组必须存在 SQL Server Windows 登录名。访问权限的精细控制是通过对这些 SQL Server 登录名授予和撤销权限来处理的。用户没有 SQL Server 登录名，或者用户所属组中的登录名无法访问 SQL Server 数据库实例。

创建 Active Directory SQL Server 登录名需要 ALTER ANY LOGIN 权限。如果您未使用此权限创建任何登录名，请使用 SQL Server 身份验证作为数据库实例的主用户进行连接。

运行数据定义语言 (DDL) 命令（如以下示例）为 Active Directory 用户或组创建 SQL Server 登录名。

> **Note**
>
> 使用 Windows 2000 之前的登录名称指定用户和组，格式为 *domainName**login_name*。您不能使用 *login_name*@*DomainName* 格式的用户主体名 (UPN)。
> 您只能使用 T-SQL 语句在 RDS for SQL Server 实例上创建 Windows 身份验证登录名。您无法使用 SQL Server Management Studio 创建 Windows 身份验证登录名。

```
USE [master]
GO
CREATE LOGIN [mydomain\myuser] FROM WINDOWS WITH DEFAULT_DATABASE = [master],
 DEFAULT_LANGUAGE = [us_english];
GO
```

有关更多信息，请参阅 Microsoft 开发人员网络文档中的 [CREATE LOGIN (Transact-SQL)](#)。

域中用户（人和应用程序）现在均可从加入域的客户端计算机使用 Windows 身份验证连接到 RDS for SQL Server 实例。

在域中管理数据库实例

您可以使用控制台、Amazon CLI 或 Amazon RDS API 来管理数据库实例及其与您的域的关系。例如，您可以将数据库实例移入域、移出域或在域之间移动。

例如，使用 Amazon RDS API，您可以执行下列操作：

- 要为失败的成员资格重新尝试域加入，请使用 [ModifyDBInstance](https://docs.aws.amazon.com/AmazonRDS/latest/APIReference/API_ModifyDBInstance.html) API 操作并指定当前成员资格的目录 ID。
- 要为成员资格更新 IAM 角色名称，请使用 `ModifyDBInstance` API 操作并指定当前成员资格的目录 ID 和新的 IAM 角色。
- 要从域中删除数据库实例，请使用 `ModifyDBInstance` API 操作并指定 none 作为域参数。
- 要将数据库实例从一个域移至另一个域，请使用 `ModifyDBInstance` API 操作并指定新域的域标识符作为域参数。
- 要列出每个数据库实例的成员资格，请使用 [DescribeDBInstances](https://docs.aws.amazon.com/AmazonRDS/latest/APIReference/API_DescribeDBInstances.html) API 操作。

了解域成员资格

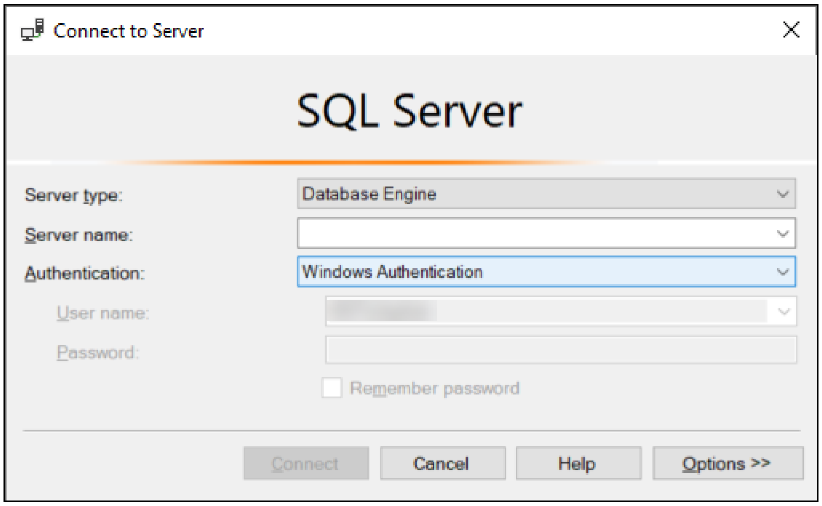
在创建或修改数据库实例后，实例将成为域的成员。Amazon 控制台将指示数据库实例的域成员资格状态。数据库实例的状态可以是以下状态之一：

- 已联接 – 实例是域的成员。
- 正在联接 – 实例正处于成为域成员的过程中。
- 待联接 – 实例成员资格待定。
- pending-maintenance-join – Amazon将在下一计划维护期间尝试使实例成为域成员。
- 待删除 – 等待从域中删除实例。
- pending-maintenance-removal – Amazon将在下一计划维护期间尝试从域中删除实例。
- 失败 – 配置问题阻碍实例联接域。在重新发出实例修改命令之前检查并修复配置。
- 正在删除 – 正从域中删除实例。

成为域成员的请求可能因网络连接问题或不正确的 IAM 角色而失败。例如，您可能会创建数据库实例或修改现有实例，并且尝试使数据库成为某个域的成员会失败。在这种情况下，您重新发出命令以创建或修改数据库实例，或者修改新创建的实例以加入域。

使用 Windows 身份验证连接到 SQL Server

要使用 Windows 身份验证连接到 SQL Server，您必须作为域用户登录联接域的计算机。在启动 SQL Server Management Studio 之后，请选择 Windows 身份验证作为身份验证类型，如下所示。



还原 SQL Server 数据库实例，然后将其添加到域中

您可还原数据库快照或为 SQL Server 数据库实例执行时间点恢复 (PITR)，然后将其添加到域中。数据库实例还原后，使用[步骤 5：创建或修改 SQL Server 数据库实例](#)中介绍的过程修改此实例以将数据库实例添加到域中。

升级 Microsoft SQL Server 数据库引擎

如果 Amazon RDS 支持数据库引擎的新版本，您可以将您的数据库实例升级到新版本。SQL Server 数据库实例具有两种升级：主要版本升级和次要版本升级。

主要版本升级会包含不与现有应用程序向后兼容的数据库更改。因此，您必须手动为数据库实例执行主要版本升级。您可以通过修改数据库实例来启动主要版本升级。不过，在执行主要版本升级之前，建议您按照测试 RDS for SQL Server 升级中所述的步骤测试升级。

次要版本升级仅包含与现有应用程序向后兼容的更改。您可以通过两种方式升级数据库实例的次要版本：

- 手动：修改您的数据库实例以启动升级
- 自动：为数据库实例启用自动次要版本升级

启用自动次要版本升级后，当较新的次要版本中有关键安全更新可用时，RDS for SQL Server 会在计划的维护时段内自动升级数据库实例。

对于 `16.00.4120.1`、`15.00.4365.2`、`14.00.3465.1`、`13.00.6435.1` 之后的次要引擎版本，默认情况下会禁用以下安全协议：

- `rds.tls10`（TLS 1.0 协议）
- `rds.tls11`（TLS 1.1 协议）
- `rds.rc4`（RC4 密码）
- `rds.curve25519`（Curve25519 加密）
- `rds.3des168`（三重 DES 加密）

对于较早的引擎版本，Amazon RDS 默认情况下启用这些安全协议。

```
...

"ValidUpgradeTarget": [
    {
        "Engine": "sqlserver-se",
        "EngineVersion": "14.00.3281.6.v1",
        "Description": "SQL Server 2017 14.00.3281.6.v1",
        "AutoUpgrade": false,
        "IsMajorVersionUpgrade": false
    }
```

```
...
```

有关执行升级的更多信息，请参阅升级 SQL Server 数据库实例。有关在 Amazon RDS 上可用的 SQL Server 版本的信息，请参阅 Amazon RDS for Microsoft SQL Server。

主题

RDS for SQL Server 的主要版本升级

Amazon RDS 当前支持对 Microsoft SQL Server 数据库实例进行以下主要版本升级。

您可以将现有数据库实例从 SQL Server 2008 之外的任何版本升级到 SQL Server 2017 或 2019。要从 SQL Server 2008 升级，请先升级到其他版本之一。

当前版本	支持的升级版本
SQL Server 2019	SQL Server 2022
SQL Server 2017	SQL Server 2022 SQL Server 2019
SQL Server 2016	SQL Server 2022 SQL Server 2019 SQL Server 2017

您可以使用 Amazon CLI 查询（如以下示例）来查找特定数据库引擎版本的可用升级。

Example

对于 Linux、macOS 或 Unix：

```
aws rds describe-db-engine-versions \
    --engine sqlserver-se \
    --engine-version 14.00.3281.6.v1 \
    --query "DBEngineVersions[*].ValidUpgradeTarget[*].{EngineVersion:EngineVersion}" \
    --output table
```

对于 Windows：

```
aws rds describe-db-engine-versions ^
    --engine sqlserver-se ^
    --engine-version 14.00.3281.6.v1 ^
    --query "DBEngineVersions[*].ValidUpgradeTarget[*].{EngineVersion:EngineVersion}" ^
    --output table
```

输出显示您可以将版本 14.00.3281.6 升级到最新推出的 SQL Server 2017 或 2019 版本。

```
--------------------------
|DescribeDBEngineVersions|
+------------------------+
|      EngineVersion     |
+------------------------+
|  14.00.3294.2.v1       |
|  14.00.3356.20.v1      |
|  14.00.3381.3.v1       |
|  14.00.3401.7.v1       |
|  14.00.3421.10.v1      |
|  14.00.3451.2.v1       |
|  15.00.4043.16.v1      |
|  15.00.4073.23.v1      |
|  15.00.4153.1.v1       |
|  15.00.4198.2.v1       |
|  15.00.4236.7.v1       |
+------------------------+
```

数据库兼容性级别

您可以使用 Microsoft SQL Server 数据库兼容性级别调整某些数据库行为，以模仿之前版本的 SQL Server。有关更多信息，请参阅 Microsoft 文档中的兼容性级别。在升级数据库实例时，任何现有的数据库将保持其原始兼容性级别。

您可以使用 ALTER DATABASE 命令更改数据库的兼容性级别。例如，要将名为 `customeracct` 的数据库更改为与 SQL Server 2016 兼容，请发出以下命令：

```
ALTER DATABASE customeracct SET COMPATIBILITY_LEVEL = 130
```

SQL Server 升级的注意事项

在升级流程期间，Amazon RDS 会创建两个数据库快照。第一个数据库快照是数据库实例在进行任何升级更改前的。第二个数据库快照是在升级结束后制作的。

> **Note**
>
> 如果您将数据库实例的备份保留期设置为大于 0 的数字，则 Amazon RDS 只需创建数据库快照。要更改您的备份保留期，请参阅 修改 Amazon RDS 数据库实例。

升级完成后，您无法恢复为之前版本的数据库引擎。如果您想恢复到以前的版本，请从升级前创建的数据库快照还原以创建新的数据库实例。

在 SQL Server 的次要或主要版本升级期间，Free Storage Space和 Disk Queue Depth 指标将显示 `-1`。升级完毕后，这两个指标都将恢复正常。

在升级 SQL Server 实例之前，请查看以下信息。

主题

启动升级之前的最佳实践

在开始升级过程之前，请实施以下准备步骤，以实现最佳的升级性能并最大限度地减少潜在问题：

时间和工作负载管理

- 安排在事务量较低的时段进行升级。

- 在升级时段内尽量减少写入操作。

  这样，Amazon RDS 就能够通过减少 RDS 在次要到主要配对期间需要还原的事务日志备份文件的数量，来更快地完成升级。

事务管理

- 确定并监控长时间运行的事务。
- 在开始升级之前，确保所有关键事务都已提交。
- 防止在升级时段内具有长时间运行的事务。

日志文件优化

  查看和优化事务日志文件：

- 缩小大小过大的日志文件。
- 减少较高的日志消耗模式。
- 管理虚拟日志文件（VLF）。
- 为正常操作保留充足的可用空间。

多可用区注意事项

Amazon RDS 通过使用 SQL Server 数据库镜像 (DBM) 或 Always On 可用性组 (AG) 支持为运行 Microsoft SQL Server 的数据库实例进行多可用区部署。有关更多信息，请参阅 Amazon RDS for Microsoft SQL Server 多可用区部署。

在多可用区部署（Mirroring/AlwaysOn）中，当请求升级时，RDS 会对主实例和辅助实例采用滚动升级策略。滚动升级可确保在辅助实例升级期间至少有一个实例可用于事务。预计中断只会持续失效转移的这段期间。

在升级期间，RDS 会将辅助实例从多可用区配置中移除，对辅助实例执行升级，并还原在主实例断开连接期间从主实例获取的所有事务日志备份。还原所有日志备份后，RDS 会将升级后的辅助实例加入主实例。当所有数据库都处于同步状态时，RDS 会失效转移到已升级的辅助实例。失效转移完成后，RDS 将继续升级旧的主实例，还原所有事务日志备份，并使其与新的主实例配对。

为了尽量减少此失效转移持续时间，我们建议，当使用在连接字符串中支持 `MultiSubnetFailover` 连接选项的客户端库时，使用 AlwaysOn AG 可用性组侦听器端点。使用可用性组侦听器端点时，失效转移时间通常少于 10 秒，但是，此持续时间不包括任何额外的崩溃恢复时间。

只读副本注意事项

在数据库版本升级期间，Amazon RDS 会升级所有只读副本以及主数据库实例。Amazon RDS 不支持单独对只读副本进行数据库版本升级。有关只读副本的更多信息，请参阅 在 Amazon RDS 中使用 Microsoft SQL Server 的只读副本。

当您执行主数据库实例的数据库版本升级时，该实例的所有只读副本也会自动升级。Amazon RDS 将在升级主数据库实例之前同时升级所有只读副本。在主数据库实例上的数据库版本升级完成之前，只读副本可能不可用。

选项组注意事项

如果数据库实例使用自定义数据库选项组，则在某些情况下，Amazon RDS 无法自动向数据库实例分配新的选项组。例如，在您升级到新的主要版本时，必须指定新的选项组。我们建议您创建新的选项组，然后向该组添加与现有自定义选项组相同的选项。

有关更多信息，请参阅 创建选项组 或 复制选项组。

参数组注意事项

如果您的数据库实例使用自定义数据库参数组：

- Amazon RDS 在升级后自动重新启动数据库实例。
- 在某些情况下，RDS 无法自动为您的数据库实例分配新参数组。

  例如，在升级到新的主要版本时，必须指定新参数组。我们建议您创建新参数组，然后配置您现有的自定义参数组中的那些参数。

有关更多信息，请参阅 在 Amazon RDS 中创建数据库参数组 或 在 Amazon RDS 中复制数据库参数组。

测试 RDS for SQL Server 升级

在数据库实例上执行主要版本升级前，应该彻底地测试数据库以及访问该数据库的所有应用程序，了解是否与新版本兼容。我们建议您使用以下过程。

测试主要版本升级

1. 对于数据库引擎的新版本，查看 Microsoft 文档中的升级 SQL Server，以了解是否存在可能会影响数据库或应用程序的兼容性问题。

2. 如果数据库实例使用自定义选项组，则创建与您正升级到的新版本兼容的新选项组。有关更多信息，请参阅“选项组注意事项”。
3. 如果数据库实例使用自定义参数组，则创建与您正升级到的新版本兼容的新参数组。有关更多信息，请参阅“参数组注意事项”。
4. 为要升级的数据库实例创建数据库快照。有关更多信息，请参阅“为 Amazon RDS 的单可用区数据库实例创建数据库快照”。
5. 恢复数据库快照，以便创建一个新的测试数据库实例。有关更多信息，请参阅“还原到数据库实例”。
6. 使用下面介绍的一种方法来修改此新测试数据库实例，以将其升级到新版本：
    - 控制台
    - Amazon CLI
    - RDS API
7. 计算升级的实例使用的存储容量以决定升级是否需要额外的存储容量。
8. 对升级的数据库实例运行所需次数的质量保证测试以确保数据库和应用程序可正常使用新版本。实施任何必需的新测试，评估您在步骤 1 中发现的任何兼容性问题的影响。测试所有的存储过程和功能。将应用程序的测试版本指向所升级的数据库实例。
9. 如果所有测试通过，那么请在生产数据库实例上执行升级。我们建议您直到确认一切正常后，再允许对数据库实例执行写入操作。

升级 SQL Server 数据库实例

有关手动或自动升级 SQL Server 数据库实例的信息，请参阅以下信息：

- 升级数据库实例引擎版本
- 在 Amazon RDS for SQL Server 上将 SQL Server 2008 R2 升级到 SQL Server 2016 的最佳实践

> **Important**
>
> 如果您有任何使用 Amazon KMS 进行加密的快照，建议您在支持结束之前启动升级。

在支持结束之前升级已弃用的数据库实例

主要版本被弃用之后，您不能在新数据库实例上安装它。RDS 将尝试自动升级所有现有数据库实例。

如果您需要恢复已弃用的数据库实例，可以执行时间点恢复 (PITR) 或还原快照。这样做可让您获得使用已弃用版本的数据库实例的临时访问权。但是，在主要版本完全弃用之后，这些数据库实例也将自动升级到支持的版本。

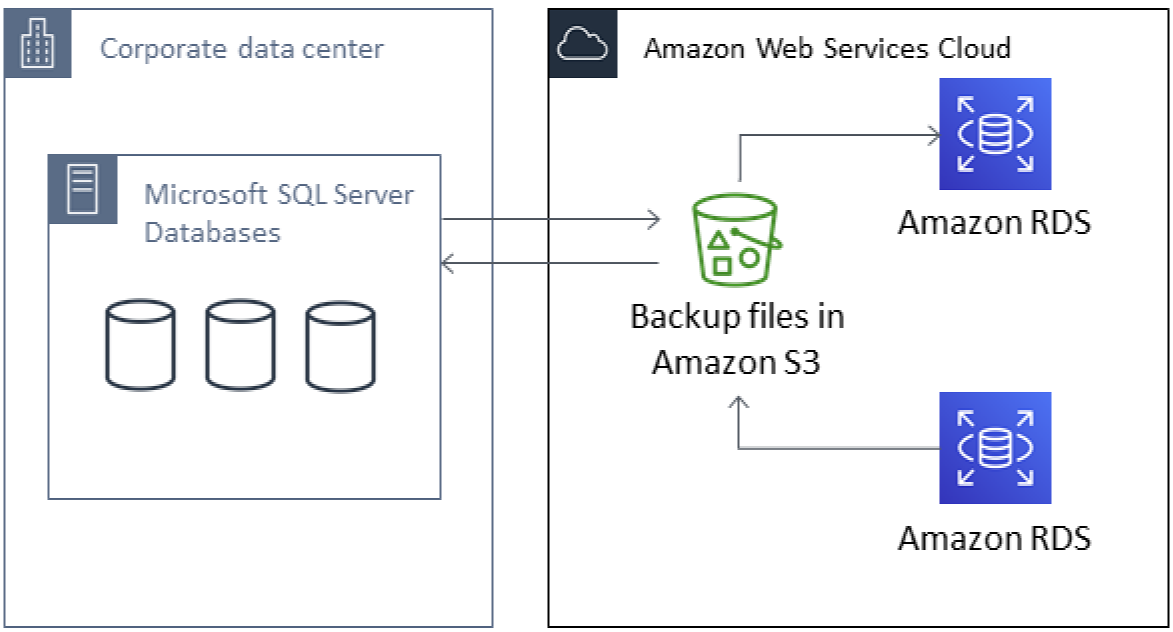
使用本机备份和还原导入和导出 SQL Server 数据库

Amazon RDS 支持使用完整备份文件 (.bak 文件) 对 Microsoft SQL Server 数据库进行本机备份和还原。当您使用 RDS 时，您将访问存储在 Amazon S3 中的文件，而不是使用数据库服务器上的本地文件系统。

例如，您可以从本地服务器创建完整备份，将该备份存储到 S3 上，然后将其还原到现有 Amazon RDS 数据库实例。您还可以从 RDS 进行备份，将其存储在 S3 上，然后在需要的任何位置还原它们。

本机备份和还原适用于所有 Amazon 区域中的单可用区和多可用区数据库实例，包括具有只读副本的多可用区数据库实例。本机备份和还原适用于 Amazon RDS 支持的所有 Microsoft SQL Server 版本。

下图说明了支持的方案。



使用本机 .bak 文件备份和还原数据库通常是备份和还原数据库的最快方法。此外，使用本机备份和还原还提供了很多其他优势。例如，您可以执行以下操作：

- 将数据库迁移到 Amazon RDS 或从后者迁移数据库。
- 在 RDS for SQL Server 数据库实例之间移动数据库。
- 迁移 .bak 文件中的数据、架构、存储过程、触发器和其他数据库代码。
- 备份和还原单个数据库，而不是整个数据库实例。
- 创建用于开发、测试、培训和演示的数据库副本。
- 通过 Amazon S3 传输备份文件，为您提供一层额外保护以进行灾难恢复。

- 创建开启了透明数据加密 (TDE) 的数据库的本机备份，并将这些备份还原到本地数据库。有关更多信息，请参阅 SQL Server 中的透明数据加密支持。
- 将开启了 TDE 的本地数据库的本机备份还原到 RDS for SQL Server 数据库实例。有关更多信息，请参阅 SQL Server 中的透明数据加密支持。

目录

限制和建议

以下是有关对本机备份和还原的使用的一些限制：

- 您不能备份到与您的 Amazon RDS 数据库实例不同的Amazon区域中的某个 Amazon S3 存储桶或从该存储桶中进行还原。
- 如果存在同名的现有数据库，则无法还原数据库。数据库名称是唯一的。
- 强烈建议您不要将备份从一个时区还原到另一个时区。如果您将备份从一个时区还原到另一个时区，则必须审核您的查询和应用程序以查看时区更改的影响。
- Amazon S3 具有每个文件 5 TB 的大小限制。若要在本机备份更大的数据库，则可以使用多文件备份。
- 可以备份到 S3 的最大数据库大小取决于数据库实例上的可用内存、CPU、I/O 和网络资源。数据库越大，备份代理占用的内存就越多。
- 不能同时对 10 个以上的备份文件执行备份或还原操作。
- 差异备份基于上一个完整备份。要使差异备份起作用，您不能在上一个完整备份和差异备份之间拍摄快照。如果您需要差异备份，但存在手动或自动快照，请先进行另一个完整备份，然后再继续进行差异备份。
- 其文件的 file_guid（唯一标识符）设置为 NULL 的数据库不支持差异还原和日志还原。
- 您最多可以同时运行两个备份或还原任务。
- 您无法在 Amazon RDS 上从 SQL Server 执行本机日志备份。
- RDS 支持最大 64 TiB 的数据库本机还原。SQL Server Express Edition 上的数据库的本机还原限制为 10 GB。

- 您无法在维护时段内或在 Amazon RDS 处于创建数据库快照过程中的任何时间对数据库进行本机备份。如果本机备份任务与 RDS 每日备份时段重叠，则会取消本机备份任务。
- 在多可用区数据库实例上，您只能在本机还原在完全恢复模式下备份的数据库。
- 不支持从多可用区实例上的差异备份还原。
- 不支持在事务内调用 RDS 过程进行本机备份和还原。
- 请使用对称加密 Amazon KMS key 来加密您的备份。Amazon RDS 不支持非对称 KMS 密钥。有关更多信息，请参阅《Amazon Key Management Service 开发人员指南》中的创建对称 KMS 密钥。
- 本机备份文件使用“仅加密”加密模式通过指定的 KMS 密钥加密。当您还原加密的备份文件时，注意它们是使用“仅加密”加密模式加密的。
- 您无法还原包含 FILESTREAM 文件组的数据库。

如果在创建、复制和还原备份文件时，您的数据库可处于脱机状态，建议您使用本机备份和还原将其迁移到 RDS。如果您的本地数据库不能处于脱机状态，建议您使用 Amazon Database Migration Service 将您的数据库迁移到 Amazon RDS。有关更多信息，请参阅什么是 Amazon Database Migration Service？

本机备份和还原不是为了替换跨区域快照复制功能的数据恢复功能。建议您使用快照复制为 Amazon RDS 中的跨区域灾难恢复将数据库快照复制到另一Amazon区域。有关更多信息，请参阅 复制 Amazon RDS 的数据库快照。

针对本机备份和还原进行设置

要设置本机备份和还原，您需要三个组件：

1. 用于存储备份文件的 Amazon S3 存储桶。

   您必须将 S3 存储桶用于备份文件，然后上传您要迁移到 RDS 的备份。如果您已有一个 Amazon S3 存储桶，则可以使用它。如果没有，则可以创建存储桶。或者，您可以选择在使用 `SQLSERVER_BACKUP_RESTORE` 添加 Amazon Web Services Management Console 选项时为自己创建新的存储桶。

   有关使用 S3 的信息，请参阅 Amazon Simple Storage Service 用户指南
2. 用于访问存储桶的 Amazon Identity and Access Management (IAM) 角色。

   如果您已有一个 IAM 角色，则可以使用它。您可以选择在使用Amazon Web Services Management Console添加 `SQLSERVER_BACKUP_RESTORE` 选项时为自己创建新的 IAM 角色。或者，您可以手动创建一个新的角色。

如果您想要手动创建新的 IAM 角色，请使用下一部分中介绍的方法。如果要将信任关系和权限策略附加到现有 IAM 角色，请执行相同操作。

3. 已添加到数据库实例上选项组的 `SQLSERVER_BACKUP_RESTORE` 选项。

   要在数据库实例上启用本机备份和还原，请将 `SQLSERVER_BACKUP_RESTORE` 选项添加到数据库实例上的选项组。有关更多信息和说明，请参阅 SQL Server 中对本机备份和还原的支持。

为本机备份和还原手动创建 IAM 角色

如果要手动创建新的 IAM 角色以用于本机备份和还原，可以这样做。在这种情况下，您将创建一个角色，以将权限从 Amazon RDS 服务委派给 Amazon S3 存储桶。创建 IAM 角色时，您将附加信任关系和权限策略。信任关系允许 RDS 代入此角色。这些权限策略定义此角色可以执行的操作。有关创建角色的更多信息，请参阅创建将权限委派给 Amazon 服务的角色。

对于本机备份和还原功能，可使用类似于本节中以下示例的信任关系和权限策略。在下面的示例中，我们使用服务委托人名称 `rds.amazonaws.com` 作为所有服务账户的别名。在其他示例中，我们指定 Amazon Resource Name (ARN) 以标识我们在信任策略中授予访问权限的其他账户、用户或角色。

我们建议在基于资源的信任关系中使用 `aws:SourceArn` 和 `aws:SourceAccount` 全局条件上下文键，以此限制服务对特定资源的权限。这是防范混淆代理问题最有效的方法。

您可以使用这两个全局条件上下文键并让 `aws:SourceArn` 值包含账户 ID。在这种情况下，当 `aws:SourceAccount` 值和 `aws:SourceArn` 值中的账户使用相同策略语句时，确保二者使用相同的账户 ID。

- 如果您想对单个资源进行跨服务访问，请使用 `aws:SourceArn`。
- 如果您想允许该账户中的任何资源与跨服务使用操作相关联，请使用 `aws:SourceAccount`。

在信任关系中，请务必使用 `aws:SourceArn` 全局条件上下文键和访问角色资源的完整 ARN。对于本机备份和还原，请确保同时包含数据库选项组和数据库实例，如以下示例所示。

Example 与本机备份和还原的全局条件上下文键的信任关系

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Principal": {
                "Service": "rds.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceArn": [
                        "arn:aws:rds:Region:0123456789:db:db_instance_identifier",
                        "arn:aws:rds:Region:0123456789:og:option_group_name",
                  "arn:aws:s3:::amzn-s3-demo-bucket"
                    ],
                    "aws:SourceAccount": "0123456789"
                }
            }
        }
    ]
}
```

以下示例使用 ARN 指定资源。有关使用 ARN 的更多信息，请参阅 Amazon 资源名称（ARN）。

Example 不带加密支持的适用于本机备份和还原的权限策略

```
{
    "Version": "2012-10-17",
    "Statement":
    [
        {
        "Effect": "Allow",
        "Action":
            [
                "s3:ListBucket",
                "s3:GetBucketLocation"
            ],
        "Resource": "arn:aws:s3:::amzn-s3-demo-bucket"
        },
        {
        "Effect": "Allow",
        "Action":
            [
                "s3:GetObjectAttributes",
                "s3:GetObject",
                "s3:PutObject",
                "s3:ListMultipartUploadParts",
                "s3:AbortMultipartUpload"
```

```
            ],
        "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
        }
    ]
}
```

使用本机备份和还原

在启用并配置本机备份和还原功能后，可以开始使用该功能。先连接到您的 Microsoft SQL Server 数据库，然后调用 Amazon RDS 存储过程来执行该操作。有关连接到数据库的说明，请参阅 连接到 Microsoft SQL Server 数据库实例。

某些存储过程要求您向 Amazon S3 存储桶和文件提供 Amazon Resource Name (ARN)。您的 ARN 的格式为 `arn:aws:s3:::bucket_name/file_name.extension`。Amazon S3 不需要在 ARN 中使用账号或 Amazon 区域。

如果您还提供可选 KMS 密钥，则密钥 ARN 的格式为 `arn:aws:kms:region:account-id:key/key-id`。有关更多信息，请参阅 Amazon Resource Name (ARN) 和 Amazon 服务命名空间。您必须使用对称加密 KMS 密钥来加密您的备份。Amazon RDS 不支持非对称 KMS 密钥。有关更多信息，请参阅《Amazon Key Management Service 开发人员指南》中的创建对称 KMS 密钥。

> **Note**
>
> 无论您是否使用 KMS 密钥，本机备份和还原任务都会对上传到 S3 的文件默认启用服务器端高级加密标准 (AES) 256 位加密。

有关如何调用每个存储过程的说明，请参阅以下主题：

- 备份数据库
- 还原数据库
- 还原日志
- 完成数据库还原
- 处理部分还原的数据库
- 取消任务
- 跟踪任务的状态

备份数据库

要备份数据库，请使用 `rds_backup_database` 存储过程。

> **Note**
> 您无法在维护时段内对数据库进行备份，也无法在 Amazon RDS 拍摄快照时对数据库进行备份。

使用量

```
exec msdb.dbo.rds_backup_database
 @source_db_name='database_name',
 @s3_arn_to_backup_to='arn:aws:s3:::bucket_name/file_name.extension',
 [@kms_master_key_arn='arn:aws:kms:region:account-id:key/key-id'],
 [@overwrite_s3_backup_file=0|1],
 [@block_size=512|1024|2048|4096|8192|16384|32768|65536],
        [@max_transfer_size=n],
        [@buffer_count=n],
 [@type='DIFFERENTIAL|FULL'],
 [@number_of_files=n];
```

以下参数为必需参数：

- `@source_db_name` – 要备份的数据库的名称。
- `@s3_arn_to_backup_to` – 指示要用于备份的 Amazon S3 存储桶以及备份文件的名称的 ARN。

  此文件可以具有任何扩展名，但通常使用 `.bak`。

以下参数可选：

- `@kms_master_key_arn` – 用于加密项目的对称加密 KMS 密钥的 ARN。
  - 您不能使用默认加密密钥。如果使用默认密钥，则不会备份数据库。
  - 如果未指定 KMS 密钥标识符，则不会对备份文件进行加密。有关更多信息，请参阅加密 Amazon RDS 资源。
  - 指定 KMS 密钥时，将使用客户端加密。
  - Amazon RDS 不支持非对称 KMS 密钥。有关更多信息，请参阅《Amazon Key Management Service 开发人员指南》中的创建对称 KMS 密钥。
- `@overwrite_s3_backup_file` – 指示是否覆盖现有备份文件的值。
  - `0` – 不覆盖现有文件。此值是默认值。

如果文件已存在，则将 @overwrite_s3_backup_file 设置为 0 会返回错误。

  - 1 – 覆盖具有指定名称的现有文件，即使该文件不是备份文件也是如此。

- @type – 备份的类型。
  - DIFFERENTIAL – 进行差异备份。
  - FULL – 进行完整备份。此值是默认值。

  差异备份基于上一个完整备份。要使差异备份起作用，您不能在上一个完整备份和差异备份之间拍摄快照。如果您需要差异备份，但快照存在，请先进行另一个完整备份，然后再继续进行差异备份。

  您可以使用以下示例 SQL 查询来查找上一个完整备份或快照：

```
select top 1
database_name
,   backup_start_date
,   backup_finish_date
from    msdb.dbo.backupset
where   database_name='mydatabase'
and     type = 'D'
order by backup_start_date desc;
```

- @number_of_files – 备份将被划分（分块）成的文件数。最大文件数为 10。
  - 完整备份和差异备份都支持多文件备份。
  - 如果您输入值 1 或省略该参数，则会创建单个备份文件。

  请提供文件拥有的通用前缀，然后使用星号 (*) 为其添加后缀。星号可位于 S3 ARN 的 *file_name* 部分中的任意位置。星号由所生成文件中的一系列字母数字字符串替换，从 1-of-*number_of_files* 开始。

  例如，如果 S3 ARN 中的文件名为 backup*.bak 并且您已设置 @number_of_files=4，则生成的备份文件为 backup1-of-4.bak、backup2-of-4.bak、backup3-of-4.bak 和 backup4-of-4.bak。

  - 如果任意文件名已存在，并且 @overwrite_s3_backup_file 设置为 0，则会返回错误。
  - 在 S3 ARN 的 *file_name* 部分，多文件备份只能有一个星号。
  - 单文件备份可以在 S3 ARN 的 *file_name* 部分中包含任意数量的星号。星号不会从所生成的文件名中删除。

- @block_size – 块大小（以字节为单位），指定备份操作的物理块大小。有效值为 512、1024、2048、4096、8192、16384、32768 和 65536

- `@max_transfer_size` – 最大传输大小表示备份过程中每次 I/O 操作传输的数据量（以字节为单位）的上限。有效值是 65536 字节（64 KB）到 4194304 字节（4 MB）的倍数。
- `@buffer_count` – 用于备份过程的 I/O 缓冲区总数。

示例

Example 差异备份

```
exec msdb.dbo.rds_backup_database
@source_db_name='mydatabase',
@s3_arn_to_backup_to='arn:aws:s3:::mybucket/backup1.bak',
@overwrite_s3_backup_file=1,
@type='DIFFERENTIAL';
```

Example 使用加密的完整备份

```
exec msdb.dbo.rds_backup_database
@source_db_name='mydatabase',
@s3_arn_to_backup_to='arn:aws:s3:::mybucket/backup1.bak',
@kms_master_key_arn='arn:aws:kms:us-east-1:123456789012:key/AKIAIOSFODNN7EXAMPLE',
@overwrite_s3_backup_file=1,
@type='FULL';
```

Example 多文件备份

```
exec msdb.dbo.rds_backup_database
@source_db_name='mydatabase',
@s3_arn_to_backup_to='arn:aws:s3:::mybucket/backup*.bak',
@number_of_files=4;
```

Example 多文件差异备份

```
exec msdb.dbo.rds_backup_database
@source_db_name='mydatabase',
@s3_arn_to_backup_to='arn:aws:s3:::mybucket/backup*.bak',
@type='DIFFERENTIAL',
@number_of_files=4;
```

Example 使用加密的多文件备份

```
exec msdb.dbo.rds_backup_database
@source_db_name='mydatabase',
@s3_arn_to_backup_to='arn:aws:s3:::mybucket/backup*.bak',
@kms_master_key_arn='arn:aws:kms:us-east-1:123456789012:key/AKIAIOSFODNN7EXAMPLE',
@number_of_files=4;
```

Example 使用 S3 覆盖进行多文件备份

```
exec msdb.dbo.rds_backup_database
@source_db_name='mydatabase',
@s3_arn_to_backup_to='arn:aws:s3:::mybucket/backup*.bak',
@overwrite_s3_backup_file=1,
@number_of_files=4;
```

Example 使用块大小进行备份

```
exec msdb.dbo.rds_backup_database
@source_db_name='mydatabase',
@s3_arn_to_backup_to='arn:aws:s3:::mybucket/backup*.bak',
@block_size=512;
```

Example 使用 `@max_transfer_size` 和 `@buffer_count` 进行多文件备份

```
exec msdb.dbo.rds_backup_database
@source_db_name='mydatabase',
@s3_arn_to_backup_to='arn:aws:s3:::mybucket/backup*.bak',
@number_of_files=4,
@max_transfer_size=4194304,
@buffer_count=10;
```

Example 使用 @number_of_files 参数进行单文件备份

此示例生成一个名为 `backup*.bak` 的备份文件。

```
exec msdb.dbo.rds_backup_database
@source_db_name='mydatabase',
@s3_arn_to_backup_to='arn:aws:s3:::mybucket/backup*.bak',
@number_of_files=1;
```

还原数据库

要还原数据库，请调用 `rds_restore_database` 存储过程。还原任务完成且数据库打开后，Amazon RDS 将创建数据库的初始快照。

使用量

```
exec msdb.dbo.rds_restore_database
 @restore_db_name='database_name',
 @s3_arn_to_restore_from='arn:aws:s3:::bucket_name/file_name.extension',
 @with_norecovery=0|1,
 [@kms_master_key_arn='arn:aws:kms:region:account-id:key/key-id'],
        [@block_size=512|1024|2048|4096|8192|16384|32768|65536],
        [@max_transfer_size=n],
        [@buffer_count=n],
 [@type='DIFFERENTIAL|FULL'];
```

以下参数为必需参数：

- `@restore_db_name`– 要还原的数据库的名称。数据库名称是唯一的。如果存在同名的现有数据库，则无法还原数据库。
- `@s3_arn_to_restore_from` – 指示用于还原数据库的备份文件的 Amazon S3 前缀和名称的 ARN。
  - 对于单文件备份，请提供整个文件名。
  - 对于多文件备份，请提供文件拥有的通用前缀，然后使用星号 (*) 为其添加后缀。
  - 如果 `@s3_arn_to_restore_from` 为空，则返回以下错误消息：S3 ARN prefix cannot be empty (S3 ARN 前缀不能为空)。

以下参数对于差异还原是必需的，但对于完整还原是可选的：

- `@with_norecovery` – 用于还原操作的恢复子句。
  - 将它设置为 `0` 来使用 RECOVERY 进行还原。在这种情况下，在还原后，数据库将处于在线状态。
  - 将它设置为 1 来使用 NORECOVERY 进行还原。在这种情况下，在还原任务完成后，数据库将保持 RESTORING 状态。使用这种方法，您可以执行以后的差异还原。
  - 对于 DIFFERENTIAL 还原，请指定 `0` 或 `1`。
  - 对于 FULL 还原，此值默认为 `0`。

以下参数可选：

- `@kms_master_key_arn` - 如果您已对备份文件进行加密，则 KMS 密钥用于解密文件。

  指定 KMS 密钥时，将使用客户端加密。
- `@type` – 还原的类型。有效类型为 `DIFFERENTIAL` 和 `FULL`。默认值为 `FULL`。
- `@block_size` – 块大小（以字节为单位），指定备份操作的物理块大小。有效值为 512、1024、2048、4096、8192、16384、32768 和 65536
- `@max_transfer_size` – 最大传输大小表示备份过程中每次 I/O 操作传输的数据量（以字节为单位）的上限。有效值是 65536 字节（64 KB）到 4194304 字节（4 MB）的倍数。
- `@buffer_count` – 用于备份过程的 I/O 缓冲区总数。

> **Note**
>
> 对于差异还原，数据库必须处于 RESTORING 状态，或者必须已存在使用 NORECOVERY 还原的任务。
> 在数据库处于在线状态时，您无法还原以后的差异备份。
> 您无法为已具有使用 RECOVERY 的挂起还原任务的数据库提交还原任务。
> 多可用区实例不支持使用 NORECOVERY 的完整还原和差异还原。
> 在具有只读副本的多可用区实例上还原数据库类似于在多可用区实例上还原数据库。您无需执行任何其他操作，即可在副本上还原数据库。

示例

Example 单文件还原

```
exec msdb.dbo.rds_restore_database
@restore_db_name='mydatabase',
@s3_arn_to_restore_from='arn:aws:s3:::mybucket/backup1.bak';
```

Example 多文件还原

为避免还原多个文件时出错，请确保所有备份文件有相同的前缀且没有任何其他文件使用该前缀。

```
exec msdb.dbo.rds_restore_database
@restore_db_name='mydatabase',
```

```
@s3_arn_to_restore_from='arn:aws:s3:::mybucket/backup*';
```

Example 使用 RECOVERY 的完整数据库还原

以下三个示例执行相同的任务，即使用 RECOVERY 进行完整还原。

```
exec msdb.dbo.rds_restore_database
@restore_db_name='mydatabase',
@s3_arn_to_restore_from='arn:aws:s3:::mybucket/backup1.bak';
```

```
exec msdb.dbo.rds_restore_database
@restore_db_name='mydatabase',
@s3_arn_to_restore_from='arn:aws:s3:::mybucket/backup1.bak',
[@type='DIFFERENTIAL|FULL'];
```

```
exec msdb.dbo.rds_restore_database
@restore_db_name='mydatabase',
@s3_arn_to_restore_from='arn:aws:s3:::mybucket/backup1.bak',
@type='FULL',
@with_norecovery=0;
```

Example 使用加密进行完整数据库还原

```
exec msdb.dbo.rds_restore_database
@restore_db_name='mydatabase',
@s3_arn_to_restore_from='arn:aws:s3:::mybucket/backup1.bak',
@kms_master_key_arn='arn:aws:kms:us-east-1:123456789012:key/AKIAIOSFODNN7EXAMPLE';
```

Example 使用块大小进行还原

```
exec msdb.dbo.rds_restore_database
@restore_db_name='mydatabase',
@s3_arn_to_restore_from='arn:aws:s3:::mybucket/backup1.bak',
@block_size=512;
```

Example 使用 @max_transfer_size 和 @buffer_count 进行多文件还原

```
exec msdb.dbo.rds_restore_database
@restore_db_name='mydatabase',
```

```
@s3_arn_to_restore_from='arn:aws:s3:::mybucket/backup*',
@max_transfer_size=4194304,
@buffer_count=10;
```

Example 使用 NORECOVERY 进行完整数据库还原

```
exec msdb.dbo.rds_restore_database
@restore_db_name='mydatabase',
@s3_arn_to_restore_from='arn:aws:s3:::mybucket/backup1.bak',
@type='FULL',
@with_norecovery=1;
```

Example 使用 NORECOVERY 的差异还原

```
exec msdb.dbo.rds_restore_database
@restore_db_name='mydatabase',
@s3_arn_to_restore_from='arn:aws:s3:::mybucket/backup1.bak',
@type='DIFFERENTIAL',
@with_norecovery=1;
```

Example 使用 RECOVERY 的差异还原

```
exec msdb.dbo.rds_restore_database
@restore_db_name='mydatabase',
@s3_arn_to_restore_from='arn:aws:s3:::mybucket/backup1.bak',
@type='DIFFERENTIAL',
@with_norecovery=0;
```

还原日志

要还原日志，请调用 `rds_restore_log` 存储过程。

使用量

```
exec msdb.dbo.rds_restore_log
 @restore_db_name='database_name',
 @s3_arn_to_restore_from='arn:aws:s3:::bucket_name/log_file_name.extension',
 [@kms_master_key_arn='arn:aws:kms:region:account-id:key/key-id'],
 [@with_norecovery=0|1],
 [@stopat='datetime'],
 [@block_size=512|1024|2048|4096|8192|16384|32768|65536],
```

```
        [@max_transfer_size=n],
        [@buffer_count=n];
```

以下参数为必需参数：

- @restore_db_name – 要还原其日志的数据库的名称。
- @s3_arn_to_restore_from – 指示用于还原日志的日志文件的 Amazon S3 前缀和名称的 ARN。此文件可以具有任何扩展名，但通常使用 .trn。

  如果 @s3_arn_to_restore_from 为空，则返回以下错误消息：S3 ARN prefix cannot be empty (S3 ARN 前缀不能为空)。

以下参数可选：

- @kms_master_key_arn - 如果您已对日志进行加密，则 KMS 密钥用于解密日志。
- @with_norecovery – 用于还原操作的恢复子句。此值默认为 1。
  - 将它设置为 0 来使用 RECOVERY 进行还原。在这种情况下，在还原后，数据库将处于在线状态。在数据库处于在线状态时，您无法还原其他日志备份。
  - 将它设置为 1 来使用 NORECOVERY 进行还原。在这种情况下，在还原任务完成后，数据库将保持 RESTORING 状态。使用这种方法，您可以执行以后的日志还原。
- @stopat – 指定将数据库还原到其在指定的日期和时间的状态的值（采用日期时间格式）。仅将在指定日期和时间之前写入的事务日志记录应用于数据库。

  如果未指定此参数（它为 NULL），则会还原完整的日志。
- @block_size – 块大小（以字节为单位），指定备份操作的物理块大小。有效值为 512、1024、2048、4096、8192、16384、32768 和 65536
- @max_transfer_size – 最大传输大小表示备份过程中每次 I/O 操作传输的数据量（以字节为单位）的上限。有效值是 65536 字节（64 KB）到 4194304 字节（4 MB）的倍数。
- @buffer_count – 用于备份过程的 I/O 缓冲区总数。

> **Note**
>
> 对于日志还原，数据库必须处于 restoring 状态，或者必须已存在使用 NORECOVERY 还原的任务。
> 在数据库处于在线状态时，您无法还原日志备份。
> 您无法在已具有使用 RECOVERY 的挂起还原任务的数据库上提交日志还原任务。

多可用区实例不支持日志还原。

示例

Example 日志还原

```
exec msdb.dbo.rds_restore_log
@restore_db_name='mydatabase',
@s3_arn_to_restore_from='arn:aws:s3:::mybucket/mylog.trn';
```

Example 使用加密的日志还原

```
exec msdb.dbo.rds_restore_log
@restore_db_name='mydatabase',
@s3_arn_to_restore_from='arn:aws:s3:::mybucket/mylog.trn',
@kms_master_key_arn='arn:aws:kms:us-east-1:123456789012:key/AKIAIOSFODNN7EXAMPLE';
```

Example 使用 NORECOVERY 的日志还原

以下两个示例执行相同的任务，即使用 NORECOVERY 进行日志还原。

```
exec msdb.dbo.rds_restore_log
@restore_db_name='mydatabase',
@s3_arn_to_restore_from='arn:aws:s3:::mybucket/mylog.trn',
@with_norecovery=1;
```

```
exec msdb.dbo.rds_restore_log
@restore_db_name='mydatabase',
@s3_arn_to_restore_from='arn:aws:s3:::mybucket/mylog.trn';
```

Example 使用块大小进行还原

```
exec msdb.dbo.rds_restore_log
@restore_db_name='mydatabase',
@s3_arn_to_restore_from='arn:aws:s3:::mybucket/mylog.trn',
@block_size=512;
```

Example 使用 RECOVERY 的日志还原

```
exec msdb.dbo.rds_restore_log
```

```
@restore_db_name='mydatabase',
@s3_arn_to_restore_from='arn:aws:s3:::mybucket/mylog.trn',
@with_norecovery=0;
```

Example 使用 STOPAT 子句的日志还原

```
exec msdb.dbo.rds_restore_log
@restore_db_name='mydatabase',
@s3_arn_to_restore_from='arn:aws:s3:::mybucket/mylog.trn',
@with_norecovery=0,
@stopat='2019-12-01 03:57:09';
```

完成数据库还原

如果数据库上的上一个还原任务是使用 `@with_norecovery=1` 执行的，则数据库现在处于 RESTORING 状态。使用 `rds_finish_restore` 存储过程打开此数据库以执行常规操作。

使用量

```
exec msdb.dbo.rds_finish_restore @db_name='database_name';
```

> **Note**
> 要使用此方法，数据库必须处于 RESTORING 状态，且没有任何挂起的还原任务。
> 多可用区实例不支持 `rds_finish_restore` 过程。
> 要完成数据库还原，请使用主登录名。或使用最近使用 NORECOVERY 还原数据库或日志的用户登录名。

处理部分还原的数据库

删除部分还原的数据库

要删除部分还原的数据库（保留为 RESTORING 状态），请使用 `rds_drop_database` 存储过程。

```
exec msdb.dbo.rds_drop_database @db_name='database_name';
```

> **Note**
> 您无法为已具有挂起的还原或完成还原任务的数据库提交 DROP 数据库请求。

要删除数据库，请使用主登录名。或使用最近使用 NORECOVERY 还原数据库或日志的用户登录名。

部分还原的数据库的快照还原和时间点恢复行为

在快照还原和时间点恢复期间，将从目标实例中删除源实例中的部分还原的数据库（保留为 RESTORING 状态）。

取消任务

要取消备份或还原任务，请调用 `rds_cancel_task` 存储过程。

> **Note**
> 您无法取消 FINISH_RESTORE 任务。

使用量

```
exec msdb.dbo.rds_cancel_task @task_id=ID_number;
```

以下参数是必需参数：

- `@task_id`– 要取消的任务的 ID。可以通过调用 `rds_task_status` 获取任务 ID。

跟踪任务的状态

要跟踪备份和还原任务的状态，请调用 `rds_task_status` 存储过程。如果您未提供任何参数，则存储过程将返回所有任务的状态。任务的状态约每 2 分钟更新一次。任务历史记录保留 36 天。

使用量

```
exec msdb.dbo.rds_task_status
 [@db_name='database_name'],
 [@task_id=ID_number];
```

以下参数可选：

- `@db_name`– 要显示其任务状态的数据库的名称。

- `@task_id`– 要显示其任务状态的任务的 ID。

示例

Example 列出特定任务的状态

```
exec msdb.dbo.rds_task_status @task_id=5;
```

Example 列出特定数据库和任务的状态

```
exec msdb.dbo.rds_task_status
@db_name='my_database',
@task_id=5;
```

Example 列出特定数据库上的所有任务及其状态

```
exec msdb.dbo.rds_task_status @db_name='my_database';
```

Example 列出当前实例上的所有任务及其状态

```
exec msdb.dbo.rds_task_status;
```

响应

`rds_task_status` 存储过程返回以下列。

列	描述
`task_id`	任务的 ID。

`task_type`	任务类型取决于输入参数，如下所示：
	• 对于备份任务：
	  • BACKUP_DB – 完整数据库备份
	  • BACKUP_DB_DIFFERENTIAL – 差异数据库备份

<table>
<tr><th>列</th><th>描述</th></tr>
<tr><td></td><td>对于还原任务：
RESTORE_DB – 使用 RECOVERY 的完整数据库还原
RESTORE_DB_NORECOVERY – 使用 NORECOVERY 的完整数据库还原
RESTORE_DB_DIFFERENTIAL – 使用 RECOVERY 的差异数据库还原
RESTORE_DB_DIFFERENTIAL_NORECOVERY – 使用 NORECOVERY 的差异数据库还原
RESTORE_DB_LOG – 使用 RECOVERY 的日志还原
RESTORE_DB_LOG_NORECOVERY – 使用 NORECOVERY 的日志还原
对于完成还原的任务：
FINISH_RESTORE – 完成还原并打开数据库
在完成以下还原任务时打开数据库后，Amazon RDS 创建数据库的初始快照：
RESTORE_DB
RESTORE_DB_DIFFERENTIAL
RESTORE_DB_LOG
FINISH_RESTORE</td></tr>
<tr><td><code>database_name</code></td><td>与任务关联的数据库的名称。</td></tr>
<tr><td><code>% complete</code></td><td>用百分比值表示的任务进度。</td></tr>
</table>

列	描述
duration (mins)	在任务上花费的时间 (以分钟为单位)。
lifecycle	任务的状态。有以下可能状态： • CREATED – 一旦您调用 rds_backup_database 或 rds_restore_database ，就会创建任务并将状态设置为 CREATED。 • IN_PROGRESS – 备份或还原任务开始后，状态将设置为 IN_PROGRESS 。要将状态从 CREATED 更改为 IN_PROGRESS 可能需要最多 5 分钟的时间。 • SUCCESS – 备份或还原任务完成之后，状态将设置为 SUCCESS。 • ERROR – 如果备份或还原任务失败，状态将设置为 ERROR。有关错误的更多信息，请参阅 task_info 列。 • CANCEL_REQUESTED – 一旦您调用 rds_cancel_task ，任务的状态就会设置为 CANCEL_REQUESTED 。 • CANCELLED – 在成功取消任务后，任务的状态将设置为 CANCELLED 。
task_info	有关任务的其他信息。 如果在备份或还原数据库时出错，则此列将包含有关错误的信息。有关可能的错误列表以及缓解策略，请参阅 故障排除。
last_updated	上次更新任务状态的日期和时间。状态每隔 5% 的进度更新一次。
created_at	任务的创建日期和时间。
S3_object_arn	指示要备份或还原的文件的 Amazon S3 前缀和名称的 ARN。

列	描述
overwrite_s3_backup_file	调用备份任务时指定的 @overwrite_s3_backup_file 参数的值。有关更多信息，请参阅 备份数据库。
KMS_master_key_arn	用于加密 (对于备份) 和解密 (对于还原) 的 KMS 密钥的 ARN。
filepath	不适用于本机备份和还原任务。
overwrite_file	不适用于本机备份和还原任务。

压缩备份文件

要节省 Amazon S3 存储桶中的空间，您可以压缩备份文件。有关压缩备份文件的更多信息，请参阅 Microsoft 文档中的备份压缩。

以下数据库版本支持压缩您的备份文件：

- Microsoft SQL Server 企业版
- Microsoft SQL Server 标准版

要为备份文件启用压缩，请运行以下代码：

```
exec rdsadmin.dbo.rds_set_configuration 'S3 backup compression', 'true';
```

要为备份文件禁用压缩，请运行以下代码：

```
exec rdsadmin.dbo.rds_set_configuration 'S3 backup compression', 'false';
```

故障排除

以下是您使用本地备份和还原时可能会遇到的问题。

问题	故障排除建议
数据库备份/恢复选项尚未启用或启用中。请稍后重试。	请确保您已将 `SQLSERVER_BACKUP_RESTORE` 选项添加到与数据库实例关联的数据库选项组。有关更多信息，请参阅 添加本机备份和还原选项。
对象“*rds_backup_database*”（数据库“msdb”，架构“dbo”）上的 EXECUTE 权限遭拒绝。	确保在执行存储过程时使用主用户。如果您以主用户身份登录后仍遇到此错误，则可能是由于管理员用户权限不符。要重置主用户，请使用 Amazon Web Services Management Console。请参阅重置 Amazon RDS for SQL Server 主用户的 db_owner 角色成员资格。
对象“*rds_restore_database*”（数据库“msdb”，架构“dbo”）上的 EXECUTE 权限遭拒绝。	确保在执行存储过程时使用主用户。如果您以主用户身份登录后仍遇到此错误，则可能是由于管理员用户权限不符。要重置主用户，请使用 Amazon Web Services Management Console。请参阅重置 Amazon RDS for SQL Server 主用户的 db_owner 角色成员资格。
拒绝访问	备份或还原进程无法访问备份文件。这通常由类似于以下的问题导致： • 引用不正确存储桶。使用不正确的格式引用存储桶。引用文件名但未使用 ARN。 • 存储桶文件的权限不正确。例如，如果文件由正在尝试访问它的其他账户创建，请添加正确的权限。 • IAM 策略不正确或不完整。您的 IAM 角色必须包含所有的必要元素，例如，包括正确的版本。使用本机备份和还原导入和导出 SQL Server 数据库中重点介绍了这些内容。
<edition_name> 版本不支持带压缩的备份数据库	仅 Microsoft SQL Server 企业版和标准版支持压缩备份文件。 有关更多信息，请参阅“压缩备份文件”。
请使用正确的类型重新发布任务并覆盖属性	如果您尝试备份数据库并提供了已存在文件的名称，但将覆盖策略设置为 false，则保存操作会失败。要修复该错误，请提供不存在的文件名，或者将覆盖属性设置为 true。

问题	故障排除建议
	有关更多信息，请参阅“备份数据库”。 还有可能是您希望还原数据库，但意外调用了 `rds_backup_database` 存储过程。在这种情况下，请改为调用 `rds_restore_database` 存储过程。 有关更多信息，请参阅“还原数据库”。 如果您希望还原数据库并调用 `rds_restore_database` 存储过程，请确保您提供了有效的备份文件名称。 有关更多信息，请参阅“使用本机备份和还原”。
请指定与 RDS 实例位于同一区域的存储桶	您不能备份到与您的 Amazon RDS 数据库实例不同的Amazon区域中的某个 Amazon S3 存储桶或从该存储桶中进行还原。您可以使用 Amazon S3 复制将备份文件复制到正确的Amazon区域。 有关更多信息，请参阅 Amazon S3 文档中的跨区域复制。
指定的存储桶不存在	验证您使用正确格式为存储桶和文件提供了正确的 ARN。 有关更多信息，请参阅“使用本机备份和还原”。
还原”任务无法从 10 个以上的备份文件中恢复。请减少匹配的文件数量，然后重试。	减少您尝试从中还原的文件数。如有必要，您可以让每个文件变得更大。
数据库“*database_name* ”已存在。不允许存在两个仅大小写或重音不同的数据库。选择其他数据库名称。	如果存在同名的现有数据库，则无法还原数据库。数据库名称是唯一的。

使用其他方法导入和导出 SQL Server 数据

接下来，您可以找到有关使用快照将 Microsoft SQL Server 数据导入到 Amazon RDS 中的信息。还可以找到有关使用快照从运行 SQL Server 的 RDS 数据库实例中导出数据的信息。

如果您的方案支持此方法，则可使用本机备份和还原功能更轻松地将数据移入和移出 Amazon RDS。有关更多信息，请参阅使用本机备份和还原导入和导出 SQL Server 数据库。

> **Note**
>
> Amazon RDS for Microsoft SQL Server 不支持将数据导入 `msdb` 数据库。

使用快照将数据导入 RDS for SQL Server

使用快照将数据导入 SQL Server 数据库实例

1. 创建数据库实例。有关更多信息，请参阅“创建 Amazon RDS 数据库实例”。
2. 阻止应用程序访问目标数据库实例。

   如果您在导入数据时阻止应用程序访问数据库实例，数据传输速度会更快。此外，您无需担心在加载数据时会因其他应用程序无法写入到数据库实例而引起冲突。如果因出现问题而必须要回滚到之前的数据库快照，则丢失的唯一更改就是导入的数据。在解决问题后，您可以再次导入这些数据。

   有关控制数据库实例访问权限的信息，请参阅 使用安全组控制访问权限。
3. 创建目标数据库快照。

   如果目标数据库已存有数据，我们建议您先为数据库创建快照，然后再导入数据。如果数据导入过程出现问题或您想要放弃更改，您可以使用此快照将数据库还原到先前的状态。有关数据库快照的信息，请参阅 为 Amazon RDS 的单可用区数据库实例创建数据库快照。

   > **Note**
   >
   > 获取数据库快照时，数据库的 I/O 操作会在备份进行期间暂停一会（以毫秒为单位）。
4. 禁用目标数据库的自动备份。

   禁用目标数据库实例上的自动备份将会提高导入数据时的性能，因为 Amazon RDS 在禁用自动备份后不会记录事务。但需要考虑以下问题。执行时间点恢复时需要使用自动备份。因此，在导入数

据时，无法将数据库还原到特定时间点。此外，除非您选择保留先前在数据库实例上创建的所有自动备份，否则系统会将其清除。

选择保留自动备份有助于防止意外删除数据。Amazon RDS 还会保存数据库实例属性以及每个自动备份以便进行恢复。利用此选项，可将已删除的数据库实例还原到备份保留期内的指定时间点，即使在删除它之后也是如此。由于自动备份是用于有效数据库实例的，它们将在特定备份时段结束后自动被删除。

您也可使用先前的快照来恢复数据库，并且您创建的所有快照仍然可用。有关自动备份的信息，请参阅 备份简介。

5. 禁用外键约束 (如适用)。

   如果需要禁用外键约束，可以使用以下脚本。

```
--Disable foreign keys on all tables
    DECLARE @table_name SYSNAME;
    DECLARE @cmd NVARCHAR(MAX);
    DECLARE table_cursor CURSOR FOR SELECT name FROM sys.tables;

    OPEN table_cursor;
    FETCH NEXT FROM table_cursor INTO @table_name;

    WHILE @@FETCH_STATUS = 0 BEGIN
      SELECT @cmd = 'ALTER TABLE '+QUOTENAME(@table_name)+' NOCHECK CONSTRAINT
 ALL';
      EXEC (@cmd);
      FETCH NEXT FROM table_cursor INTO @table_name;
    END

    CLOSE table_cursor;
    DEALLOCATE table_cursor;

    GO
```

6. 删除索引 (如果适用)。
7. 禁用触发器 (如果适用)。

   如果您需要禁用触发器，则可使用以下脚本执行该操作。

```
--Disable triggers on all tables
    DECLARE @enable BIT = 0;
```

```
    DECLARE @trigger SYSNAME;
    DECLARE @table SYSNAME;
    DECLARE @cmd NVARCHAR(MAX);
    DECLARE trigger_cursor CURSOR FOR SELECT trigger_object.name trigger_name,
     table_object.name table_name
    FROM sysobjects trigger_object
    JOIN sysobjects table_object ON trigger_object.parent_obj = table_object.id
    WHERE trigger_object.type = 'TR';

    OPEN trigger_cursor;
    FETCH NEXT FROM trigger_cursor INTO @trigger, @table;

    WHILE @@FETCH_STATUS = 0 BEGIN
      IF @enable = 1
         SET @cmd = 'ENABLE ';
      ELSE
         SET @cmd = 'DISABLE ';

      SET @cmd = @cmd + ' TRIGGER dbo.'+QUOTENAME(@trigger)+' ON
 dbo.'+QUOTENAME(@table)+' ';
      EXEC (@cmd);
      FETCH NEXT FROM trigger_cursor INTO @trigger, @table;
    END

    CLOSE trigger_cursor;
    DEALLOCATE trigger_cursor;

    GO
```

8. 查询源 SQL Server 实例，获取要导入到目标数据库实例的所有登录名。

   SQL Server 将登录名和密码存储在 `master` 数据库中。由于 Amazon RDS 未授予访问 `master` 数据库的权限，因此您无法直接将登录名和密码导入目标数据库实例。您必须在源 SQL Server 实例上查询 `master` 数据库以生成数据定义语言 (DDL) 文件。此文件应包括要添加到目标数据库实例的所有登录名和密码。此文件还应包括要传输的角色成员资格和权限。

   有关查询 `master` 数据库的信息，请参阅 Microsoft 知识库中的 [Transfer logins and passwords between instances of SQL Server](#)。

   该脚本输出是可在目标数据库实例上运行的另一个脚本。知识库文章中的脚本包含以下代码：

   ```
   p.type IN
   ```

在 p.type 出现的所有位置，改用以下代码：

```
p.type = 'S'
```

9. 使用导入数据中的方法导入数据。
10. 授予应用程序访问目标数据库实例的权限。

    完成数据导入后，您可以向导入期间被阻止的应用程序授予访问数据库实例的权限。有关控制数据库实例访问权限的信息，请参阅 使用安全组控制访问权限。
11. 启用目标数据库实例的自动备份。

    有关自动备份的信息，请参阅 备份简介。
12. 启用外键约束。

    如果您之前禁用了外键约束，现在可通过以下脚本重新启用。

```
--Enable foreign keys on all tables
    DECLARE @table_name SYSNAME;
    DECLARE @cmd NVARCHAR(MAX);
    DECLARE table_cursor CURSOR FOR SELECT name FROM sys.tables;

    OPEN table_cursor;
    FETCH NEXT FROM table_cursor INTO @table_name;

    WHILE @@FETCH_STATUS = 0 BEGIN
      SELECT @cmd = 'ALTER TABLE '+QUOTENAME(@table_name)+' CHECK CONSTRAINT ALL';
      EXEC (@cmd);
      FETCH NEXT FROM table_cursor INTO @table_name;
    END

    CLOSE table_cursor;
    DEALLOCATE table_cursor;
```

13. 启用索引 (如果适用)。
14. 启用触发器 (如果适用)。

    如果您之前禁用了触发器，现在可以通过以下脚本启用触发器。

```
--Enable triggers on all tables
    DECLARE @enable BIT = 1;
```

```
    DECLARE @trigger SYSNAME;
    DECLARE @table SYSNAME;
    DECLARE @cmd NVARCHAR(MAX);
    DECLARE trigger_cursor CURSOR FOR SELECT trigger_object.name trigger_name,
     table_object.name table_name
    FROM sysobjects trigger_object
    JOIN sysobjects table_object ON trigger_object.parent_obj = table_object.id
    WHERE trigger_object.type = 'TR';

    OPEN trigger_cursor;
    FETCH NEXT FROM trigger_cursor INTO @trigger, @table;

    WHILE @@FETCH_STATUS = 0 BEGIN
      IF @enable = 1
         SET @cmd = 'ENABLE ';
      ELSE
         SET @cmd = 'DISABLE ';

      SET @cmd = @cmd + ' TRIGGER dbo.'+QUOTENAME(@trigger)+' ON
 dbo.'+QUOTENAME(@table)+' ';
      EXEC (@cmd);
      FETCH NEXT FROM trigger_cursor INTO @trigger, @table;
    END

    CLOSE trigger_cursor;
    DEALLOCATE trigger_cursor;
```

导入数据

Microsoft SQL Server Management Studio 是一种包含在所有 Microsoft SQL Server 版本 (Express 版除外) 中的图像化 SQL Server 客户端。您可以从 Microsoft 上免费下载 SQL Server Management Studio Express。要找到此下载，请访问 Microsoft 网站。

> **Note**
>
> SQL Server Management Studio 应用程序只能在 Windows 环境中运行。

SQL Server Management Studio 包括能够帮助将数据导入 SQL Server 数据库实例的以下工具：

- 生成和发布脚本向导

- 导入和导出向导
- 批量复制

生成和发布脚本向导

生成和发布脚本向导可创建一个包含数据库架构或数据本身，或兼具两者的脚本。您可以在本地 SQL Server 部署中为数据库生成脚本。然后，您可以运行该脚本以将其包含的信息传输到 Amazon RDS 数据库实例。

> **Note**
> 对于 1GiB 或更大的数据库，只编写数据库架构的脚本会更高效。然后，您可以使用导入和导出向导，或 SQL Server 的批量复制功能来传输数据。

有关生成和发布脚本向导的详细信息，请参阅 [Microsoft SQL Server 文档]。

在此向导中，请特别注意设置脚本选项页面中的高级选项，确保已选中脚本需要包含的所有选项。例如，默认情况下，脚本不包含数据库触发器。

生成并保存脚本后，您可以使用 SQL Server Management Studio 连接数据库实例，然后运行脚本。

导入和导出向导

导入和导出向导工具可创建一种特殊的集成服务包，您可以使用此集成服务包将数据从本地 SQL Server 数据库复制到目标数据库实例。此向导可筛选要将哪些表格甚至是表格内的哪些元组复制到目标数据库实例。

> **Note**
> 导入和导出向导在用于大型数据集时非常有效，但是就从本地部署中远程导出数据而言，这可能不是最快的方式。要采用更快的方式，请考虑 SQL Server 批量复制功能。

有关导入和导出向导的详细信息，请参阅 [Microsoft SQL Server 文档]。

在此向导中，请在 Choose a Destination (选择目标) 页面上执行以下操作：

- 对于 Server Name (服务器名称)，键入数据库实例的终端节点名称。

- 对于服务器身份验证模式，请选择 Use SQL Server Authentication。
- 对于 User name (用户名) 和 Password (密码)，请键入您为数据库实例创建的主用户凭证。

批量复制

SQL Server 批量复制功能是一种将数据从源数据库复制到数据库实例的高效工具。批量复制可将指定的数据写入一个数据文件，如 ASCII 文件。然后您可以再次运行批量复制，将此文件内容写入目标数据库实例。

本部分使用了 bcp 实用工具，所有版本的 SQL Server 都包含此工具。有关批量导入和导出操作的详细信息，请参阅 Microsoft SQL Server 文档。

> **Note**
>
> 使用批量复制前，您必须先将数据库架构导入目标数据库实例。本主题前面介绍的生成和发布脚本向导这一工具非常适合用于该操作。

以下命令用于连接到本地 SQL Server 实例。它在您现有 SQL Server 部署的 C:\ 根目录中生成指定表的制表符定界文件。指定表时需使用完全限定名，文本文件的名称与复制的表相同。

```
bcp dbname.schema_name.table_name out C:\table_name.txt -n -S localhost -U username -
P password -b 10000
```

上述代码包含以下选项：

- -n 指定，批量复制将使用待复制数据的本机数据类型。
- -S 指定 bcp 实用工具连接到的 SQL Server 实例。
- -U 指定要登录到 SQL Server 实例的账户的用户名。
- -P 为 指定的用户指定密码。-U
- -b 指定每批导入数据的行数。

> **Note**
>
> 可能有对导入情况来说很重要的其他参数。例如，您可能需要与身份值有关的 -E 参数。有关更多信息，请参阅 Microsoft SQL Server 文档中的 bcp 实用工具的命令行语法的完整说明。

例如，假设某一数据库名为 store，使用了默认架构 dbo 且包含一个名为 customers 的表。密码为 admin 的用户账户 insecure 将 customers 表中的 10000 行复制到名为 customers.txt 的文件中。

```
bcp store.dbo.customers out C:\customers.txt -n -S localhost -U admin -P insecure -b
 10000
```

生成数据文件后，可以使用类似的命令将数据上传到数据库实例。事先在目标数据库实例上创建数据库和架构。然后使用 in 参数指定输入文件，而不是使用 out 指定输出文件。您需要指定数据库实例的终端节点，而不是使用 localhost 指定本地 SQL Server 实例。如果您使用的端口不是 1433 端口，也需要执行该指定操作。用户名和密码是数据库实例的主用户名和密码。语法如下所示。

```
bcp dbname.schema_name.table_name
     in C:\table_name.txt -n -S endpoint,port -U master_user_name -
P master_user_password -b 10000
```

为继续上述示例，我们假设主用户名是 admin，密码为 insecure。数据库实例的终端节点是 rds.ckz2kqd4qsn1.us-east-1.rds.amazonaws.com，且您使用端口 4080。命令如下所示。

```
bcp store.dbo.customers in C:\customers.txt -n -S rds.ckz2kqd4qsn1.us-
east-1.rds.amazonaws.com,4080 -U admin -P insecure -b 10000
```

Note

作为安全最佳实践，请指定除此处所示提示以外的密码。

从 RDS for SQL Server 中导出数据

您可以选择以下选项之一来从 RDS for SQL Server 数据库实例中导出数据：

- 使用完整备份文件 (.bak) 的本机数据库备份 – 使用 .bak 文件备份数据库的过程已得到大幅优化，因此，它通常是最快的导出数据的方法。有关更多信息，请参阅“使用本机备份和还原导入和导出 SQL Server 数据库”。
- SQL Server 导入和导出向导 – 有关更多信息，请参阅 SQL Server 导入和导出向导。
- SQL Server 生成和发布脚本向导与 bcp 实用工具 – 有关更多信息，请参阅 SQL Server 生成和发布脚本向导与 bcp 实用工具。

SQL Server 导入和导出向导

您可以使用 SQL Server 导入和导出向导将一个或多个表、视图或查询从 RDS for SQL Server 数据库实例复制到另一个数据存储。如果目标数据存储不是 SQL Server，这将是最佳选择。有关更多信息，请参阅 SQL Server 文档中的 SQL Server 导入和导出向导。

SQL Server 导入和导出向导是 Microsoft SQL Server Management Studio 的一部分。此图形化 SQL Server 客户端包含在所有 Microsoft SQL Server 版本（Express 版除外）中。SQL Server Management Studio 应用程序只能在 Windows 环境中运行。您可以从 Microsoft 上免费下载 SQL Server Management Studio Express。要找到此下载，请访问 Microsoft 网站。

使用 SQL Server 导入和导出向导来导出数据

1. 在 SQL Server Management Studio 中，连接到 RDS for SQL Server 数据库实例。有关如何执行该操作的详细信息，请参阅连接到 Microsoft SQL Server 数据库实例。
2. 在 Object Explorer 中，展开 Databases (数据库)，打开源数据库的上下文（右键单击）菜单，选择 Tasks (任务)，然后选择 Export Data (导出数据)。此时将显示向导。
3. 在 Choose a Data Source (选择数据源) 页面上，执行以下操作：

    a. 对于 Data source (数据源)，选择 **SQL Server Native Client 11.0**。
    b. 确认服务器名称框中显示 RDS for SQL Server 数据库实例的终端节点。
    c. 选择 Use SQL Server Authentication (使用 SQL Server 身份验证)。对于 用户名 和 密码，键入您的数据库实例的主用户名和密码。
    d. 确认 Database (数据库) 框中显示要从中导出数据的数据库。
    e. 选择下一步。
4. 在 Choose a Destination (选择目标) 页面上，执行以下操作：

    a. 对于 Destination (目标)，选择 **SQL Server Native Client 11.0**。

    > **Note**
    >
    > 提供了其他目标数据源。这些包括 .NET Framework 数据提供程序、OLE DB 提供程序、SQL Server Native Client 提供程序、ADO.NET 提供程序、Microsoft Office Excel、Microsoft Office Access 以及平面文件源。如果您选择这些数据源之一作为目标，请跳过步骤 4 的其余部分。有关接下来要提供的连接信息的详细信息，请参阅 SQL Server 文档中的选择目标。

    b. 对于 Server name (服务器名称)，键入目标 SQL Server 数据库实例的服务器名称。

c. 选择适当的身份验证类型。如有必要，键入用户名和密码。

d. 对于 Database (数据库)，选择目标数据库的名称，或选择 New (新建) 以创建一个包含已导出数据的新数据库。

如果选择新建，请参阅 SQL Server 文档中的创建数据库，以了解要提供的数据库信息。

e. 选择下一步。

5. 在 Table Copy or Query (表复制或查询) 页面上，选择 Copy data from one or more tables or views (从一个或多个表或视图复制数据) 或 Write a query to specify the data to transfer (编写查询以指定要传输的数据)。选择下一步。

6. 如果您选择 Write a query to specify the data to transfer (编写查询以指定要传输的数据)，则将看到 Provide a Source Query (提供源查询) 页面。键入或贴入 SQL 查询，然后选择 Parse (分析) 以进行验证。在验证查询后，选择 Next (下一步)。

7. 在 Select Source Tables and Views (选择源表和视图) 页面上，执行以下操作：

   a. 选择要导出的表和视图，或确认已选择您提供的查询。

   b. 选择 Edit Mappings (编辑映射) 并指定数据库和列映射信息。有关详细信息，请参阅 SQL Server 文档中的列映射。

   c. (可选) 若要查看要导出的数据的预览，请选择表、视图或查询，然后选择 Preview。

   d. 选择下一步。

8. 在 Run Package (运行程序包) 页面上，确认已选择 Run immediately (立即运行)。选择下一步。

9. 在 Complete the Wizard (完成向导) 页面上，确认数据导出详细信息是预期的详细信息。选择 Finish (结束)。

10. 在 The execution was successful (执行已成功) 页面上，单击 Close (关闭)。

SQL Server 生成和发布脚本向导与 bcp 实用工具

您可以使用 SQL Server 生成和发布脚本向导来为整个数据库或仅为所选对象创建脚本。可在目标 SQL Server 数据库实例上运行这些脚本以重新创建已编写脚本的对象。然后，可以使用 bcp 实用工具将选定对象的数据批量导出到目标数据库实例。若要在两个 SQL Server 数据库实例之间移动整个数据库 (包括表以外的对象) 或大量数据，这将是最佳选择。有关 bcp 命令行语法的完整说明，请参阅 Microsoft SQL Server 文档中的 bcp 实用工具。

SQL Server 生成和发布脚本向导是 Microsoft SQL Server Management Studio 的一部分。此图形化 SQL Server 客户端包含在所有 Microsoft SQL Server 版本（Express 版除外）中。SQL Server

Management Studio 应用程序只能在 Windows 环境中运行。您可以从 Microsoft 上免费下载 SQL Server Management Studio Express。

使用 SQL Server 生成和发布脚本向导与 bcp 实用工具来导出数据

1. 在 SQL Server Management Studio 中，连接到 RDS for SQL Server 数据库实例。有关如何执行该操作的详细信息，请参阅 连接到 Microsoft SQL Server 数据库实例。
2. 在 Object Explorer 中，展开 Databases (数据库) 节点并选择要编写脚本的数据库。
3. 按照 SQL Server 文档中的生成和发布脚本向导中的说明，创建脚本文件。
4. 在 SQL Server Management Studio 中，连接到目标 SQL Server 数据库实例。
5. 在 Object Explorer（对象资源管理器）中选择目标 SQL Server 数据库实例后，在 File（文件）菜单上选择 Open（打开），选择 File（文件），然后打开脚本文件。
6. 如果您已编写了整个数据库的脚本，请查看脚本中的 CREATE DATABASE 语句。请确保将在所需位置使用所需参数创建数据库。有关更多信息，请参阅 SQL Server 文档中的 CREATE DATABASE。
7. 如果在脚本中创建数据库用户，请检查目标数据库实例中是否有这些用户的服务器登录名。如果没有登录名，请为这些用户创建登录名；否则，用于创建数据库用户的脚本命令将失败。有关更多信息，请参阅 SQL Server 文档中的创建登录名。
8. 选择 SQL 编辑器菜单上的 !Execute 以运行脚本文件并创建数据库对象。完成脚本后，验证是否存在预期的所有数据库对象。
9. 使用 bcp 实用工具将数据从 RDS for SQL Server 数据库实例导出到文件。打开命令提示符并键入以下命令。

```
bcp database_name.schema_name.table_name out data_file -n -S aws_rds_sql_endpoint -
U username -P password
```

上述代码包含以下选项：

- table_name 是您已在目标数据库中重新创建且现在需要填充数据的其中一个表的名称。
- data_file 是要创建的数据文件的完整路径和名称。
- -n 指定，批量复制将使用待复制数据的本机数据类型。
- -S 指定要从中导出数据的 SQL Server 数据库实例。
- -U 指定在连接到 SQL Server 数据库实例时要使用的用户名。
- -P 为 -U 指定的用户指定密码。

下面显示了示例命令。

```
bcp world.dbo.city out C:\Users\JohnDoe\city.dat -n -S sql-jdoe.1234abcd.us-
west-2.rds.amazonaws.com,1433 -U JohnDoe -P ClearTextPassword
```

重复此步骤，直到您获得要导出的所有表的数据文件。

10. 按照 SQL Server 文档中的批量导入数据基本指南中的说明，准备目标数据库实例以便批量导入数据。
11. 在考虑 SQL Server 文档中的关于批量导入和批量导出操作中讨论的性能和其他问题后，决定要使用的批量导入方法。
12. 从使用 bcp 实用工具创建的数据文件中批量导入数据。为此，请按照 SQL Server 文档中的使用 bcp 实用工具导入和导出批量数据或使用 BULK INSERT 或 OPENROWSET(BULK...) 导入批量数据中的说明进行操作，具体取决于您在步骤 11 中所做的决定。

在 Amazon RDS 中使用 Microsoft SQL Server 的只读副本

一般使用只读副本配置 Amazon RDS 数据库实例之间的复制。有关只读副本的一般信息，请参阅使用数据库实例只读副本。

在本部分中，您可以找到有关使用 Amazon RDS for SQL Server 上的只读副本的特定信息。

- 将数据库用户和对象与 SQL Server 只读副本同步
- 排除 SQL Server 只读副本问题

为 SQL Server 配置只读副本

数据库实例可充当复制的源实例之前，您必须在源数据库实例上启用自动备份。为此，请将备份保留期设定为非 0 值。设置此类部署还会强制启用自动备份。

创建 SQL Server 只读副本不需要中断源数据库实例。Amazon RDS 为源数据库实例和只读副本设置必要的参数和权限，而不会中断任务服务。将创建源数据库实例的快照，此快照将变为只读副本。在删除只读副本时不会发生中断。

您可以从一个源数据库实例创建最多 15 个只读副本。为了有效地进行复制，我们建议您为每个只读副本配置与源数据库实例数量相同的计算和存储资源。如果扩展源数据库实例，则还应扩展只读副本。

源数据库实例及其所有只读副本的 SQL Server 数据库引擎版本必须相同。无论维护时段如何，Amazon RDS 都会在升级只读副本后立即升级主要副本。有关升级数据库引擎版本的更多信息，请参阅升级 Microsoft SQL Server 数据库引擎。

为了让只读副本可以接收和应用源数据库实例的更改，它应具有足够的计算和存储资源。如果只读副本达到计算、网络或存储资源容量，则只读副本会停止接收或应用来自其源实例的更改。您可以单独修改某个只读副本的存储和 CPU 资源，不受其源实例及其他只读副本的影响。

有关如何创建只读副本的更多信息，请参阅创建只读副本。

SQL Server 的只读副本限制

以下限制适用于 Amazon RDS 上的 SQL Server 只读副本：

- 只读副本仅在 SQL Server 企业版 (EE) 引擎中可用。
- 只读副本可用于 SQL Server 版本 2016–2022。
- 您可以从一个源数据库实例创建最多 15 个只读副本。当您的源数据库实例的只读副本超过 5 个时，复制可能会滞后。

- 只读副本仅适用于在具有四个或更多 vCPU 的数据库实例类上运行的数据库实例。
- 只读副本支持最多 100 个数据库，具体取决于实例类类型和可用性模式。您必须在源数据库实例上创建数据库，才能自动将数据库复制到只读副本。您无法选择要复制的单独数据库。有关更多信息，请参阅 Microsoft SQL Server 数据库实例的限制。
- 您无法从只读副本中删除数据库。要删除数据库，请使用 `rds_drop_database` 存储过程将其从源数据库实例中删除。有关更多信息，请参阅 删除 Amazon RDS for Microsoft SQL Server 数据库实例中的数据库。
- 如果源数据库实例使用透明数据加密（TDE）来加密数据，则只读副本还会自动配置 TDE。

  如果源数据库实例使用 KMS 密钥来加密数据，则同一区域中的只读副本使用相同的 KMS 密钥。对于跨区域只读副本，您必须在创建只读副本时，指定只读副本所在区域中的 KMS 密钥。您无法更改只读副本的 KMS 密钥。
- 只读副本与源数据库实例具有相同的时区和排序规则，无论它们是在哪个可用区中创建的。
- Amazon RDS for SQL Server 不支持以下内容：
  - 只读副本的备份保留
  - 从只读副本进行时间点恢复
  - 只读副本的手动快照
  - 多可用区只读副本
  - 创建只读副本的只读副本
  - 将用户登录同步到只读副本
- Amazon RDS for SQL Server 不会通过干预来缓解源数据库实例及其只读副本之间的高副本滞后。确保源数据库实例及其只读副本在计算能力和存储方面具有适当的大小，可以适合其运营负载。
- 您可以在 Amazon GovCloud（美国东部）和 Amazon GovCloud（美国西部）区域之间进行复制，但无法复制到 Amazon GovCloud (US) Regions 中或从其中复制。

RDS for SQL Server 副本的选项注意事项

在创建 RDS for SQL Server 副本之前，请考虑以下要求、限制和建议：

- 如果 SQL Server 副本与其源数据库实例位于同一区域，请确保它与源数据库实例属于同一选项组。对源选项组或源选项组成员资格做出的修改会传播到副本。不管副本的维护时段是什么，这些更改在应用到源数据库实例后，都会立即应用到副本。

  有关选项组的更多信息，请参阅 使用选项组。

- 在您创建 SQL Server 跨区域副本时，Amazon RDS 会为其创建专用选项组。

  您无法从专用选项组中删除 SQL Server 跨区域副本。任何其他数据库实例都无法使用 SQL Server 跨区域副本的专用选项组。

  以下选项为复制的选项。要将复制的选项添加到 SQL Server 跨区域只读副本，请将其添加到源数据库实例的选项组中。选项还会安装在源数据库实例的所有副本上。

  - `TDE`

  以下选项为非复制的选项。您可以在专用选项组中添加或删除未复制的选项。

  - `MSDTC`
  - `SQLSERVER_AUDIT`
  - 要对跨区域只读副本启用 `SQLSERVER_AUDIT` 选项，请在跨区域只读副本上的专用选项组和源实例的选项组上添加 `SQLSERVER_AUDIT` 选项。通过在 SQL Server 跨区域只读副本的源实例上添加 `SQLSERVER_AUDIT` 选项，可以在源实例的每个跨区域只读副本上创建服务器级审计对象和服务器级审计规范。要允许跨区域只读副本访问以将完成的审核日志上载到 Amazon S3 桶，请将 `SQLSERVER_AUDIT` 选项添加到专用选项组并配置选项设置。您用作审计文件目标的 Amazon S3 桶必须与跨区域只读副本位于同一区域中。您可以单独修改每个跨区域只读副本的 `SQLSERVER_AUDIT` 选项的选项设置，以便每个只读副本都能访问各自区域中的 Amazon S3 桶。

  跨区域只读副本不支持以下选项。

  - `SSRS`
  - `SSAS`
  - `SSIS`

  跨区域只读副本部分支持以下选项。

  - `SQLSERVER_BACKUP_RESTORE`
  - SQL Server 跨区域副本的源数据库实例可以具有 `SQLSERVER_BACKUP_RESTORE` 选项，但在删除源数据库实例的所有跨区域副本之前，您无法对源数据库实例执行原生还原。在创建跨区域副本期间，任何现有的原生还原任务都将被取消。您无法将 `SQLSERVER_BACKUP_RESTORE` 选项添加到专用选项组。

    有关原生备份和还原的更多信息，请参阅 [使用本机备份和还原导入和导出 SQL Server 数据库]。

  提升 SQL Server 跨区域只读副本时，提升后的副本的行为与其他 SQL Server 数据库实例相同，包括其选项的管理。有关选项组的更多信息，请参阅 [使用选项组]。

将数据库用户和对象与 SQL Server 只读副本同步

创建只读副本时主数据库实例中存在的任何登录名、自定义服务器角色、SQL 代理任务或其他服务器级对象，都应出现在新创建的只读副本中。但是，在创建只读副本后在主数据库实例中创建的任何服务器级对象不会自动复制，您必须在只读副本中手动创建它们。

数据库用户将自动从主数据库实例复制到只读副本。由于只读副本数据库处于只读模式，因此无法在数据库中更新数据库用户的安全标识符（SID）。因此，当在只读副本中创建 SQL 登录名时，务必确保该登录名的 SID 与主数据库实例中相应 SQL 登录名的 SID 相匹配。如果您未同步 SQL 登录名的 SID，这些登录名将无法访问只读副本中的数据库。经 Windows Active Directory（AD）身份验证的登录名不会遇到此问题，因为 SQL Server 从 Active Directory 获取 SID。

将 SQL 登录名从主数据库实例同步到只读副本

1. 连接到主数据库实例。
2. 在主数据库实例中创建新的 SQL 登录名。

    ```
    USE [master]
    GO
    CREATE LOGIN TestLogin1
    WITH PASSWORD = 'REPLACE WITH PASSWORD';
    ```

    > **Note**
    >
    > 作为安全最佳实践，请指定除此处所示提示以外的密码。

3. 为数据库中的 SQL 登录名创建新的数据库用户。

    ```
    USE [REPLACE WITH YOUR DB NAME]
    GO
    CREATE USER TestLogin1 FOR LOGIN TestLogin1;
    GO
    ```

4. 检查主数据库实例中新创建的 SQL 登录名的 SID。

    ```
    SELECT name, sid FROM sys.server_principals WHERE name =  TestLogin1;
    ```

5. 连接到只读副本。创建新的 SQL 登录名。

```
CREATE LOGIN TestLogin1 WITH PASSWORD = 'REPLACE WITH PASSWORD', SID=[REPLACE WITH
 sid FROM STEP #4];
```

或者，如果您有权访问只读副本数据库，则可以按如下方式修复孤立用户：

1. 连接到只读副本。
2. 识别数据库中的孤立用户。

```
USE [REPLACE WITH YOUR DB NAME]
GO
EXEC sp_change_users_login 'Report';
GO
```

3. 为孤立的数据库用户创建新的 SQL 登录名。

```
CREATE LOGIN TestLogin1 WITH PASSWORD = 'REPLACE WITH PASSWORD', SID=[REPLACE WITH
 sid FROM STEP #2];
```

例如：

```
CREATE LOGIN TestLogin1 WITH PASSWORD = 'TestPa$$word#1',
 SID=[0x1A2B3C4D5E6F7G8H9I0J1K2L3M4N5O6P];
```

> **Note**
>
> 作为安全最佳实践，请指定除此处所示提示以外的密码。

排除 SQL Server 只读副本问题

您可以通过查看 Amazon RDS `ReplicaLag` 指标，在 Amazon CloudWatch 中监控复制滞后。有关使用复制滞后时间的信息，请参阅 监控只读复制。

如果复制滞后时间太长，则您可以使用以下查询，获取有关滞后的信息。

```
SELECT AR.replica_server_name
     , DB_NAME (ARS.database_id) 'database_name'
     , AR.availability_mode_desc
```

```
     , ARS.synchronization_health_desc
     , ARS.last_hardened_lsn
     , ARS.last_redone_lsn
     , ARS.secondary_lag_seconds
FROM sys.dm_hadr_database_replica_states ARS
INNER JOIN sys.availability_replicas AR ON ARS.replica_id = AR.replica_id
--WHERE DB_NAME(ARS.database_id) = 'database_name'
ORDER BY AR.replica_server_name;
```

Amazon RDS for Microsoft SQL Server 多可用区部署

多可用区部署为数据库实例提供了更高的可用性、数据持久性和容错能力。在进行计划的数据维护或发生未计划的服务中断时，Amazon RDS 会自动故障转移到最新的辅助数据库实例。此功能让数据库操作快速恢复，而不需要手动干预。主实例和备用实例使用相同的终端节点，其物理网络地址作为故障转移过程的一部分转换到辅助副本。在故障转移时，您不必重新配置应用程序。

Amazon RDS 通过使用 SQL Server 数据库镜像（DBM）或 Always On 可用性组（AG）支持为运行 Microsoft SQL Server 进行多可用区部署。Amazon RDS 监控并维护您的多可用区部署的运行状况。如果出现问题，RDS 会自动修复运行状况不佳的数据库实例，重新建立同步，并发起故障转移。仅当备用实例和主要实例完全同步时，才会发生故障转移。您不必管理任何内容。

当您设置 SQL Server 多可用区时，RDS 自动将实例上的所有数据库配置为使用 DBM 或 AG。Amazon RDS 负责处理主要、见证以及辅助数据库实例。因为配置是自动的，所以 RDS 基于您部署的 SQL Server 版本选择 DBM 或 Always On AG。

Amazon RDS 支持在以下 SQL Server 版本中使用带 Always On AG 的多可用区：

- SQL Server 2022：
  - 标准版
  - Enterprise Edition
- SQL Server 2019：
  - 标准版 15.00.4073.23 及更高版本
  - Enterprise Edition
- SQL Server 2017：
  - 标准版 14.00.3401.7 及更高版本
  - 企业版 14.00.3049.1 及更高版本
- SQL Server 2016：企业版 13.00.5216.0 及更高版本

Amazon RDS 支持以下 SQL Server 版本带 DBM 的多可用区，前面提到的版本除外：

- SQL Server 2019：标准版 15.00.4043.16
- SQL Server 2017：标准版和企业版
- SQL Server 2016：标准版和企业版

您可以使用以下 SQL 查询来确定 SQL Server 数据库实例是单可用区、DBM 多可用区还是 Always On AG 多可用区。

```
SELECT CASE WHEN dm.mirroring_state_desc IS NOT NULL THEN 'Multi-AZ (Mirroring)'
    WHEN dhdrs.group_database_id IS NOT NULL THEN 'Multi-AZ (AlwaysOn)'
    ELSE 'Single-AZ'
    END 'high_availability'
FROM sys.databases sd
LEFT JOIN sys.database_mirroring dm ON sd.database_id = dm.database_id
LEFT JOIN sys.dm_hadr_database_replica_states dhdrs ON sd.database_id =
 dhdrs.database_id AND dhdrs.is_local = 1
WHERE DB_NAME(sd.database_id) = 'rdsadmin';
```

输出与以下内容类似：

```
high_availability
Multi-AZ (AlwaysOn)
```

向 Microsoft SQL Server 数据库实例添加多可用区

当您使用 Amazon Web Services Management Console创建新的 SQL Server 数据库实例时，您可以添加带数据库镜像 (DBM) 或 Always On AG 的多可用区。您可以通过从多可用区部署中选择 Yes (Mirroring / Always On) (是（镜像/Always On）) 来执行此操作。有关更多信息，请参阅 创建 Amazon RDS 数据库实例。

当使用控制台修改现有 SQL Server 数据库实例时，可以通过从 Modify DB instance（修改数据库实例）页面上的 Multi-AZ deployment（多可用区部署）中选择 Yes (Mirroring / Always On) [是（镜像/始终使用）]，以添加带 DBM 或 AG 的多可用区。有关更多信息，请参阅 修改 Amazon RDS 数据库实例。

> **Note**
>
> 如果您的数据库实例运行的是数据库镜像 (DBM)（并非 Always On 可用性组 (AG)），则可能需要在添加多可用区之前禁用内存中优化。如果数据库实例运行 SQL Server 2016 或 2017 企业版并启用了内存优化，则在添加多可用区之前通过 DBM 禁用内存优化。
> 如果您的数据库实例正在运行 AG，则不需要此步骤。

从 Microsoft SQL Server 数据库实例删除多可用区

当您使用 Amazon Web Services Management Console修改现有 SQL Server 数据库实例时，您可以删除带 DBM 或 AG 的多可用区。您可以通过在 Modify DB instance（修改数据库实例）页面上的 Multi-AZ deployment（多可用区部署）中选择 No (Mirroring / Always On) [否（镜像/始终打开）] 来实现上述目的。有关更多信息，请参阅 修改 Amazon RDS 数据库实例。

Microsoft SQL Server 多可用区部署限制、说明和建议

以下是在 RDS for SQL Server 数据库实例上使用多可用区部署时的一些限制：

- 不支持跨区域多可用区。
- 不支持停止多可用区部署中的 RDS for SQL Server 数据库实例。
- 您不能将辅助数据库实例配置为接受数据库读取活动。
- 带 Always On 可用性组 (AG) 的多可用区支持内存中优化。
- 带 Always On 可用性组 (AG) 的多可用区不支持对可用性组侦听器进行 Kerberos 身份验证。这是因为侦听器没有服务主体名称 (SPN)。
- 您不能重命名位于 SQL Server 多可用区部署中的 SQL Server 数据库实例上的数据库。如果您需要在此类实例上重命名一个数据库，请先为数据库实例禁用多可用区，然后重命名数据库。最后，为数据库实例重新启用多可用区。
- 您只能还原使用完全恢复模式备份的多可用区数据库实例。
- 多可用区部署具有 10000 个 SQL Server 代理任务的限制。

  如果您需要更高的限制，则可联系 Amazon Web Services 支持 请求增加限制。打开 Amazon Web Services 支持 Center (Amazon Web Services Support 中心) 页面，登录（如有必要），然后选择 Create case (创建案例)。选择 Service Limit increase (提高服务限制)。填写并提交表格。
- 在位于 SQL Server 多可用区部署中的 SQL Server 数据库实例上不能有脱机数据库。
- RDS for SQL Server 不会将 MSDB 数据库权限复制到辅助实例。如果您对辅助实例需要这些权限，则必须手动重新创建它们。

以下是有关在 RDS for SQL Server 数据库实例上使用多可用区部署的一些说明：

- Amazon RDS 公开 Always On AG 可用性组侦听器终端节点。此端点显示在控制台中，由 `DescribeDBInstances` API 操作作为端点字段中的条目返回。
- Amazon RDS 支持可用性组多子网故障转移。

- 要对虚拟私有云（VPC）中的 SQL Server 数据库实例使用 SQL Server 多可用区，您需要先创建一个数据库子网组，此数据库子网组在至少两个不同可用区中具有子网。然后，将该数据库子网组分配给 SQL Server 数据库实例的主副本。
- 在将数据库实例修改为多可用区部署的过程中，数据库实例的状态为 modifying (正在修改)。Amazon RDS 创建备用数据库实例，并创建主数据库实例的备份。这个过程完成后，主数据库实例的状态变为 available (可用)。
- 多可用区部署在同一节点上维护所有数据库。如果主要主机上的某个数据库发生故障转移，所有 SQL Server 数据库都将作为一个原子单元故障转移到备用主机。Amazon RDS 预置正常运行的新主机并替换运行状况不佳的主机。
- 带 DBM 或 AG 的多可用区支持单个备用副本。
- 在辅助镜像上，将会自动复制用户、登录名和权限。您无需重新创建它们。用户定义的服务器角色仅在对多可用区部署使用 Always On AG 的数据库实例中复制。
- 在多可用区部署中，RDS for SQL Server 会创建 SQL Server 登录名，来允许 Always On AG 或数据库镜像。RDS 使用模式 `db_<dbiResourceId>_node1_login`、`db_<dbiResourceId>_node2_login` 和 `db_<dbiResourceId>_witness_login` 创建登录名。
- RDS for SQL Server 创建 SQL Server 登录名来允许访问只读副本。RDS 使用以下模式 `db_<readreplica_dbiResourceId>_node_login` 创建登录名。
- 在多可用区部署中，启用作业复制功能时，SQL Server Agent 作业将从主要主机复制到辅助主机。有关更多信息，请参阅 启用 SQL Server Agent 作业复制。
- 您可以看到，由于执行同步数据复制，因此相对于标准数据库实例部署（在单一可用区中），延迟有所提升。
- 故障转移时间受完成恢复过程所用的时间的影响。大型事务会增加故障转移时间。
- 在 SQL Server 多可用区部署中，通过故障转移重新启动仅重新启动主数据库实例。故障转移后，主数据库实例将成为新的备用数据库实例。多可用区实例的参数可能不更新。对于不进行失效转移的重新启动，主数据库实例和备用数据库实例将重新启动，并在重新启动后更新参数。如果数据库实例没有响应，我们建议重新启动而不进行故障转移。

以下是有关使用针对 RDS for Microsoft SQL Server 数据库实例的多可用区部署的一些建议：

- 对于生产或预生产中使用的数据库，我们建议使用以下选项：
  - 多可用区部署，可实现高可用性
  - “Provisioned IOPS (预置 IOPS)”，可实现快速一致的性能
  - “内存优化”而非“通用型”

- 您无法为辅助实例选择可用区 (AZ)，因此，在部署应用程序主机时请注意这一点。您的数据库可能故障转移到其他可用区，并且应用程序主机与数据库可能不在同一可用区中。因此，我们建议您在给定 Amazon 区域中的所有可用区之间平衡应用程序主机。
- 为了获得最佳性能，请勿在大型数据加载操作期间启用数据库镜像或 Always On AG。如果您希望尽可能快地完成数据加载，请先完成数据加载，然后再将数据库实例转换到多可用区部署。
- 访问 SQL Server 数据库的应用程序应具有可捕获连接错误的异常处理功能。下面的代码示例显示了一个可捕获通信错误的 try/catch 块。在此示例中，如果连接成功，`break` 语句会退出 `while` 循环，但如果引发异常，则最多可重试 10 次。

```
int RetryMaxAttempts = 10;
int RetryIntervalPeriodInSeconds = 1;
int iRetryCount = 0;
while (iRetryCount < RetryMaxAttempts)
{
   using (SqlConnection connection = new SqlConnection(DatabaseConnString))
   {
      using (SqlCommand command = connection.CreateCommand())
      {
         command.CommandText = "INSERT INTO SOME_TABLE VALUES ('SomeValue');";
         try
         {
            connection.Open();
            command.ExecuteNonQuery();
            break;
         }
         catch (Exception ex)
         {
            Logger(ex.Message);
            iRetryCount++;
         }
         finally {
            connection.Close();
         }
      }
   }
   Thread.Sleep(RetryIntervalPeriodInSeconds * 1000);
}
```

- 在使用多可用区实例时，请勿使用 `Set Partner Off` 命令。例如，请勿执行以下操作。

```
--Don't do this
```

```
ALTER DATABASE db1 SET PARTNER off
```

- 请勿将恢复模式设置为 simple。例如，请勿执行以下操作。

```
--Don't do this
ALTER DATABASE db1 SET RECOVERY simple
```

- 在多可用区数据库实例上创建新登录名时，请勿使用 DEFAULT_DATABASE 参数，因为这些设置不能应用于备用镜像。例如，请勿执行以下操作。

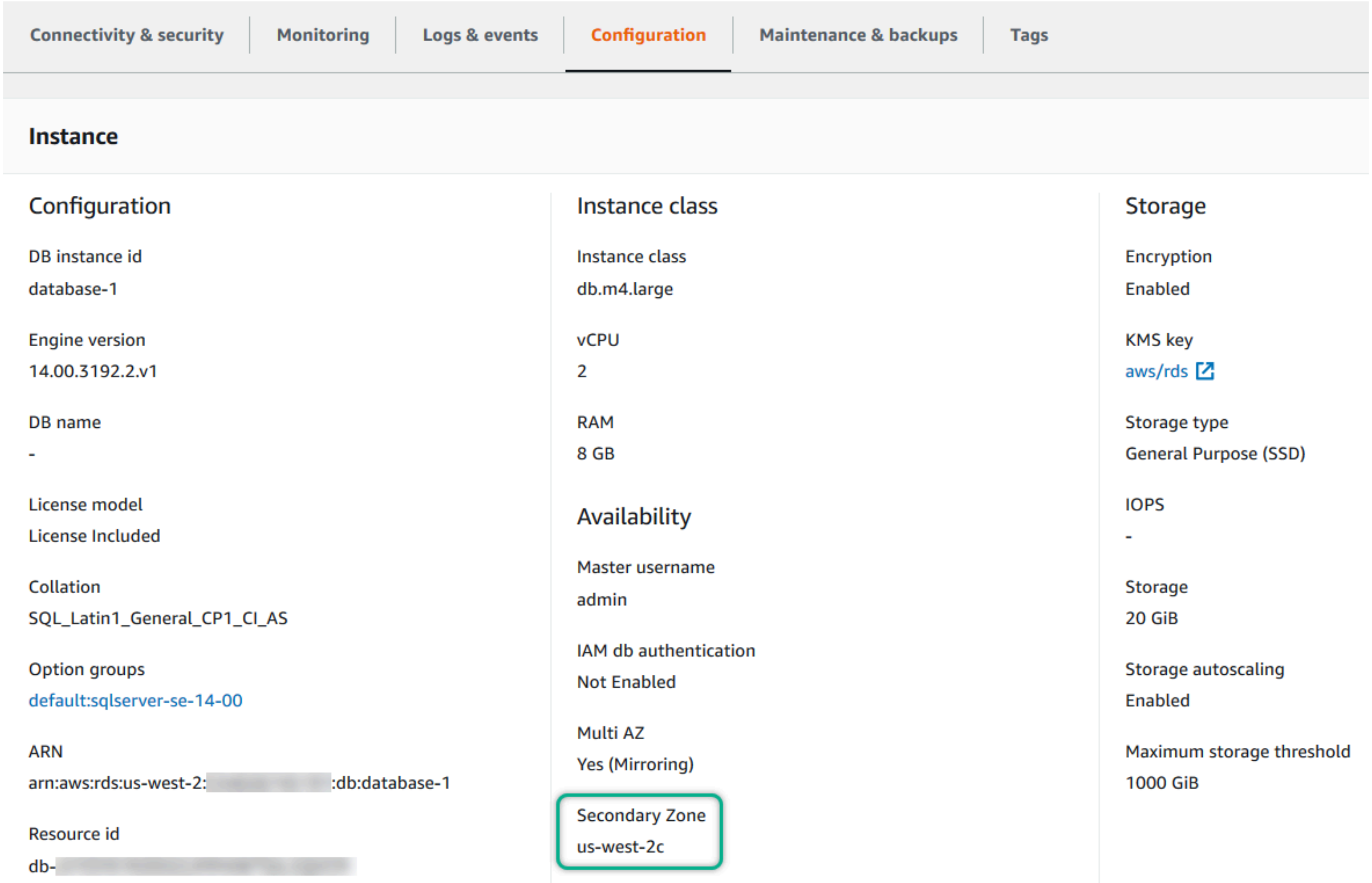
```
--Don't do this
CREATE LOGIN [test_dba] WITH PASSWORD=foo, DEFAULT_DATABASE=[db2]
```

  此外，请勿执行以下操作。

```
--Don't do this
ALTER LOGIN [test_dba] SET DEFAULT_DATABASE=[db3]
```

确定辅助副本的位置

您可以使用 Amazon Web Services Management Console确定辅助副本的位置。如果您在 VPC 中设置主数据库实例，则需要了解辅助副本的位置。



您还可以使用 Amazon CLI 命令 `describe-db-instances` 或 RDS API 操作 `DescribeDBInstances` 查看辅助副本的可用区。输出会显示备用镜像所处的辅助可用区。

从数据库镜像迁移到 Always On 可用性组

在 Microsoft SQL Server 企业版的版本 14.00.3049.1 中，默认已启用 Always On 可用性组 (AG)。

若要从数据库镜像 (DBM) 迁移到 AG，请先检查您的版本。如果您使用的数据库实例的版本低于企业版 13.00.5216.0，请修改此实例以便将其版本修补到 13.00.5216.0 或更高版本。如果您使用的数据库实例的版本低于企业版 14.00.3049.1，请修改实例以便将其版本修补到 14.00.3049.1 或更高版本。

如果您想要升级镜像的数据库实例以使用 AG，请先运行升级，修改实例以删除多可用区，然后再次修改它以添加多可用区。这会将实例转换为使用 Always On AG。

Amazon RDS 上的 Microsoft SQL Server 的其他功能

在以下各节中，您可以找到有关增强运行 Microsoft SQL Server 数据库引擎的 Amazon RDS 实例的信息。

主题

在 RDS for SQL Server 上为 SQL Server 登录名使用密码策略

Amazon RDS 支持您为运行 Microsoft SQL Server 的 Amazon RDS 数据库实例设置密码策略。使用这一点为使用 SQL Server 身份验证对数据库实例进行身份验证的登录名设置复杂性、长度和锁定要求。

关键术语

登录

在 SQL Server 中，可以对数据库实例进行身份验证的服务器级主体称为登录名。其它数据库引擎可能会将此主体称为用户。在 RDS for SQL Server 中，登录名可以使用 SQL Server 身份验证或 Windows 身份验证进行身份验证。

SQL Server 登录名

使用用户名和密码通过 SQL Server 身份验证进行身份验证的登录名是 SQL Server 登录名。您通过数据库参数配置的密码策略仅适用于 SQL Server 登录名。

Windows 登录名

基于 Windows 主体并使用 Windows 身份验证进行身份验证的登录名是 Windows 登录名。您可以在 Active Directory 中为 Windows 登录名配置密码策略。有关更多信息，请参阅 将 Active Directory 用于 RDS for SQL Server。

为每个登录名启用和禁用策略

每个 SQL Server 登录名都有用于 CHECK_POLICY 和 CHECK_EXPIRATION 的标志。默认情况下，创建新登录名时，CHECK_POLICY 设置为 ON，而 CHECK_EXPIRATION 设置为 OFF。

如果为登录名启用 CHECK_POLICY，则 RDS for SQL Server 会根据复杂性和最小长度要求验证密码。锁定策略也适用。用于启用 CHECK_POLICY 和 CHECK_EXPIRATION 的示例 T-SQL 语句：

```
ALTER LOGIN [master_user] WITH CHECK_POLICY = ON, CHECK_EXPIRATION = ON;
```

如果启用 CHECK_EXPIRATION，则密码受密码使用期限策略的约束。用于检查是否设置了 CHECK_POLICY 和 CHECK_EXPIRATION 的 T-SQL 语句：

```
SELECT name, is_policy_checked, is_expiration_checked FROM sys.sql_logins;
```

密码策略参数

所有密码策略参数都是动态的，不需要重启数据库即可生效。下表列出了您可以设置来修改 SQL Server 登录名的密码策略的数据库参数：

数据库参数	描述	允许的值	默认值	

rds.password_complexity_enabled	创建或更改 SQL Server 登录名的密码时，必须满足密码复杂性要求。必须满足以下约束：
	
	• 密码必须包含以下三个类别的字符：
	  • 拉丁文小写字母（a 到 z）
	  • 拉丁文大写字母（A 到 Z）
	  • 非字母数字字符，例如：感叹号（!）、美元符号（$）、数字符号（#）或百分比（%）。
	• 密码不包含用户的账户名称。
		0、1	0	

数据库参数	描述	允许的值	默认值
rds.password_min_length	SQL Server 登录名的密码中所需的最小字符数。	0-14	0
rds.password_min_age	用户在更改 SQL Server 登录名密码之前必须使用该密码的最少天数。当设置为 0 时，可以立即更改密码。	0-998	0
rds.password_max_age	可以使用 SQL Server 登录名密码的最大天数，之后用户需要更改密码。设置为 0 时，密码从不过期。	0-999	42
rds.password_lockout_threshold	导致 SQL Server 登录名被锁定的连续失败登录尝试次数。	0-999	0
rds.password_lockout_duration	锁定的 SQL Server 登录名在被解锁之前必须等待的分钟数。	1-60	10
rds.password_lockout_reset_counter_after	登录尝试失败后必须经过的分钟数，之后才会将失败的登录尝试计数器重置为 0。	1-60	10

> **Note**
>
> 有关 SQL Server 密码策略的更多信息，请参阅密码策略。
>
> 密码复杂性和最小长度策略也适用于所包含的数据库中的数据库用户。有关更多信息，请参阅包含的数据库。

以下约束适用于密码策略参数：

- 除非 `rds.password_max_age` 设置为 0，否则 `rds.password_min_age` 参数必须小于 `rds.password_max_age parameter`
- `rds.password_lockout_reset_counter_after` 参数必须小于或等于 `rds.password_lockout_duration` 参数。
- 如果 `rds.password_lockout_threshold` 设置为 0，则 `rds.password_lockout_duration` 和 `rds.password_lockout_reset_counter_after` 不适用。

现有登录名的注意事项

修改实例上的密码策略后，不会根据新的密码复杂性和长度要求对登录名的现有密码进行追溯性评估。只有新密码才会根据新策略进行验证。

SQL Server 确实会根据使用期限要求评估现有密码。

修改密码策略后，密码可能会立即过期。例如，如果登录名已启用 `CHECK_EXPIRATION`，其密码上次更改是在 100 天前，并且您将 `rds.password_max_age` 参数设置为 5 天，则密码将立即过期，并且登录名需要在下次尝试登录时更改其密码。

> **Note**
>
> RDS for SQL Server 不支持密码历史记录策略。历史记录策略可禁止登录名重用以前用过的密码。

多可用区部署的注意事项

多可用区实例的失败登录尝试计数器和锁定状态不会在节点之间复制。如果在多可用区实例失效转移时登录名被锁定，则登录名可能已经在新节点上被解锁。

主登录名的密码注意事项

创建 RDS for SQL Server 数据库实例时，不会根据密码策略评估主用户密码。在对主用户执行操作时，特别是在 `ModifyDBInstance` 命令中设置 `MasterUserPassword` 时，也不会根据密码评估新的主密码。在这两种情况下，您都可以为不符合密码策略的主用户设置密码，并且操作仍然会成功。如果不符合该策略，则 RDS 会尝试引发 RDS 事件，并建议设置强密码。请注意，只对主用户使用强密码。

当主用户密码不符合密码策略要求时，RDS 会尝试生成以下事件消息：

- 已创建主用户，但密码不符合密码策略的最小长度要求。请考虑使用更强的密码。
- 已创建主用户，但密码不符合密码策略的复杂性要求。请考虑使用更强的密码。
- 已重置主用户密码，但密码不符合密码策略的最小长度要求。请考虑使用更强的密码。
- 已重置主用户密码，但密码不符合密码策略的复杂性要求。请考虑使用更强的密码。

默认情况下，创建主用户时，`CHECK_POLICY` 和 `CHECK_EXPIRATION` 设置为 `OFF`。要将密码策略应用于主用户，必须在创建数据库实例后，手动为主用户启用这些标志。启用这些标志后，直接在 SQL Server 中修改主用户密码（例如，通过 T-SQL 语句或 SSMS），来根据密码策略验证新密码。

> **Note**
>
> 如果主用户被锁定，则可以通过使用 `ModifyDBInstance` 命令重置主用户密码来为该用户解锁。

修改主用户密码

可以使用 ModifyDBInstance 命令修改主用户密码。

> **Note**
>
> 重置主用户密码时，RDS 会为主用户重置各种权限，而主用户可能会失去某些权限。如果主用户已被锁定，则重置主用户密码也会解除其锁定。

RDS 会验证新的主用户密码，如果密码不符合策略，则尝试发出 RDS 事件。即使密码不符合密码策略，RDS 也会设置密码。

将 Amazon RDS for SQL Server 数据库实例与 Amazon S3 集成

您可以在运行 Amazon RDS for SQL Server 的数据库实例和 Amazon S3 存储桶之间传输文件。通过执行此操作，您可以结合使用 Amazon S3 和 SQL Server 功能，例如批量插入。例如，您可以将 .csv、.xml、.txt 和其他文件从 Amazon S3 下载到数据库实例主机，并将数据从 `D:\S3\` 导入到数据库中。所有文件都存储在数据库实例上的 `D:\S3\` 中。

适用以下限制：

- 在多可用区实例上进行故障转移后，将在备用副本上删除 `D:\S3` 文件夹中的文件。有关更多信息，请参阅“S3 集成的多可用区限制”。
- 数据库实例和 S3 存储桶必须位于同一 Amazon 区域。
- 如果您一次运行多个 S3 集成任务，则这些任务将按顺序运行，而不是并行运行。

> **Note**
>
> S3 集成任务与本机备份和还原任务共享相同的队列。在此队列中，任何时候最多只能有两个正在进行的任务。因此，两个正在运行的本机备份和还原任务将阻止所有 S3 集成任务。

- 您必须在已还原的实例上重新启用 S3 集成功能。S3 集成不会从源实例传播到已还原的实例。`D:\S3` 中的文件在还原的实例上被删除。
- 下载到数据库实例的文件数不能超过 100 个。换句话说，`D:\S3\` 中的文件数不能超过 100 个。
- 仅支持下载无文件扩展名或具有以下文件扩展名的文件：.abf、.asdatabase、.bcp、configsettings、.csv、.dat、.deploymentoptions、.deploymenttargets、.fm 和 .xmla。
- S3 存储桶必须与相关 Amazon Identity and Access Management (IAM) 角色具有同一所有者。因此，不支持跨账户 S3 集成。
- S3 存储桶不能向公众开放。
- 从 RDS 上传到 S3 的文件大小限制为每个文件 50 GB。
- 从 S3 下载到 RDS 的文件大小限制为 S3 支持的最大文件大小。

主题

- 列出 RDS 数据库实例上的文件
- 删除 RDS 数据库实例上的文件
- 监控文件传输任务的状态
- 取消任务
- S3 集成的多可用区限制
- 禁用 RDS for SQL Server 与 S3 的集成

有关使用 Amazon S3 中的文件的更多信息，请参阅 Amazon Simple Storage Service 入门。

将 RDS for SQL Server 与 S3 集成的先决条件

在开始之前，请找到或创建要使用的 S3 存储桶。另外，添加权限，以便 RDS 数据库实例可以访问 S3 存储桶。要配置此访问权限，请同时创建 IAM 策略和 IAM 角色。

控制台

创建用于访问 Amazon S3 的 IAM 策略

1. 在 IAM 管理控制台中，选择导航窗格中的策略。
2. 创建新策略，并使用 Visual editor (可视化编辑器) 选项卡执行以下步骤。
3. 对于 Service (服务)，输入 **S3**，然后选择 S3 服务。
4. 对于操作，选择以下各项以授予数据库实例所需的访问权限：
   - `ListAllMyBuckets` – 必需
   - `ListBucket` – 必需
   - `GetBucketAcl` – 必需
   - `GetBucketLocation` – 必需
   - `GetObject` – 将文件从 S3 下载到 所必需`D:\S3\`
   - `PutObject` – 将文件从 `D:\S3\` 上传到 S3 所必需
   - `ListMultipartUploadParts` – 将文件从 `D:\S3\` 上传到 S3 所必需
   - `AbortMultipartUpload` – 将文件从 `D:\S3\` 上传到 S3 所必需
5. 对于 Resources (资源)，显示的选项取决于您在上一步中选择的操作。您可能会看到针对 bucket (存储桶) 和/或 object (对象) 的选项。对于其中的每一个，添加适当的 Amazon Resource Name (ARN)。

对于 bucket (存储桶)，为要使用的存储桶添加 ARN。例如，如果存储桶名为 *amzn-s3-demo-bucket*，请将 ARN 设置为 `arn:aws:s3:::amzn-s3-demo-bucket`。

对于 object (对象)，为存储桶输入 ARN，然后选择以下各项之一：

- 要授予对指定存储桶中所有文件的访问权限，请为 Bucket name (存储桶名称) 和 Object name (对象名称) 选择 Any (任何)。
- 要授予对存储桶中特定文件或文件夹的访问权限，请提供您希望 SQL Server 访问的特定存储桶和对象的 ARN。

6. 按照控制台中的说明进行操作，直到您完成策略创建。

   前面是设置策略的简要指南。有关创建 IAM 策略的更多详细说明，请参阅 IAM 用户指南中的 创建 IAM 策略。

创建使用上一过程中的 IAM 策略的 IAM 角色

1. 在 IAM 管理控制台中，选择导航窗格中的角色。
2. 创建新的 IAM 角色，然后选择控制台中显示的以下选项：
   - Amazon 服务
   - RDS
   - RDS – Add Role to Database (将角色添加到数据库)

   然后选择底部的 Next:Permissions (下一步: 权限)。
3. 对于 Attach permissions policies (附加权限策略)，输入您之前创建的 IAM 策略的名称。然后，从列表中选择该策略。
4. 按照控制台中的说明进行操作，直到您完成角色创建。

   前面是设置角色的简要指南。如果您需要有关创建角色的更多详细说明，请参阅 IAM 用户指南中的 IAM 角色。

Amazon CLI

要授予 Amazon RDS 对 Amazon S3 存储桶的访问权限，请使用以下过程：

1. 创建向 Amazon RDS 授予对 S3 存储桶的访问权限的 IAM 策略。

2. 创建一个让 Amazon RDS 可代表您访问 S3 存储桶的 IAM 角色。

   有关更多信息，请参阅《IAM 用户指南》中的创建向 IAM 用户委派权限的角色。
3. 将您创建的 IAM 策略附加到您创建的 IAM 角色。

创建 IAM policy

包括适当的操作以授予数据库实例所需的访问权限：

- `ListAllMyBuckets` – 必需
- `ListBucket` – 必需
- `GetBucketAcl` – 必需
- `GetBucketLocation` – 必需
- `GetObject` – 将文件从 S3 下载到 所必需`D:\S3\`
- `PutObject` – 将文件从 `D:\S3\` 上传到 S3 所必需
- `ListMultipartUploadParts` – 将文件从 `D:\S3\` 上传到 S3 所必需
- `AbortMultipartUpload` – 将文件从 `D:\S3\` 上传到 S3 所必需

1. 以下 Amazon CLI 命令使用这些选项创建一个名为 `rds-s3-integration-policy` 的 IAM 策略。该策略授予对名为 *`amzn-s3-demo-bucket`* 的存储桶的访问权限。

   Example

   对于 Linux、macOS 或 Unix：

```
aws iam create-policy \
  --policy-name rds-s3-integration-policy \
  --policy-document '{
         "Version": "2012-10-17",
         "Statement": [
             {
                 "Effect": "Allow",
                 "Action": "s3:ListAllMyBuckets",
                 "Resource": "*"
             },
             {
                 "Effect": "Allow",
                 "Action": [
```

```
                    "s3:ListBucket",
                    "s3:GetBucketAcl",
                    "s3:GetBucketLocation"
                ],
                "Resource": "arn:aws:s3:::amzn-s3-demo-bucket"
            },
            {
                "Effect": "Allow",
                "Action": [
                    "s3:GetObject",
                    "s3:PutObject",
                    "s3:ListMultipartUploadParts",
                    "s3:AbortMultipartUpload"
                ],
                "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/key_prefix/*"
            }
        ]
    }'
```

对于 Windows：

确保将行尾更改为您的界面支持的行尾（^ 而不是 \）。另外，在 Windows 中，您必须使用 \ 来转义所有双引号。为了避免需要转义 JSON 中的引号，您可以将其保存到文件中并将该文件作为参数传入。

首先，创建包含以下权限策略的 policy.json 文件：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:ListAllMyBuckets",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket",
                "s3:GetBucketAcl",
                "s3:GetBucketLocation"
            ],
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket"
```

```
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:PutObject",
                "s3:ListMultipartUploadParts",
                "s3:AbortMultipartUpload"
            ],
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/key_prefix/*"
        }
    ]
}
```

然后使用以下命令创建策略：

```
aws iam create-policy ^
     --policy-name rds-s3-integration-policy ^
     --policy-document file://file_path/assume_role_policy.json
```

2. 创建策略之后，请记下策略的 Amazon Resource Name (ARN)。后续步骤需要该 ARN。

创建 IAM 角色

- 以下 Amazon CLI 命令创建 `rds-s3-integration-role` IAM 角色来实现此目的。

  Example

  对于 Linux、macOS 或 Unix：

```
aws iam create-role \
    --role-name rds-s3-integration-role \
    --assume-role-policy-document '{
      "Version": "2012-10-17",
      "Statement": [
        {
          "Effect": "Allow",
          "Principal": {
             "Service": "rds.amazonaws.com"
           },
          "Action": "sts:AssumeRole"
        }
```

```
        ]
    }'
```

对于 Windows：

确保将行尾更改为您的界面支持的行尾（^ 而不是 \）。另外，在 Windows 中，您必须使用 \ 来转义所有双引号。为了避免需要转义 JSON 中的引号，您可以将其保存到文件中并将该文件作为参数传入。

首先，创建包含以下策略的 `assume_role_policy.json` 文件：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "rds.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

然后使用以下命令创建 IAM 角色：

```
aws iam create-role ^
     --role-name rds-s3-integration-role ^
     --assume-role-policy-document file://file_path/assume_role_policy.json
```

Example 使用全局条件上下文键创建 IAM 角色

我们建议在基于资源的策略中使用 [`aws:SourceArn`] 和 [`aws:SourceAccount`] 全局条件上下文键，以此限制服务对特定资源的权限。这是防范[混淆代理问题]最有效的方法。

您可以使用这两个全局条件上下文键并让 `aws:SourceArn` 值包含账户 ID。在这种情况下，当 `aws:SourceAccount` 值和 `aws:SourceArn` 值中的账户使用相同策略语句时，确保二者使用相同的账户 ID。

- 如果您想对单个资源进行跨服务访问，请使用 aws:SourceArn。
- 如果您想允许该账户中的任何资源与跨服务使用操作相关联，请使用 aws:SourceAccount。

在此策略中，请务必使用 aws:SourceArn 全局条件上下文键和访问角色资源的完整 Amazon Resource Name (ARN)。对于 S3 集成，请确保包含数据库实例 ARN，如以下示例所示。

对于 Linux、macOS 或 Unix：

```
aws iam create-role \
    --role-name rds-s3-integration-role \
    --assume-role-policy-document '{
      "Version": "2012-10-17",
      "Statement": [
        {
          "Effect": "Allow",
          "Principal": {
             "Service": "rds.amazonaws.com"
           },
          "Action": "sts:AssumeRole",
                "Condition": {
                    "StringEquals": {

 "aws:SourceArn":"arn:aws:rds:Region:my_account_ID:db:db_instance_identifier"
                    }
                }
        }
      ]
    }'
```

对于 Windows：

将全局条件上下文键添加至 assume_role_policy.json。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
```

```
                    "rds.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
 "aws:SourceArn":"arn:aws:rds:Region:my_account_ID:db:db_instance_identifier"
                }
            }
        }
    ]
}
```

将 IAM 策略附加到 IAM 角色

- 以下 Amazon CLI 命令将策略附加到名为 `rds-s3-integration-role` 的角色。将 *`your-policy-arn`* 替换为您在上一步中记下的策略 ARN。

  Example

  对于 Linux、macOS 或 Unix：

  ```
  aws iam attach-role-policy \
      --policy-arn your-policy-arn \
      --role-name rds-s3-integration-role
  ```

  对于 Windows：

  ```
  aws iam attach-role-policy ^
      --policy-arn your-policy-arn ^
      --role-name rds-s3-integration-role
  ```

启用 RDS for SQL Server 与 S3 的集成

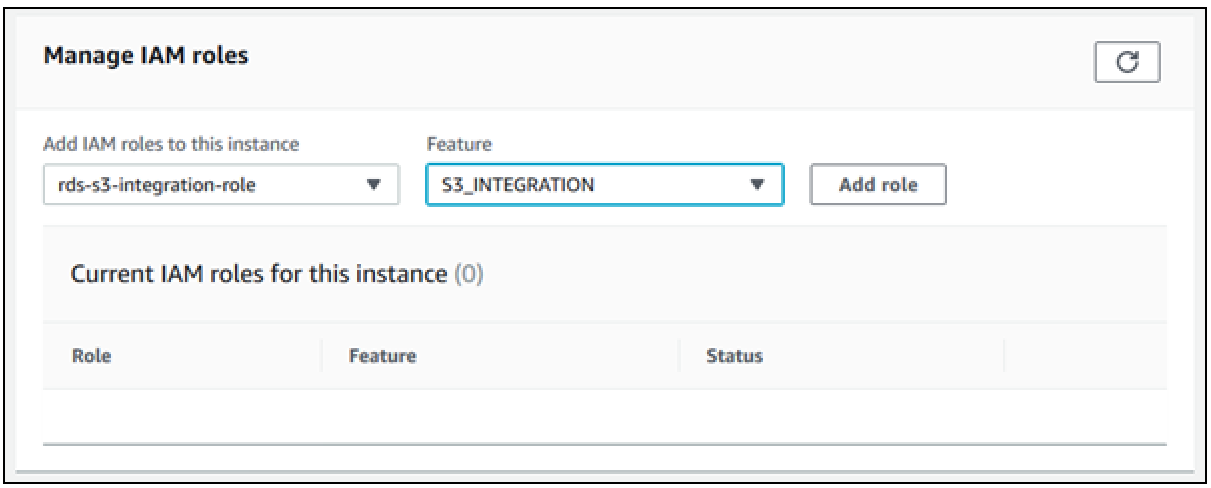
在以下部分中，您可以找到如何启用 Amazon S3 与 Amazon RDS for SQL Server 的集成。要使用 S3 集成，您的数据库实例必须与您先前创建的 IAM 角色相关联，然后才能使用 `S3_INTEGRATION` feature-name 参数。

> **Note**
>
> 要将 IAM 角色添加到数据库实例，数据库实例的状态必须为 available。

控制台

将您的 IAM 角色与数据库实例关联

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 选择 RDS for SQL Server 数据库实例名称以显示其详细信息。
3. 在 Connectivity & security (连接性和安全性) 选项卡上的 Manage IAM roles (管理 IAM 角色) 部分中，为 Add IAM roles to this instance (向此实例添加 IAM 角色) 选择要添加的 IAM 角色。
4. 对于 Feature (功能)，选择 S3_INTEGRATION。



5. 选择 Add role (添加角色)。

Amazon CLI

将 IAM 角色添加到 RDS for SQL Server 数据库实例

- 以下 Amazon CLI 命令将您的 IAM 角色添加到名为 *mydbinstance* 的 RDS for SQL Server 数据库实例中。

  Example

  对于 Linux、macOS 或 Unix：

```
aws rds add-role-to-db-instance \
    --db-instance-identifier mydbinstance \
    --feature-name S3_INTEGRATION \
    --role-arn your-role-arn
```

对于 Windows：

```
aws rds add-role-to-db-instance ^
    --db-instance-identifier mydbinstance ^
    --feature-name S3_INTEGRATION ^
    --role-arn your-role-arn
```

将 *your-role-arn* 替换为您在上一步中记下的角色 ARN。必须为 S3_INTEGRATION 选项指定 --feature-name。

在 RDS for SQL Server 和 Amazon S3 之间传输文件

您可以使用 Amazon RDS 存储过程在 Amazon S3 和 RDS 数据库实例之间下载和上传文件。您还可以使用 Amazon RDS 存储过程来列出和删除 RDS 实例上的文件。

您从 S3 下载和上传到 S3 的文件存储在 D:\S3 文件夹中。这是可用于访问文件的唯一文件夹。您可以将文件组织到子文件夹中，这些子文件夹是您在下载过程中包含目标文件夹时为您创建的。

某些存储过程要求您向 &S3 存储桶和文件提供 Amazon Resource Name (ARN)。您的 ARN 的格式为 arn:aws:s3:::*amzn-s3-demo-bucket*/file_name。Amazon S3 不需要在 ARN 中使用账号或 Amazon区域。

S3 集成任务按顺序运行，并且与本机备份和还原任务共用同一队列。在此队列中，任何时候最多只能有两个正在进行的任务。任务可能需要长达五分钟时间才能开始处理。

将文件从 Amazon S3 存储桶下载到 SQL Server 数据库实例

要将文件从 S3 存储桶下载到 RDS for SQL Server 数据库实例，请使用具有以下参数的 Amazon RDS 存储过程 msdb.dbo.rds_download_from_s3。

参数名称	数据类型	默认值	必需	描述
@s3_arn_of_file	NVARCHAR	–	必需	要下载的文件的 S3 ARN，例如：arn:aws:s

参数名称	数据类型	默认值	必需	描述
				3::: *amzn-s3-demo-bucket* /mydata.csv
@rds_file_path	NVARCHAR	–	可选	RDS 实例的文件路径。如果未指定，则文件路径为 D:\S3*<filename in s3>*。RDS 支持绝对路径和相对路径。如果要创建子文件夹，请将其包含在文件路径中。
@overwrite_file	INT	0	可选	覆盖现有文件： 0 = 不覆盖 1 = 覆盖

您可以下载不带文件扩展名的文件和具有以下文件扩展名的文件：.bcp、.csv、.dat、.fmt、.info、.lst、.tbl、.txt 和 .xml。

> **Note**
>
> 启用 SQL Server Integration Services 时，支持下载具有 .ispac 文件扩展名的文件。有关启用 SSIS 的更多信息，请参阅SQL Server Integration Services。
>
> 启用 SQL Server Analysis Services 时，支持下载具有以下文件扩展名的文件：.abf、.asdatabase、.configsettings、.deploymentoptions、.deploymenttargets 和 .xmla。有关启用 SSAS 的更多信息，请参阅SQL Server Analysis Services。

以下示例显示了从 S3 下载文件的存储过程。

```
exec msdb.dbo.rds_download_from_s3
     @s3_arn_of_file='arn:aws:s3:::amzn-s3-demo-bucket/bulk_data.csv',
     @rds_file_path='D:\S3\seed_data\data.csv',
     @overwrite_file=1;
```

示例 rds_download_from_s3 操作在 seed_data 中创建一个名为 D:\S3\ 的文件夹（如果该文件夹尚不存在）。然后，该示例将源文件 bulk_data.csv 从 S3 下载到数据库实例上名为 data.csv 的新文件。如果以前存在此文件，则会覆盖它，因为 @overwrite_file 参数设置为 1。

将文件从 SQL Server 数据库实例上传到 Amazon S3 存储桶

要将文件从 RDS for SQL Server 数据库实例上传到 S3 存储桶，请使用具有以下参数的 Amazon RDS 存储过程 msdb.dbo.rds_upload_to_s3。

参数名称	数据类型	默认值	必需	描述
@s3_arn_of_file	NVARCHAR	–	必需	要在 S3 中创建的文件的 S3 ARN，例如：arn:aws:s3::: *amzn-s3-demo-bucket* /mydata.csv
@rds_file_path	NVARCHAR	–	必需	要上传到 S3 的文件的文件路径。支持绝对路径和相对路径。
@overwrite_file	INT	–	可选	覆盖现有文件： 0 = 不覆盖 1 = 覆盖

以下示例将名为 data.csv 的文件从 D:\S3\seed_data\ 中的指定位置上传到 ARN 指定的 S3 存储桶中的 new_data.csv 文件。

```
exec msdb.dbo.rds_upload_to_s3
  @rds_file_path='D:\S3\seed_data\data.csv',
  @s3_arn_of_file='arn:aws:s3:::amzn-s3-demo-bucket/new_data.csv',
  @overwrite_file=1;
```

如果 S3 中以前存在此文件，则会覆盖它，因为 @overwrite_file 参数设置为 1。

列出 RDS 数据库实例上的文件

要列出数据库实例上可用的文件，请同时使用存储过程和函数。首先，运行以下存储过程来从 D:\S3\ 中的文件收集文件详细信息。

```
exec msdb.dbo.rds_gather_file_details;
```

此存储过程返回任务的 ID。像其他任务一样，此存储过程异步运行。一旦任务的状态为 SUCCESS，您就可以在 rds_fn_list_file_details 函数中使用任务 ID 来列出 D:\S3\ 中的现有文件和目录，如下所示。

```
SELECT * FROM msdb.dbo.rds_fn_list_file_details(TASK_ID);
```

rds_fn_list_file_details 函数将返回具有以下列的表格。

输出参数	描述
filepath	文件的绝对路径（例如，D:\S3\mydata.csv ）
size_in_bytes	文件大小（以字节为单位）
last_modified_utc	上次修改日期和时间（采用 UTC 格式）
is_directory	指示项目是否为目录的选项 (true/false)

删除 RDS 数据库实例上的文件

要删除数据库实例上可用的文件，请使用具有以下参数的 Amazon RDS 存储过程 msdb.dbo.rds_delete_from_filesystem。

参数名称	数据类型	默认值	必需	描述
@rds_file_path	NVARCHAR	–	必需	要删除的文件的文件路径。支持绝对路径和相对路径。

参数名称	数据类型	默认值	必需	描述
@force_delete	INT	0	可选	要删除目录，必须包含此标志并将其设置为 1。 1 = 删除目录 如果要删除文件，则忽略此参数。

要删除目录，@rds_file_path 必须以反斜杠 (\) 结尾，并且 @force_delete 必须设置为 1。

以下示例将删除 D:\S3\delete_me.txt 文件。

```
exec msdb.dbo.rds_delete_from_filesystem
    @rds_file_path='D:\S3\delete_me.txt';
```

以下示例将删除 D:\S3\example_folder\ 目录。

```
exec msdb.dbo.rds_delete_from_filesystem
    @rds_file_path='D:\S3\example_folder\',
    @force_delete=1;
```

监控文件传输任务的状态

要跟踪 S3 集成任务的状态，请调用 rds_fn_task_status 函数。它获取两个参数。第一个参数应该始终为 NULL，因为它不适用于 S3 集成。第二个参数接受任务 ID。

要查看所有任务的列表，请将第一个参数设置为 NULL，将第二个参数设置为 0，如以下示例所示。

```
SELECT * FROM msdb.dbo.rds_fn_task_status(NULL,0);
```

要获取特定任务，请将第一个参数设置为 NULL，将第二个参数设置为任务 ID，如以下示例所示。

```
SELECT * FROM msdb.dbo.rds_fn_task_status(NULL,42);
```

rds_fn_task_status 函数将返回以下信息。

输出参数	描述
task_id	任务的 ID。
task_type	对于 S3 集成，任务可以具有以下任务类型： • DOWNLOAD_FROM_S3 • UPLOAD_TO_S3 • LIST_FILES_ON_DISK • DELETE_FILES_ON_DISK
database_name	不适用于 S3 集成任务。
% complete	用百分比表示的任务进度。
duration(mins)	在任务上花费的时间 (以分钟为单位)。
lifecycle	任务的状态。有以下可能状态： • CREATED – 调用 S3 集成存储过程之一后，将创建一个任务并将状态设置为 CREATED。 • IN_PROGRESS – 任务开始后，状态将设置为 IN_PROGRESS 。要将状态从 CREATED 更改为 IN_PROGRESS 可能需要最多 5 分钟的时间。 • SUCCESS – 任务完成后，状态将设置为 SUCCESS。 • ERROR – 如果任务失败，状态将设置为 ERROR。有关错误的更多信息，请参阅 task_info 列。 • CANCEL_REQUESTED – 调用 rds_cancel_task 后，任务的状态将设置为 CANCEL_REQUESTED 。 • CANCELLED – 在成功取消任务后，任务的状态将设置为 CANCELLED 。

输出参数	描述
task_info	有关任务的其他信息。如果在处理过程中发生错误，则此列包含有关错误的信息。
last_updated	上次更新任务状态的日期和时间。
created_at	任务的创建日期和时间。
S3_object_arn	从中下载或上传到其中的 S3 对象的 ARN。
overwrite_S3_backup_file	不适用于 S3 集成任务。
KMS_master_key_arn	不适用于 S3 集成任务。
filepath	RDS 数据库实例上的文件路径。
overwrite_file	指示是否覆盖现有文件的选项。
task_metadata	不适用于 S3 集成任务。

取消任务

要取消 S3 集成任务，请使用带有 msdb.dbo.rds_cancel_task 参数的 task_id 存储过程。无法取消正在进行的删除和列出任务。以下示例显示了取消任务的请求。

```
exec msdb.dbo.rds_cancel_task @task_id = 1234;
```

要获取所有任务及其任务 ID 的概述，请使用rds_fn_task_status中介绍的 监控文件传输任务的状态 函数。

S3 集成的多可用区限制

在多可用区实例上，故障转移之后将删除备用副本上 D:\S3 文件夹中的文件。故障转移可以是预先计划的，例如在数据库实例修改（如更改实例类或升级引擎版本）期间。或者，故障转移可能是未计划的，在主实例停机时进行。

> **Note**
>
> 我们不建议将 `D:\S3` 文件夹用于文件存储。最佳做法是将创建的文件上传到 Amazon S3 以使其持久，并在需要导入数据时下载文件。

要确定上次故障转移时间，您可以使用 `msdb.dbo.rds_failover_time` 存储过程。有关更多信息，请参阅“[确定 Amazon RDS for SQL Server 的上次失效转移时间](#)”。

Example 最近无故障转移示例

此示例显示错误日志中最近没有故障转移的输出。自 2020-04-29 23:59:00.01 以来，没有发生故障转移。

因此，在该时间之后下载的所有文件，如未使用 `rds_delete_from_filesystem` 存储过程删除，则仍可在当前主机上访问。在该时间之前下载的文件可能也可用。

errorlog_available_from	recent_failover_time
2020-04-29 23:59:00.0100000	null

Example 最近故障转移示例

此示例显示错误日志中最近有故障转移的输出。最近一次故障转移的时间是在 2020-05-05 18:57:51.89。

在该时间之后下载的所有文件，如未使用 `rds_delete_from_filesystem` 存储过程删除，则仍可在当前主机上访问。

errorlog_available_from	recent_failover_time
2020-04-29 23:59:00.0100000	2020-05-05 18:57:51.8900000

禁用 RDS for SQL Server 与 S3 的集成

在下面，您可以找到如何禁用 Amazon S3 与 Amazon RDS for SQL Server 的集成。禁用 S3 集成时，不会删除 `D:\S3\` 中的文件。

> **Note**
>
> 要从数据库实例中删除 IAM 角色，数据库实例的状态必须为 `available`。

控制台

从数据库实例中取消关联您的 IAM 角色

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 选择 RDS for SQL Server 数据库实例名称以显示其详细信息。
3. 在 Connectivity & security (连接性和安全性) 选项卡上的 Manage IAM roles (管理 IAM 角色) 部分中，选择要删除的 IAM 角色。
4. 选择删除。

Amazon CLI

从 RDS for SQL Server 数据库实例中删除 IAM 角色

- 以下 Amazon CLI 命令从名为 *mydbinstance* 的 RDS for SQL Server 数据库实例中删除 IAM 角色。

  Example

  对于 Linux、macOS 或 Unix：

```
aws rds remove-role-from-db-instance \
    --db-instance-identifier mydbinstance \
    --feature-name S3_INTEGRATION \
    --role-arn your-role-arn
```

  对于 Windows：

```
aws rds remove-role-from-db-instance ^
    --db-instance-identifier mydbinstance ^
    --feature-name S3_INTEGRATION ^
    --role-arn your-role-arn
```

使用 *your-role-arn* 选项的适当 IAM 角色 ARN 替换 `--feature-name`。

在 Amazon RDS for SQL Server 上使用数据库邮件

您可以使用数据库邮件从 Amazon RDS on SQL Server 数据库实例向用户发送电子邮件。这些消息可以包含文件和查询结果。数据库邮件包括以下组件：

- 配置和安全对象 – 这些对象可以创建配置文件和账户，并存储在 `msdb` 数据库中。
- 消息收发对象 – 这些对象包括用于发送消息的 sp_send_dbmail 存储过程以及保存有关消息信息的数据结构。相关内容均存储在 `msdb` 数据库中。
- 日志记录和审计对象 – 数据库邮件将日志记录信息写入 `msdb` 数据库和 Microsoft Windows 应用程序事件日志中。
- 数据库邮件可执行文件 – `DatabaseMail.exe` 读取 `msdb` 数据库中的队列并发送电子邮件。

RDS 在 Web 版、标准版和企业版上支持所有 SQL Server 版本的数据库邮件。

限制

以下限制适用于在 SQL Server 数据库实例上使用数据库邮件：

- SQL Server 精简版不支持数据库邮件。
- 不支持修改数据库邮件配置参数。如欲查看预设（默认）值，请使用 sysmail_help_configure_sp 存储过程。
- 不完全支持文件附件。有关更多信息，请参阅“使用文件附件”。
- 文件附件的最大大小为 1 MB。
- 数据库邮件需要对多可用区数据库实例进行额外配置 有关更多信息，请参阅“多可用区部署的注意事项”。
- 不支持将 SQL Server Agent 配置为向预定义运算符发送电子邮件。

启用数据库邮件

使用以下过程为数据库实例启用数据库邮件：

1. 创建新的参数组。
2. 修改参数组以将 `database mail xps` 参数设置为 1。
3. 将参数组与数据库实例相关联。

为数据库邮件创建参数组

为与 SQL Server 版本和数据库实例版本对应的 `database mail xps` 参数创建参数组。

> **Note**
>
> 您也可以修改现有参数组。按照修改启用数据库邮件的参数中过程操作。

控制台

以下示例为 SQL Server 标准版 2016 创建一个参数组。

创建参数组

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择参数组。
3. 选择创建参数组。
4. 在创建参数组窗格中，执行以下操作：

   a. 对于参数组系列，选择 sqlserver-se-13.0。
   b. 对于组名称，输入参数组的标识符，如 **`dbmail-sqlserver-se-13`**。
   c. 对于描述，输入 **`Database Mail XPs`**。
5. 选择创建。

CLI

以下示例为 SQL Server 标准版 2016 创建一个参数组。

创建参数组

- 使用以下命令之一。

  Example

  对于 Linux、macOS 或 Unix：

  ```
  aws rds create-db-parameter-group \
  ```

```
    --db-parameter-group-name dbmail-sqlserver-se-13 \
    --db-parameter-group-family "sqlserver-se-13.0" \
    --description "Database Mail XPs"
```

对于 Windows：

```
aws rds create-db-parameter-group ^
    --db-parameter-group-name dbmail-sqlserver-se-13 ^
    --db-parameter-group-family "sqlserver-se-13.0" ^
    --description "Database Mail XPs"
```

修改启用数据库邮件的参数

修改与 SQL Server 版本和数据库实例版本对应的参数组中的 `database mail xps` 参数。

要启用数据库邮件，请将 `database mail xps` 参数设置为 1。

控制台

以下示例修改您为 SQL Server 标准版 2016 创建的参数组。

修改参数组

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择参数组。
3. 选择参数组，例如 dbmail-sqlserver-se-13。
4. 在参数下，从参数列表中筛选 **mail**。
5. 选择数据库邮件 xps。
6. 选择编辑参数。
7. 输入 **1**。
8. 选择 Save changes（保存更改）。

CLI

以下示例修改您为 SQL Server 标准版 2016 创建的参数组。

修改参数组

- 使用以下命令之一。

  Example

  对于 Linux、macOS 或 Unix：

  ```
  aws rds modify-db-parameter-group \
      --db-parameter-group-name dbmail-sqlserver-se-13 \
      --parameters "ParameterName='database mail
   xps',ParameterValue=1,ApplyMethod=immediate"
  ```

  对于 Windows：

  ```
  aws rds modify-db-parameter-group ^
      --db-parameter-group-name dbmail-sqlserver-se-13 ^
      --parameters "ParameterName='database mail
   xps',ParameterValue=1,ApplyMethod=immediate"
  ```

将参数组与数据库实例相关联

您可以使用 Amazon Web Services Management Console 或 Amazon CLI 将数据库邮件参数组与数据库实例相关联。

控制台

您可以将数据库邮件参数组与新的或现有的数据库实例相关联。

- 对于新的数据库实例，请在启动实例时将其关联。有关更多信息，请参阅"创建 Amazon RDS 数据库实例"。
- 对于现有数据库实例，请通过修改实例将其关联。有关更多信息，请参阅 修改 Amazon RDS 数据库实例。

CLI

您可以将数据库邮件参数组与新的或现有的数据库实例相关联。

如欲使用数据库邮件参数组创建数据库实例

- 指定在创建参数组时使用的相同数据库引擎类型和主要版本。

  Example

  对于 Linux、macOS 或 Unix：

```
aws rds create-db-instance \
    --db-instance-identifier mydbinstance \
    --db-instance-class db.m5.2xlarge \
    --engine sqlserver-se \
    --engine-version 13.00.5426.0.v1 \
    --allocated-storage 100 \
    --manage-master-user-password \
    --master-username admin \
    --storage-type gp2 \
    --license-model li
    --db-parameter-group-name dbmail-sqlserver-se-13
```

  对于 Windows：

```
aws rds create-db-instance ^
    --db-instance-identifier mydbinstance ^
    --db-instance-class db.m5.2xlarge ^
    --engine sqlserver-se ^
    --engine-version 13.00.5426.0.v1 ^
    --allocated-storage 100 ^
    --manage-master-user-password ^
    --master-username admin ^
    --storage-type gp2 ^
    --license-model li ^
    --db-parameter-group-name dbmail-sqlserver-se-13
```

如欲修改数据库实例并关联数据库邮件参数组

- 使用以下命令之一。

  Example

  对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier mydbinstance \
    --db-parameter-group-name dbmail-sqlserver-se-13 \
    --apply-immediately
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier mydbinstance ^
    --db-parameter-group-name dbmail-sqlserver-se-13 ^
    --apply-immediately
```

配置数据库邮件

您可以执行以下任务来配置数据库邮件：

1. 创建数据库邮件配置文件。
2. 创建数据库邮件账户。
3. 将数据库邮件账户添加到数据库邮件配置文件中。
4. 将用户添加到数据库邮件配置文件中。

> **Note**
>
> 如欲配置数据库邮件，请确保您对 `execute` 数据库中的存储过程拥有 `msdb` 权限。

创建数据库邮件配置文件

如欲创建数据库邮件配置文件，请使用 sysmail_add_profile_sp 存储过程。以下示例创建一个名为 `Notifications` 的配置文件。

如欲创建配置文件

- 使用以下 SQL 语句。

```
USE msdb
GO
```

```
EXECUTE msdb.dbo.sysmail_add_profile_sp
    @profile_name         = 'Notifications',
    @description          = 'Profile used for sending outgoing notifications using
 Amazon SES.';
GO
```

创建数据库邮件账户

如欲创建数据库邮件账户，请使用 sysmail_add_account_sp 存储过程。以下示例使用 Amazon Simple Email Service 在私有 VPC 中的 RDS for SQL Server 数据库实例上创建一个名为 SES 的账户。

使用 Amazon SES 需要以下参数：

- `@email_address` – 经过 Amazon SES 验证的身份。有关更多信息，请参阅在 Amazon SES 中验证身份。
- `@mailserver_name` – Amazon SES SMTP 端点。有关更多信息，请参阅连接到 Amazon SES SMTP 端点。
- `@username` – Amazon SES SMTP 用户名。有关更多信息，请参阅获取 Amazon SES SMTP 凭证。

  请勿使用Amazon Identity and Access Management用户名。
- `@password` – Amazon SES SMTP 密码。有关更多信息，请参阅获取 Amazon SES SMTP 凭证。

如欲创建账户

- 使用以下 SQL 语句。

```
USE msdb
GO

EXECUTE msdb.dbo.sysmail_add_account_sp
    @account_name        = 'SES',
    @description         = 'Mail account for sending outgoing notifications.',
    @email_address       = 'nobody@example.com',
    @display_name        = 'Automated Mailer',
    @mailserver_name     = 'vpce-0a1b2c3d4e5f-01234567.email-smtp.us-
west-2.vpce.amazonaws.com',
    @port                = 587,
    @enable_ssl          = 1,
```

```
    @username             = 'Smtp_Username',
    @password             = 'Smtp_Password';
GO
```

Note

作为安全最佳实践，请指定除此处所示提示以外的凭证。

将数据库邮件账户添加到数据库邮件配置文件中

如欲将数据库邮件账户添加到数据库邮件配置文件中，请使用 sysmail_add_profileaccount_sp 存储过程。以下示例将 SES 账户添加到 Notifications 配置文件中。

如欲将账户添加到配置文件中

- 使用以下 SQL 语句。

```
USE msdb
GO

EXECUTE msdb.dbo.sysmail_add_profileaccount_sp
    @profile_name         = 'Notifications',
    @account_name         = 'SES',
    @sequence_number      = 1;
GO
```

将用户添加到数据库邮件配置文件中

如欲向 msdb 数据库委托人授予使用数据库邮件配置文件的权限，请使用 sysmail_add_principalprofile_sp 存储过程。委托人指可以请求获取 SQL Server 资源的实体。数据库委托人必须映射到 SQL Server 身份验证用户、Windows 身份验证用户或 Windows 身份验证组。

以下示例授予对 Notifications 配置文件的公有访问权限。

如欲将用户添加到配置文件

- 使用以下 SQL 语句。

```
USE msdb
```

```
GO

EXECUTE msdb.dbo.sysmail_add_principalprofile_sp
    @profile_name       = 'Notifications',
    @principal_name     = 'public',
    @is_default         = 1;
GO
```

数据库邮件的 Amazon RDS 存储过程和函数

Microsoft 提供存储过程用于使用数据库邮件，例如创建、列出、更新和删除帐户和配置文件。此外，RDS 还提供了数据库邮件的存储过程和函数，如下表所示。

过程/功能	描述
rds_fn_sysmail_allitems	显示已发送的消息，包括其他用户提交的消息。
rds_fn_sysmail_event_log	显示事件，包括其他用户提交的消息事件。
rds_fn_sysmail_mailattachments	显示附件，包括其他用户提交的消息附件。
rds_sysmail_control	启动和停止邮件队列（DatabaseMail.exe 进程）。
rds_sysmail_delete_mailitems_sp	从数据库邮件内部表中删除所有用户发送的电子邮件。

使用数据库邮件发送电子邮件

您可以使用数据库邮件的 sp_send_dbmail 存储过程发送电子邮件。

使用量

```
EXEC msdb.dbo.sp_send_dbmail
@profile_name = 'profile_name',
@recipients = 'recipient1@example.com[; recipient2; ... recipientn]',
@subject = 'subject',
@body = 'message_body',
[@body_format = 'HTML'],
[@file_attachments = 'file_path1; file_path2; ... file_pathn'],
```

```
[@query = 'SQL_query'],
[@attach_query_result_as_file = 0|1]';
```

以下参数为必需参数：

- @profile_name – 要从中发送消息的数据库邮件配置文件的名称。
- @recipients – 要向其发送消息的电子邮件地址列表，用分号分隔。
- @subject – 消息主题。
- @body – 消息的正文。您也可以使用声明的变量作为主体。

以下参数为可选参数：

- @body_format – 此参数与声明的变量一起使用，从而以 HTML 格式发送电子邮件。
- @file_attachments – 以分号分隔的消息附件列表。文件路径必须是绝对路径。
- @query – 要运行的 SQL 查询。查询结果可以作为文件附加，也可以包含在消息正文中。
- @attach_query_result_as_file – 是否将查询结果附加为文件。设置为 0 表示“否”，设置为 1 表示“是”。默认值是 0。

示例

以下示例演示了如何发送电子邮件。

Example 向单个收件人发送消息

```
USE msdb
GO

EXEC msdb.dbo.sp_send_dbmail
     @profile_name       = 'Notifications',
     @recipients         = 'nobody@example.com',
     @subject            = 'Automated DBMail message - 1',
     @body               = 'Database Mail configuration was successful.';
GO
```

Example 向多个收件人发送消息

```
USE msdb
GO

```

```
EXEC msdb.dbo.sp_send_dbmail
     @profile_name       = 'Notifications',
     @recipients         = 'recipient1@example.com;recipient2@example.com',
     @subject            = 'Automated DBMail message - 2',
     @body               = 'This is a message.';
GO
```

Example 将 SQL 查询结果作为文件附件发送

```
USE msdb
GO

EXEC msdb.dbo.sp_send_dbmail
     @profile_name       = 'Notifications',
     @recipients         = 'nobody@example.com',
     @subject            = 'Test SQL query',
     @body               = 'This is a SQL query test.',
     @query              = 'SELECT * FROM abc.dbo.test',
     @attach_query_result_as_file = 1;
GO
```

Example 以 HTML 格式发送消息

```
USE msdb
GO

DECLARE @HTML_Body as NVARCHAR(500) = 'Hi, <h4> Heading </h4> </br> See the report. <b>
 Regards </b>';

EXEC msdb.dbo.sp_send_dbmail
     @profile_name       = 'Notifications',
     @recipients         = 'nobody@example.com',
     @subject            = 'Test HTML message',
     @body               = @HTML_Body,
     @body_format        = 'HTML';
GO
```

Example 数据库中发生特定事件时，使用触发器发送消息

```
USE AdventureWorks2017
GO
IF OBJECT_ID ('Production.iProductNotification', 'TR') IS NOT NULL
```

```
DROP TRIGGER Purchasing.iProductNotification
GO

CREATE TRIGGER iProductNotification ON Production.Product
   FOR INSERT
   AS
   DECLARE @ProductInformation nvarchar(255);
   SELECT
   @ProductInformation = 'A new product, ' + Name + ', is now available for $' +
 CAST(StandardCost AS nvarchar(20)) + '!'
   FROM INSERTED i;

EXEC msdb.dbo.sp_send_dbmail
     @profile_name       = 'Notifications',
     @recipients         = 'nobody@example.com',
     @subject            = 'New product information',
     @body               = @ProductInformation;
GO
```

查看消息、日志和附件

您可以使用 RDS 存储过程查看消息、事件日志和附件。

如欲查看所有电子邮件

- 使用以下 SQL 查询。

```
SELECT * FROM msdb.dbo.rds_fn_sysmail_allitems(); --WHERE sent_status='sent' or
 'failed' or 'unsent'
```

如欲查看所有电子邮件事件日志

- 使用以下 SQL 查询。

```
SELECT * FROM msdb.dbo.rds_fn_sysmail_event_log();
```

如欲查看所有电子邮件附件

- 使用以下 SQL 查询。

```
SELECT * FROM msdb.dbo.rds_fn_sysmail_mailattachments();
```

删除消息

您可以使用 `rds_sysmail_delete_mailitems_sp` 存储过程删除消息。

> **Note**
>
> 当 DBMail 历史数据大小达到 1 GB 时，RDS 会自动删除邮件表项目，且保留期至少为 24 小时。
>
> 如果您希望延长邮件项目的保留时间，可以将其存档。有关详细信息，请参阅 Microsoft 文档中的创建 SQL Server Agent 作业以对数据库邮件消息和事件日志进行存档。

如欲删除所有电子邮件

- 使用以下 SQL 语句。

```
DECLARE @GETDATE datetime
SET @GETDATE = GETDATE();
EXECUTE msdb.dbo.rds_sysmail_delete_mailitems_sp @sent_before = @GETDATE;
GO
```

如欲删除具有特定状态的所有电子邮件

- 使用以下 SQL 语句删除所有失败的消息。

```
DECLARE @GETDATE datetime
SET @GETDATE = GETDATE();
EXECUTE msdb.dbo.rds_sysmail_delete_mailitems_sp @sent_status = 'failed';
GO
```

启动和停止邮件队列

使用以下说明启动和停止数据库邮件队列：

主题

- [启动邮件队列]
- [停止邮件队列]

启动邮件队列

您可以使用 `rds_sysmail_control` 存储过程启动数据库邮件过程。

> **Note**
> 启用数据库邮件会自动启动邮件队列。

如欲启动邮件队列

- 使用以下 SQL 语句。

```
EXECUTE msdb.dbo.rds_sysmail_control start;
GO
```

停止邮件队列

您可以使用 `rds_sysmail_control` 存储过程停止数据库邮件过程。

如欲停止邮件队列

- 使用以下 SQL 语句。

```
EXECUTE msdb.dbo.rds_sysmail_control stop;
GO
```

使用文件附件

在 SQL Server 上，来自 RDS 的数据库邮件消息不支持以下文件扩展名：.ade、.adp、.apk、.appx、.appxbundle、.bat、.bak、.cab、.chm、.cmd、.com、.cpl、.dll、.dmg、.ex 和.wsh。

数据库邮件使用当前用户的 Microsoft Windows 安全环境来控制对文件的访问。使用 SQL Server 身份验证登录的用户无法使用带 `@file_attachments` 存储过程的 `sp_send_dbmail` 参数来附加文

件。Windows 不允许 SQL Server 从一台远程计算机向另一台远程计算机提供凭证。因此，当命令从运行 SQL Server 的计算机以外的计算机运行时，数据库邮件无法从网络共享中的附加文件。

但是，您可以使用 SQL Server Agent 作业来附加文件。有关 SQL Server Agent 的详细信息，请参阅 Microsoft 文档中的 使用 SQL Server Agent for Amazon RDS 和 SQL Server Agent。

多可用区部署的注意事项

在多可用区数据库实例上配置数据库邮件时，配置不会自动传播到辅助节点。我们建议将多可用区实例转换为单可用区实例，配置数据库邮件，然后将数据库实例转换回多可用区实例。然后，主节点和辅助节点都具有数据库邮件配置。

如果您从配置了数据库邮件的多可用区实例中创建只读副本，则副本将继承配置，但不会向 SMTP 服务器提供密码。使用密码更新数据库邮件账户。

删除 SMTP（端口 25）限制

默认情况下，Amazon 会阻止 RDS for SQL Server 数据库实例的 SMTP（端口 25）上的出站流量。这样做是为了根据弹性网络接口所有者的策略来阻止垃圾邮件。如果需要，您可以删除此限制。有关更多信息，请参阅如何删除 Amazon EC2 实例或 Lambda 函数的端口 25 的限制？。

Amazon RDS for SQL Server 上 的 tempdb 数据库的实例存储支持。

实例存储为数据库实例提供临时性块级存储。此存储位于已物理附加到主机的磁盘上。这些磁盘具有基于固态硬盘 (SSD) 的非易失性存储规范(NVMe) 实例存储。此存储针对低延迟、极高随机 I/O 性能和高速连续读取吞吐量进行了优化。

与基于 Amazon EBS 的标准存储相比，将 `tempdb` 数据文件和 `tempdb` 日志文件放置于实例存储之上可实现更低的读写延迟。

> **Note**
>
> SQL Server 数据库文件和数据库日志文件不会放置于实例存储之上。

启用实例存储

当 RDS 预置具有以下实例类之一的数据库实例时，会自动将 `tempdb` 数据库放置到实例存储中：

- db.m5d
- db.r5d
- db.x2iedn

要启用实例存储，请执行下列操作之一：

- 使用这些实例类型其中之一创建 SQL Server 数据库实例。有关更多信息，请参阅"创建 Amazon RDS 数据库实例"。
- 修改现有 SQL Server 数据库实例以使用其中一个实例。有关更多信息，请参阅"修改 Amazon RDS 数据库实例"。

在支持其中一种或多种实例类型的所有 Amazon 区域均可使用实例存储。有关 `db.m5d` 和 `db.r5d` 实例类的更多信息，请参阅 数据库实例类。有关 Amazon RDS for SQL Server 支持的实例类的更多信息，请参阅 Microsoft SQL Server 的数据库实例类支持。

文件位置和大小注意事项

在没有实例存储的实例上，RDS 将 `tempdb` 数据和日志文件存储在 `D:\rdsdbdata\DATA` 目录中。默认情况下，两个文件的初始大小均为 8 MB。

在具有实例存储的实例上，RDS 将 `tempdb` 数据和日志文件存储在 `T:\rdsdbdata\DATA` 目录中。

当 `tempdb` 只有一个数据文件 (`tempdb.mdf`) 和一个日志文件 (`templog.ldf`)时，默认情况下，`templog.ldf` 的初始大小为 8 MB，而 `tempdb.mdf` 的初始大小为实例存储容量的 80% 或更多。存储容量的 20% 或 200 GB（以较低者为准）可以免费作为初始大小。多个 `tempdb` 数据文件均匀分配 80% 的磁盘空间，而日志文件的初始大小始终为 8 MB。

例如，如果将数据库实例类从 `db.m5.2xlarge` 修改为 `db.m5d.2xlarge`，则 `tempdb` 数据文件的大小从每个 8 MB 增加到总共 234 GB。

> **Note**
>
> 除了实例存储 (`tempdb`) 上的 `T:\rdsdbdata\DATA` 数据和日志文件，您还可以在数据卷 (`tempdb`) 上创建额外的 `D:\rdsdbdata\DATA` 数据和日志文件。这些文件的初始大小始终为 8 MB。

备份注意事项

您可能需要长时间保留备份，这会随着时间的推移产生成本。`tempdb` 数据和日志数据块可能会经常发生变化，具体取决于工作负载。这可以大大增加数据库快照的大小。

`tempdb` 位于实例存储上时，快照不包括临时文件。这意味着与仅限 EBS 的存储相比，快照大小更小，占用的免费备份分配更少。

磁盘已满错误

如果您使用了实例存储中的所有可用空间，则可能会收到以下等错误：

- The transaction log for database 'tempdb' is full due to 'ACTIVE_TRANSACTION'.(由于“ACTIVE_TRANSACTION”，数据库“tempdb”的事务日志已满。)
- Could not allocate space for object 'dbo.SORT temporary run storage: 140738941419520' in database 'tempdb' because the 'PRIMARY' filegroup is full. (由于 “PRIMARY”文件组已满，无法为数据库“tempdb”中的对象“dbo.SORT temporary run storage: 140738941419520”分配空间。) Create disk space by deleting unneeded files, dropping objects in the filegroup, adding additional files to the filegroup, or setting autogrowth on for existing files in the filegroup. (通过删除不需要的文件、删除文件组中的对象、向文件组添加其他文件或为文件组中的现有文件设置自动增长来增加磁盘空间。)

实例存储已满时，您可以执行以下一项或多项操作：

- 调整工作负载或使用方式 `tempdb`。
- 向上扩展以使用具有更多 NVMe 存储空间的数据库实例类。
- 停止使用实例存储，然后使用仅限 EBS 存储的实例类。
- 通过为 EBS 卷上的 `tempdb` 添加辅助数据或日志文件来使用混合模式。

删除实例存储

要删除实例存储，请修改 SQL Server 数据库实例以使用不支持实例存储的实例类型，如 db.m5、db.r5 或 db.x1e。

> **Note**
>
> 删除实例存储时，会将临时文件移至 `D:\rdsdbdata\DATA` 目录，并将其大小缩减到 8 MB。

通过 Amazon RDS for Microsoft SQL Server 使用扩展事件。

您可以使用 Microsoft SQL Server 中的扩展事件来捕获 Amazon RDS for SQL Server 的调试和故障排除信息。扩展事件取代了已被微软弃用的 SQL Trace 和 Server Profiler。扩展事件类似于 Profiler 跟踪，但对跟踪的事件进行更精细的控制。Amazon RDS 上的 SQL Server 版本 2016 及更高版本支持扩展事件。有关更多信息，请参阅微软文档中的扩展事件概述。

对于在 Amazon RDS for SQL Server 中具有主用户权限的用户，将自动打开扩展事件。

主题

限制和建议

在 RDS for SQL Server 上使用扩展事件时，适用以下限制：

- 仅企业版和标准版支持扩展事件。
- 您无法更改默认的扩展事件会话。
- 确保将会话内存分区模式设置为`NONE`。
- 会话事件保留模式可以是`ALLOW_SINGLE_EVENT_LOSS`或`ALLOW_MULTIPLE_EVENT_LOSS`。
- 不支持 Event Tracing for Windows (ETW) 目标。
- 确保文件目标位于`D:\rdsdbdata\log`目录中。
- 为了匹配目标，请将`respond_to_memory_pressure`属性设置为`1`。
- 环形缓冲区目标内存不能大于 4 MB。
- 不支持以下操作：
  - `debug_break`
  - `create_dump_all_threads`
  - `create_dump_single_threads`
- 以下版本及更高版本支持该 `rpc_completed` 活动：15.0.4083.2、14.0.3370.1、13.0.5865.1、12.0.6433.1、11.0.7507.2。

在 RDS for SQL Server 上配置扩展事件

在 RDS for SQL Server 上，您可以配置扩展事件会话的某些参数的值。下表介绍了可配置的参数。

参数名称	描述
xe_session_max_memory	指定分配给会话以进行事件缓冲的最大内存量。此值对应
xe_session_max_event_size	指定允许用于大型事件的最大内存大小。此值对应于事件
xe_session_max_dispatch_latency	指定事件在被发送到扩展事件会话目标之前在内存中缓冲 tch_latency 设置。
xe_file_target_size	指定文件目标的最大大小。此值对应于文件目标的max_fi
xe_file_retention	指定事件会话的文件目标生成的文件的保留天数。

> **Note**
>
> 通过将xe_file_retention设置为零，.xel 文件将在 SQL Server 解锁后自动删除。当 .xel 文件达到 xe_file_target_size 中设置的大小限制时，就会解锁。

您可以使用rdsadmin.dbo.rds_show_configuration存储过程显示这些参数的当前值。例如，使用以下 SQL 语句查看xe_session_max_memory的当前设置。

```
exec rdsadmin.dbo.rds_show_configuration 'xe_session_max_memory'
```

您可以使用rdsadmin.dbo.rds_set_configuration存储过程对其进行修改。例如，使用以下 SQL 语句将xe_session_max_memory设置为 4 MB。

```
exec rdsadmin.dbo.rds_set_configuration 'xe_session_max_memory', 4
```

多可用区部署的注意事项

在主数据库实例上创建扩展事件会话时，它不会传播到备用副本。您可以在新的主数据库实例上进行故障转移并创建扩展事件会话。您也可以先删除，然后再重新添加多可用区配置，以将扩展事件会话传播

到备用副本。RDS 停止备用副本上的所有非默认扩展事件会话，以使这些会话不会消耗备用副本上的资源。因此，在备用副本成为主数据库实例后，确保在新的主数据库实例上手动启动扩展事件会话。

> **Note**
>
> 此方法同时适用于 Always On 可用性组和数据库镜像。

您还可以使用 SQL Server Agent 作业跟踪备用副本，并在备用副本成为主数据库实例时启动会话。例如，在 SQL Server Agent 作业步骤中使用以下查询重新启动主数据库实例上的事件会话。

```
BEGIN
    IF (DATABASEPROPERTYEX('rdsadmin','Updateability')='READ_WRITE'
    AND DATABASEPROPERTYEX('rdsadmin','status')='ONLINE'
    AND (DATABASEPROPERTYEX('rdsadmin','Collation') IS NOT NULL OR
 DATABASEPROPERTYEX('rdsadmin','IsAutoClose')=1)
    )
    BEGIN
        IF NOT EXISTS (SELECT 1 FROM sys.dm_xe_sessions WHERE name='xe1')
            ALTER EVENT SESSION xe1 ON SERVER STATE=START
        IF NOT EXISTS (SELECT 1 FROM sys.dm_xe_sessions WHERE name='xe2')
            ALTER EVENT SESSION xe2 ON SERVER STATE=START
    END
END
```

如果这些会话处于停止状态，则此查询将在主数据库实例上重新启动事件会话`xe1`和`xe2`。您还可以为此查询添加一个时间间隔比较方便的计划。

查询扩展事件文件

您可以使用 SQL Server Management Studio 或`sys.fn_xe_file_target_read_file`函数来查看使用文件目标的扩展事件中的数据。有关此函数的更多信息，请参阅微软文档中的[sys.fn_xe_file_target_read_file (Transact-SQL)](#)。

扩展事件文件目标只能将文件写入 RDS for SQL Server 上的 `D:\rdsdbdata\log` 目录。

例如，使用以下 SQL 查询列出名称以`xe`开头的扩展事件会话的所有文件的内容。

```
SELECT * FROM sys.fn_xe_file_target_read_file('d:\rdsdbdata\log\xe*', null,null,null);
```

使用 RDS for SQL Server 访问事务日志备份

通过访问 RDS for SQL Server 的事务日志备份，您可以列出数据库的事务日志备份文件并将它们复制到目标 Amazon S3 桶。通过在 Amazon S3 桶中复制事务日志备份，您可以将它们与完整和差异数据库备份结合使用，以执行时间点数据库还原。您可以使用 RDS 存储过程设置对事务日志备份的访问权限，列出可用的事务日志备份，并将它们复制到您的 Amazon S3 桶。

访问事务日志备份可提供以下功能和好处：

- 列出并查看 RDS for SQL Server 数据库实例上数据库的可用事务日志备份的元数据。
- 将可用事务日志备份从 RDS for SQL Server 复制到目标 Amazon S3 桶。
- 无需还原整个数据库实例，即可执行数据库的时间点还原。有关将数据库实例还原到某个时间点的更多信息，请参阅将 Amazon RDS 的数据库实例还原到指定时间。

可用性和支持

所有 Amazon 区域都支持访问事务日志备份。访问事务日志备份适用于 Amazon RDS 支持的所有 Microsoft SQL Server 版本。

要求

在启用访问事务日志备份之前，必须满足以下要求：

- 必须在数据库实例上启用自动备份，并且备份保留期必须设置为一天或多天的值。有关启用自动备份和配置保留策略的更多信息，请参阅启用自动备份。
- Amazon S3 桶必须与源数据库实例存在于同一账户和区域中。在启用访问事务日志备份之前，请选择现有的 Amazon S3 桶或创建新桶，用于存储事务日志备份文件。
- 必须按如下方式配置 Amazon S3 桶权限策略，以允许 Amazon RDS 将事务日志文件复制到其中：
  1. 将桶的对象账户拥有权属性设置为 Bucket Owner Preferred（首选桶拥有者）。
  2. 添加以下策略。原定设置情况下没有策略，因此，请使用桶访问控制列表（ACL）编辑并添加桶策略。

  以下示例使用 ARN 指定资源。我们建议在基于资源的信任关系中使用 `SourceArn` 和 `SourceAccount` 全局条件上下文键，以此限制服务对特定资源的权限。有关使用 ARN 的更多信息，请参阅 Amazon 资源名称（ARN）和Amazon RDS 中的 Amazon 资源名称（ARN）。

Example 用于访问事务日志备份的 Amazon S3 权限策略

```
    {
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Only allow writes to my bucket with bucket owner full control",
            "Effect": "Allow",
            "Principal": {
                "Service": "backups.rds.amazonaws.com"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/{customer_path}/*",
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl": "bucket-owner-full-control",
                    "aws:sourceAccount": "{customer_account}",
                    "aws:sourceArn": "{db_instance_arn}"
                }
            }
        }
    ]
}
```

- 用于访问 Amazon S3 桶的 Amazon Identity and Access Management（IAM）角色。如果您已有一个 IAM 角色，则可以使用它。您可以选择在使用Amazon Web Services Management Console添加 SQLSERVER_BACKUP_RESTORE 选项时为自己创建新的 IAM 角色。或者，您可以手动创建一个新的角色。有关使用 SQLSERVER_BACKUP_RESTORE 创建和配置 IAM 角色的更多信息，请参阅为本机备份和还原手动创建 IAM 角色。
- 必须将 SQLSERVER_BACKUP_RESTORE 选项添加到数据库实例上的选项组。有关添加 SQLSERVER_BACKUP_RESTORE 选项的更多信息，请参阅SQL Server 中对本机备份和还原的支持。

> **Note**
>
> 如果您的数据库实例启用了存储加密，则必须以原生备份和还原选项组中提供的 IAM 角色提供 Amazon KMS（KMS）操作和密钥。

（可选）如果您打算使用 `rds_restore_log` 存储过程执行时间点数据库还原，我们建议将相同的 Amazon S3 路径用于原生备份和还原选项组以及访问事务日志备份。此方法可确保当 Amazon RDS 代入选项组中的角色以执行还原日志功能时，它有权从相同的 Amazon S3 路径检索事务日志备份。

- 如果数据库实例已加密，则无论加密类型如何（Amazon 托管密钥或客户托管密钥），您都必须在 IAM 角色和 `rds_tlog_backup_copy_to_S3` 存储过程中提供客户托管的 KMS 密钥。

限制和建议

访问事务日志备份有以下限制和建议：

- 对于任何配置了 1 到 35 天备份保留期的数据库实例，您最多可以列出并复制最近七天的事务日志备份。
- 用于访问事务日志备份的 Amazon S3 桶必须与源数据库实例位于相同的账户和区域中。不支持跨账户和跨区域复制。
- 只能将一个 Amazon S3 桶配置为将事务日志备份复制到的目标。您可以使用 `rds_tlog_copy_setup` 存储过程选择新的目标 Amazon S3 桶。有关选择新目标 Amazon S3 桶的更多信息，请参阅设置访问事务日志备份。
- 如果您的 RDS 实例未启用存储加密，则在使用 `rds_tlog_backup_copy_to_S3` 存储过程时无法指定 KMS 密钥。
- 不支持多账户复制。用于复制的 IAM 角色仅允许对数据库实例的拥有者账户中的 Amazon S3 桶拥有写入权限。
- 在 RDS for SQL Server 数据库实例上只能运行任何类型的两个并发任务。
- 在给定时间只能为单个数据库运行一个复制任务。如果要复制数据库实例上多个数据库的事务日志备份，请对每个数据库使用单独的复制任务。
- 如果您复制 Amazon S3 桶中已存在的同名事务日志备份，则现有的事务日志备份将被覆盖。
- 您只能在主数据库实例上运行可访问事务日志备份的存储过程。您无法在 RDS for SQL Server 只读副本或多可用区数据库集群的辅助实例上运行这些存储过程。
- 如果在 `rds_tlog_backup_copy_to_S3` 存储过程运行时重启 RDS for SQL Server 数据库实例，则当数据库实例恢复联机时，任务将自动从头开始重新启动。重启前任务运行时已复制到 Amazon S3 桶的任何事务日志备份都将被覆盖。
- 无法将 Microsoft SQL Server 系统数据库和 `RDSAdmin` 数据库配置为访问事务日志备份。
- 不支持复制到由 SSE-KMS 加密的桶。

设置访问事务日志备份

要设置访问事务日志备份，请完成要求部分中的要求列表，然后运行 `rds_tlog_copy_setup` 存储过程。该过程将在数据库实例级别启用访问事务日志备份功能。您无需为数据库实例上的每个单独数据库运行该过程。

> **Important**
>
> 必须在 SQL Server 中向数据库用户授予对于每个数据库的 `db_owner` 角色，才能配置和使用访问事务日志备份功能。

Example 用法：

```
exec msdb.dbo.rds_tlog_copy_setup
@target_s3_arn='arn:aws:s3:::amzn-s3-demo-bucket/myfolder';
```

以下参数是必需参数：

- `@target_s3_arn` – 要将事务日志备份文件复制到的目标 Amazon S3 桶的 ARN。

Example 设置 Amazon S3 目标桶：

```
exec msdb.dbo.rds_tlog_copy_setup @target_s3_arn='arn:aws:s3:::amzn-s3-demo-logging-bucket/mytestdb1';
```

要验证配置，请调用 `rds_show_configuration` 存储过程。

Example 验证配置：

```
exec rdsadmin.dbo.rds_show_configuration @name='target_s3_arn_for_tlog_copy';
```

要修改对事务日志备份的访问权限以指向其他 Amazon S3 桶，您可以查看当前 Amazon S3 桶值，并使用 `@target_s3_arn` 的新值重新运行 `rds_tlog_copy_setup` 存储过程。

Example 查看为访问事务日志备份而配置的现有 Amazon S3 桶

```
exec rdsadmin.dbo.rds_show_configuration @name='target_s3_arn_for_tlog_copy';
```

Example 更新到新的目标 Amazon S3 桶

```
exec msdb.dbo.rds_tlog_copy_setup @target_s3_arn='arn:aws:s3:::amzn-s3-demo-logging-
bucket1/mynewfolder';
```

列出可用的事务日志备份

在 RDS for SQL Server 中，配置为使用完整恢复模式并将数据库实例备份保留期设置为一天或多天的数据库实例备份会自动启用事务日志备份。通过启用访问事务日志备份，您可以将多达七天的这些事务日志备份复制到 Amazon S3 桶中。

启用访问事务日志备份后，就可以开始使用它来列出和复制可用的事务日志备份文件。

列出事务日志备份

要列出可用于单个数据库的所有事务日志备份，请调用 `rds_fn_list_tlog_backup_metadata` 函数。调用函数时可以使用 `ORDER BY` 或 `WHERE` 子句。

Example 列出和筛选可用的事务日志备份文件

```
SELECT * from msdb.dbo.rds_fn_list_tlog_backup_metadata('mydatabasename');
SELECT * from msdb.dbo.rds_fn_list_tlog_backup_metadata('mydatabasename') WHERE
 rds_backup_seq_id = 3507;
SELECT * from msdb.dbo.rds_fn_list_tlog_backup_metadata('mydatabasename') WHERE
 backup_file_time_utc > '2022-09-15 20:44:01' ORDER BY backup_file_time_utc DESC;
```

100 %

Results Messages

	db_name	db_id	family_guid	rds_backup_seq_id	backup_file_epoch	backup_file_time_utc	starting_lsn	ending_lsn	is_log_chain_broken	file_size_bytes	Error
1	tpcc	6	CD11CB3D-B5E4-46D9-B462-CE40CDA97E89	43	1661846641	2022-08-30 08:04:01	545000085730100001	545000085731000001	0	35564	NULL
2	tpcc	6	CD11CB3D-B5E4-46D9-B462-CE40CDA97E89	44	1661846941	2022-08-30 08:09:01	545000085731000001	545000085731900001	0	35473	NULL
3	tpcc	6	CD11CB3D-B5E4-46D9-B462-CE40CDA97E89	45	1661847241	2022-08-30 08:14:01	545000085731900001	545000085732800001	0	35394	NULL
4	tpcc	6	CD11CB3D-B5E4-46D9-B462-CE40CDA97E89	46	1661847541	2022-08-30 08:19:01	545000085732800001	545000085733700001	0	35374	NULL
5	tpcc	6	CD11CB3D-B5E4-46D9-B462-CE40CDA97E89	47	1661847841	2022-08-30 08:24:01	545000085733700001	545000085734600001	0	35601	NULL
6	tpcc	6	CD11CB3D-B5E4-46D9-B462-CE40CDA97E89	48	1661848142	2022-08-30 08:29:02	545000085734600001	545000085735500001	0	35470	NULL
7	tpcc	6	CD11CB3D-B5E4-46D9-B462-CE40CDA97E89	49	1661848441	2022-08-30 08:34:01	545000085735500001	545000085736400001	0	35491	NULL
8	tpcc	6	CD11CB3D-B5E4-46D9-B462-CE40CDA97E89	50	1661848741	2022-08-30 08:39:01	545000085736400001	545000085737300001	0	35520	NULL
9	tpcc	6	CD11CB3D-B5E4-46D9-B462-CE40CDA97E89	51	1661849041	2022-08-30 08:44:01	545000085737300001	545000085738200001	0	35326	NULL
10	tpcc	6	CD11CB3D-B5E4-46D9-B462-CE40CDA97E89	52	1661849341	2022-08-30 08:49:01	545000085738200001	545000085739100001	0	35407	NULL
11	tpcc	6	CD11CB3D-B5E4-46D9-B462-CE40CDA97E89	53	1661849641	2022-08-30 08:54:01	545000085739100001	545000085740000001	0	35491	NULL
12	tpcc	6	CD11CB3D-B5E4-46D9-B462-CE40CDA97E89	54	1661849941	2022-08-30 08:59:01	545000085740000001	545000085740900001	0	35438	NULL
13	tpcc	6	CD11CB3D-B5E4-46D9-B462-CE40CDA97E89	55	1661850241	2022-08-30 09:04:01	545000085740900001	545000085741800001	0	35319	NULL
14	tpcc	6	CD11CB3D-B5E4-46D9-B462-CE40CDA97E89	56	1661850541	2022-08-30 09:09:01	545000085741800001	545000085742700001	0	35270	NULL
15	tpcc	6	CD11CB3D-B5E4-46D9-B462-CE40CDA97E89	57	1661850841	2022-08-30 09:14:01	545000085742700001	545000085743600001	0	35476	NULL

`rds_fn_list_tlog_backup_metadata` 函数返回以下输出：

列名称	数据类型	描述
`db_name`	sysname	为列出其事务日志备份而提供的数据库名称。
`db_id`	int	输入参数 `db_name` 的内部数据库标识符。
`family_guid`	uniqueidentifier	创建时原始数据库的唯一 ID。还原数据库时，该值保持不变，即使还原到不同的数据库名称也是如此。
`rds_backup_seq_id`	int	RDS 在内部用于维护每个事务日志备份文件的序列号的 ID。
`backup_file_epoch`	bigint	生成事务备份文件的纪元时间。
`backup_file_time_utc`	datetime	`backup_file_epoch` 值的 UTC 时间转换值。
`starting_lsn`	numeric(25,0)	事务日志备份文件的第一条或最早的日志记录的日志序列号。
`ending_lsn`	numeric(25,0)	事务日志备份文件的上一条或下一条日志记录的日志序列号。
`is_log_chain_broken`	bit	一个布尔值，表示当前事务日志备份文件和先前的事务日志备份文件之间的日志链是否断开。

列名称	数据类型	描述
file_size _bytes	bigint	事务备份集的大小（以字节为单位）。
Error	varchar(4000)	如果 rds_fn_list_tlog_backup_met adata 函数引发异常，则显示错误消息。如果没有异常，则为 NULL。

复制事务日志备份

要将单个数据库的一组可用事务日志备份复制到您的 Amazon S3 桶，请调用 rds_tlog_backup_copy_to_S3 存储过程。rds_tlog_backup_copy_to_S3 存储过程将启动一项新任务以复制事务日志备份。

> **Note**
>
> rds_tlog_backup_copy_to_S3 存储过程将在不对 is_log_chain_broken 属性进行验证的情况下复制事务日志备份。因此，在运行 rds_tlog_backup_copy_to_S3 存储过程之前，您应该手动确认未断开的日志链。有关进一步的解释，请参阅验证事务日志备份日志链。

Example **rds_tlog_backup_copy_to_S3** 存储过程的使用

```
exec msdb.dbo.rds_tlog_backup_copy_to_S3
 @db_name='mydatabasename',
 [@kms_key_arn='arn:aws:kms:region:account-id:key/key-id'],
 [@backup_file_start_time='2022-09-01 01:00:15'],
 [@backup_file_end_time='2022-09-01 21:30:45'],
 [@starting_lsn=149000000112100001],
 [@ending_lsn=149000000120400001],
 [@rds_backup_starting_seq_id=5],
 [@rds_backup_ending_seq_id=10];
```

以下输入参数可用：

参数	描述
@db_name	要为其复制事务日志备份的数据库的名称
@kms_key_arn	客户托管的 KMS 密钥。如果您使用 Amazon 托管的 KMS 密钥加密数据库实例，则必须创建客户托管的密钥。如果您使用客户托管的密钥加密数据库实例，则可以使用相同的 KMS 密钥 ARN。
@backup_file_start_time	rds_fn_list_tlog_backup_metadata 函数的 [backup_file_time_utc] 列中提供的 UTC 时间戳。
@backup_file_end_time	rds_fn_list_tlog_backup_metadata 函数的 [backup_file_time_utc] 列中提供的 UTC 时间戳。
@starting_lsn	rds_fn_list_tlog_backup_metadata 函数的 [starting_lsn] 列中提供的日志序列号（LSN）
@ending_lsn	rds_fn_list_tlog_backup_metadata 函数的 [ending_lsn] 列中提供的日志序列号（LSN）。
@rds_backup_starting_seq_id	rds_fn_list_tlog_backup_metadata 函数的 [rds_backup_seq_id] 列中提供的序列 ID。
@rds_backup_ending_seq_id	rds_fn_list_tlog_backup_metadata 函数的 [rds_backup_seq_id] 列中提供的序列 ID。

您可以指定一组时间、LSN 或序列 ID 参数。只需要一组参数。

您也可以在任何参数集中仅指定一个参数。例如，通过仅为 backup_file_end_time 参数提供值，则该时间之前在七天限制内的所有可用事务日志备份文件都将复制到您的 Amazon S3 桶中。

以下是 rds_tlog_backup_copy_to_S3 存储过程的有效输入参数组合。

提供的参数	预期结果	
`exec msdb.dbo.rds_tlog_backup_copy_to_S3`	复制最近七天且在提供的 backup_file_start_time	

提供的参数	预期结果
`@db_name = 'testdb1', @backup_file_start_time='2022-08-23 00:00:00', @backup_file_end_time='2022-08-30 00:00:00';`	和 `backup_file_end_time` 范围之间存在的事务日志备份。在此示例中，存储过程将复制在“2022-08-23 00:00:00”与“2022-08-30 00:00:00”之间生成的事务日志备份。
`exec msdb.dbo.rds_tlog_backup_copy_to_S3 @db_name = 'testdb1', @backup_file_start_time='2022-08-23 00:00:00';`	复制最近七天的事务日志备份，并从提供的 `backup_file_start_time` 开始。在此示例中，存储过程将复制从“2022-08-23 00:00:00”开始的事务日志备份，直到最新的事务日志备份。

提供的参数	预期结果
`exec msdb.dbo.rds_tlog_backup_copy_to_S3 @db_name = 'testdb1', @backup_file_end_time='2022-08-30 00:00:00';`	复制最近七天的事务日志备份，直至提供的 `backup_file_end_time` 。在此示例中，存储过程将复制从“2022-08-23 00:00:00”到“2022-08-30 00:00:00”的事务日志备份。
`exec msdb.dbo.rds_tlog_backup_copy_to_S3 @db_name='testdb1', @starting_lsn =1490000000040007, @ending_lsn = 1490000000050009;`	复制最近七天内可用且介于提供的 `starting_lsn` 和 `ending_lsn` 范围之间的事务日志备份。在此示例中，存储过程将复制最近七天且 LSN 范围介于 1490000000040007 到 1490000000050009 之间的事务日志备份。

提供的参数	预期结果
`exec msdb.dbo.rds_tlog_backup_copy_to_S3 @db_name='testdb1', @starting_lsn =1490000000040007;`	复制最近七天内可用的事务日志备份，从提供的 `starting_lsn` 开始。在此示例中，存储过程将复制从 LSN 1490000000040007 开始的事务日志备份，直至最新的事务日志备份。
`exec msdb.dbo.rds_tlog_backup_copy_to_S3 @db_name='testdb1', @ending_lsn =1490000000050009;`	复制最近七天内可用的事务日志备份，直到提供的 `ending_lsn` 。在本例中，存储过程将复制从最近七天开始直至 lsn 1490000000050009 的事务日志备份。

提供的参数	预期结果	
`exec msdb.dbo.rds_tlog_backup_copy_to_S3 @db_name='testdb1', @rds_backup_starting_seq_id= 2000, @rds_backup_ending_seq_id= 5000;`	复制最近七天内可用且在提供的范围 `rds_backup_starting_seq_id` 和 `rds_backup_ending_seq_id` 之间存在的事务日志备份。在本例中，存储过程将复制从最近七天开始且介于提供的 rds 备份序列 id 范围（从 seq_id 2000 到 seq_id5000）内的事务日志备份。	
`exec msdb.dbo.rds_tlog_backup_copy_to_S3 @db_name='testdb1', @rds_backup_starting_seq_id= 2000;`	复制最近七天内可用的事务日志备份，从提供的 `rds_backup_starting_seq_id` 开始。在此示例中，存储过程将复制从 seq_id 2000 开始的事务日志备份，直到最新的事务日志备份。	

提供的参数	预期结果
`exec msdb.dbo.rds_tlog_backup_copy_to_S3 @db_name='testdb1', @rds_backup_ending_seq_id= 5000;`	复制最近七天内可用的事务日志备份，直到提供的 `rds_backup_ending_seq_id` 。在此示例中，存储过程将复制从最近七天开始的事务日志备份，直到 seq_id 5000。
`exec msdb.dbo.rds_tlog_backup_copy_to_S3 @db_name='testdb1', @rds_backup_starting_seq_id= 2000; @rds_backup_ending_seq_id= 2000;`	使用提供的 `rds_backup_starting_seq_id` 复制单个事务日志备份（如果在过去七天内可用）。在此示例中，存储过程将复制 seq_id 为 2000 的单个事务日志备份，前提是该备份在过去七天内存在。

验证事务日志备份日志链

为访问事务日志备份而配置的数据库必须启用自动备份保留。自动备份保留将数据库实例上的数据库设置为 FULL 恢复模式。要支持数据库的时间点还原，请避免更改数据库恢复模式，因为这可能会导致日志链断开。我们建议将数据库设置为 FULL 恢复模式。

要在复制事务日志备份之前手动验证日志链，请调用 `rds_fn_list_tlog_backup_metadata` 函数并查看 `is_log_chain_broken` 列中的值。值为“1”表示当前日志备份和前一个日志备份之间的日志链已断开。

以下示例显示了 `rds_fn_list_tlog_backup_metadata` 存储过程的输出中存在中断的日志链。

rds_sequence_id	first_lsn	last_lsn	is_log_chain_broken
43	90023	90457	0
44	90457	90985	0
45	90987	92034	1

在正常日志链中，给定 rds_sequence_id 的 first_lsn 的日志序列号（LSN）值应与前一个 rds_sequence_id 中的 last_lsn 值相匹配。在图中，rds_sequence_id 为 45 的 first_lsn 值为 90987，这与前一个 rds_seque_id 为 44 的 last_lsn 值 90985 不匹配。

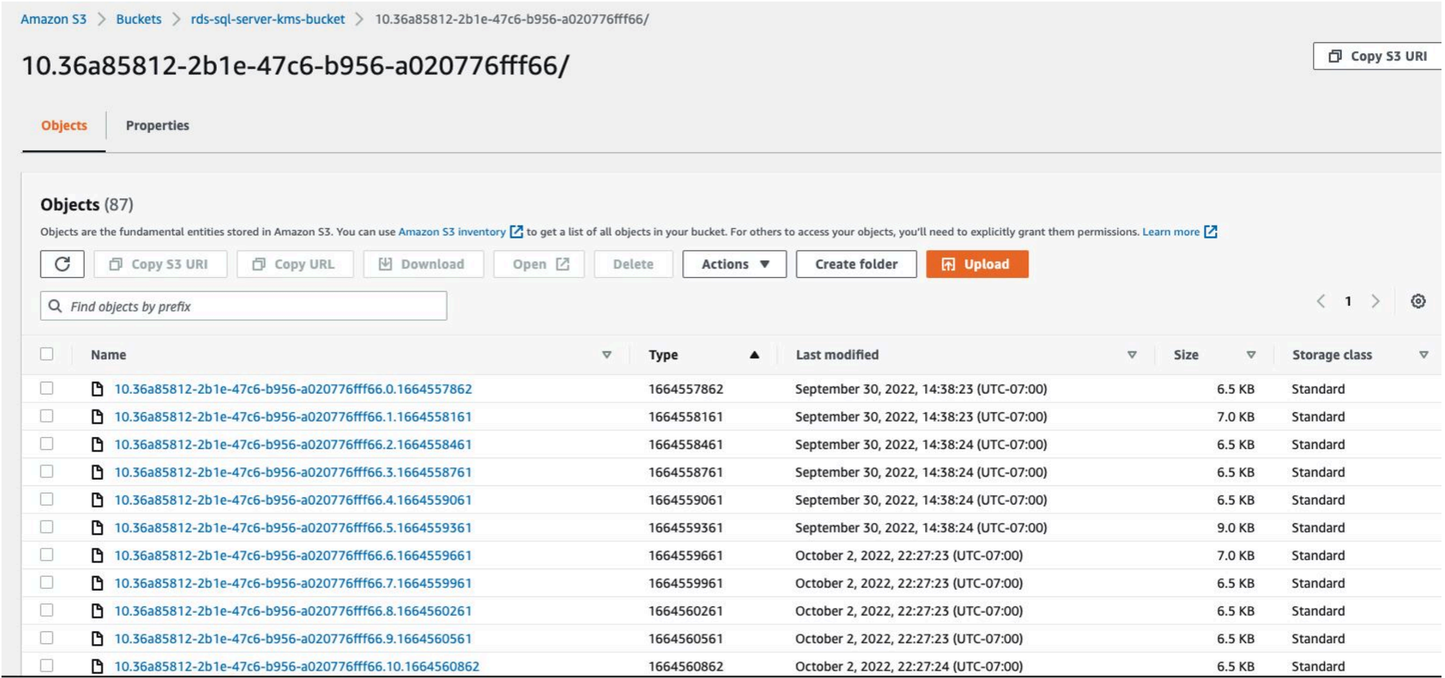
有关 SQL Server 事务日志架构和日志序列号的更多信息，请参阅 Microsoft SQL Server 文档中的[事务日志逻辑架构](#)。

Amazon S3 桶文件夹和文件结构

在 Amazon S3 桶中，事务日志备份具有以下标准结构和命名约定：

- 在每个数据库的 `target_s3_arn` 路径下创建一个新文件夹，其命名结构为 `{db_id}.{family_guid}`。
- 在该文件夹中，事务日志备份的文件名结构为 `{db_id}.{family_guid}.{rds_backup_seq_id}.{backup_file_epoch}`。
- 您可以使用 `rds_fn_list_tlog_backup_metadata` 函数查看 `family_guid,db_id,rds_backup_seq_id and backup_file_epoch` 的详细信息。

以下示例显示 Amazon S3 桶内一组事务日志备份的文件夹和文件结构。



跟踪任务的状态

要跟踪复制任务的状态，请调用 `rds_task_status` 存储过程。如果您未提供任何参数，则存储过程将返回所有任务的状态。

Example 用法：

```
exec msdb.dbo.rds_task_status
  @db_name='database_name',
  @task_id=ID_number;
```

以下参数为可选参数：

- `@db_name`– 要显示其任务状态的数据库的名称。
- `@task_id`– 要显示其任务状态的任务的 ID。

Example 列出特定任务 ID 的状态：

```
exec msdb.dbo.rds_task_status @task_id=5;
```

Example 列出特定数据库和任务的状态：

```
exec msdb.dbo.rds_task_status@db_name='my_database',@task_id=5;
```

Example 列出特定数据库的所有任务及其状态：

```
exec msdb.dbo.rds_task_status @db_name='my_database';
```

Example 列出当前数据库实例上的所有任务及其状态：

```
exec msdb.dbo.rds_task_status;
```

取消任务

要取消正在运行的任务，请调用 `rds_cancel_task` 存储过程。

Example 用法：

```
exec msdb.dbo.rds_cancel_task @task_id=ID_number;
```

以下参数是必需参数：

- `@task_id`– 要取消的任务的 ID。可以通过调用 `rds_task_status` 存储过程查看任务 ID。

有关查看和取消正在运行的任务的更多信息，请参阅使用本机备份和还原导入和导出 SQL Server 数据库。

访问事务日志备份故障排除

以下是您使用存储过程访问事务日志备份时可能遇到的问题。

存储过程	错误消息	问题	故障排除建议
rds_tlog_ copy_setup	此数据库实例上已禁用	未为数据库实例启用自动备份。	必须启用数据库实例备份保留，保留期至少为一天。有关启用自

存储过程	错误消息	问题	故障排除建议
	备份。启用保留期至少为“1”的数据库实例备份，然后重试。		动备份和配置备份保留期的更多信息，请参阅备份保留期。
rds_tlog_copy_setup	运行 rds_tlog_copy_setup 存储过程时出错。重新连接到 RDS 端点并重试。	出现内部错误。	重新连接到 RDS 端点并再次运行 `rds_tlog_copy_setup` 存储过程。
rds_tlog_copy_setup	不支持在事务内运行 rds_tlog_backup_copy_setup 存储过程。请验证会话没有未完成的事务，然后重试。	在事务中使用 `BEGIN` 和 `END` 尝试执行了此存储过程。	运行 `rds_tlog_copy_setup` 存储过程时避免使用 `BEGIN` 和 `END`。
rds_tlog_copy_setup	输入参数 `@target_s3_arn` 的 S3 桶名称应包含除空格外的至少一个字符。	为输入参数 `@target_s3_arn` 提供的值不正确。	确保输入参数 `@target_s3_arn` 指定了完整的 Amazon S3 桶 ARN。

存储过程	错误消息	问题	故障排除建议
rds_tlog_copy_setup	SQLSERVER_BACKUP_RESTORE 选项未启用或处于启用过程中。启用该选项或稍后重试。	SQLSERVER_BACKUP_RESTORE 选项未在数据库实例上启用，或者刚刚启用并等待内部激活。	按照“要求”部分指定的方式启用 SQLSERVER_BACKUP_RESTORE 选项。等待几分钟，然后再次运行 rds_tlog_copy_setup 存储过程。
rds_tlog_copy_setup	输入参数 @target_s3_arn 的目标 S3 arn 不能为空或 null。	为输入参数 @target_s3_arn 提供了 NULL 值，或者没有提供值。	确保输入参数 @target_s3_arn 指定了完整的 Amazon S3 桶 ARN。
rds_tlog_copy_setup	输入参数 @target_s3_arn 的目标 S3 arn 必须以 arn:aws 开头。	提供的输入参数 @target_s3_arn 前面没有 arn:aws。	确保输入参数 @target_s3_arn 指定了完整的 Amazon S3 桶 ARN。
rds_tlog_copy_setup	目标 S3 ARN 桶已设置为所提供的值。	rds_tlog_copy_setup 存储过程之前已运行并使用 Amazon S3 桶 ARN 进行配置。	要修改访问事务日志备份的 Amazon S3 桶值，请提供不同的 target S3 ARN。

存储过程	错误消息	问题	故障排除建议
rds_tlog_copy_setup	无法生成用于启用“访问事务日志备份”的凭证。确认通过 `rds_tlog_copy_setup` 提供了 S3 路径 ARN，稍后重试。	生成凭证以启用访问事务日志备份时出现未指定的错误。	检查您的设置配置，然后重试。
rds_tlog_copy_setup	存在待处理任务时，您无法运行 rds_tlog_copy_setup 存储过程。等待待处理任务完成，然后重试。	任何时候只能运行两个任务。有待处理任务等待完成。	查看待处理任务并等待其完成。有关监控任务状态的更多信息，请参阅跟踪任务的状态。
rds_tlog_backup_copy_to_S3	已为数据库发出 T-log 备份文件复制任务：%s，任务 ID：%d，请稍后重试。	给定数据库在任何时候只能运行一个复制任务。有待处理的复制任务等待完成。	查看待处理任务并等待其完成。有关监控任务状态的更多信息，请参阅跟踪任务的状态。

存储过程	错误消息	问题	故障排除建议
rds_tlog_backup_copy_to_S3	必须提供这三个参数集中的至少一个。SET-1:(@backup_file_start_time, @backup_file_end_time) | SET-2:(@starting_lsn, @ending_lsn) | SET-3:(@rds_backup_starting_seq_id, @rds_backup_ending_seq_id)	三个参数集均未提供，或者提供的参数集缺少必需的参数。	您可以指定时间、lsn 或序列 ID 参数。这三组参数中的一组是必需的。有关所需参数的更多信息，请参阅复制事务日志备份。
rds_tlog_backup_copy_to_S3	您的实例上已禁用备份。请启用备份，稍后重试。	未为数据库实例启用自动备份。	有关启用自动备份和配置备份保留期的更多信息，请参阅备份保留期。
rds_tlog_backup_copy_to_S3	找不到给定的数据库 %s。	为输入参数 `@db_name` 提供的值与数据库实例上的数据库名称不匹配。	使用正确的数据库名称。要按名称列出所有数据库，请运行 `SELECT * from sys.databases`

存储过程	错误消息	问题	故障排除建议
rds_tlog_backup_copy_to_S3	无法对于 SQL Server 系统数据库或 rdsadmin 数据库运行 rds_tlog_backup_copy_to_S3 存储过程。	为输入参数 `@db_name` 提供的值与 SQL Server 系统数据库名称或 RDSAdmin 数据库相匹配。	不允许使用以下数据库来访问事务日志备份：`master, model, msdb, tempdb, RDSAdmin.`
rds_tlog_backup_copy_to_S3	输入参数 @db_name 的数据库名称不能为空或为 Null。	为输入参数 `@db_name` 提供的值为空或 `NULL`。	使用正确的数据库名称。要按名称列出所有数据库，请运行 `SELECT * from sys.databases`
rds_tlog_backup_copy_to_S3	数据库实例备份保留期必须设置为至少 1 才能运行 rds_tlog_backup_copy_setup 存储过程。	未为数据库实例启用自动备份。	有关启用自动备份和配置备份保留期的更多信息，请参阅备份保留期。
rds_tlog_backup_copy_to_S3	运行存储过程 rds_tlog_backup_copy_to_S3 时出错。重新连接到 RDS 端点并重试。	出现内部错误。	重新连接到 RDS 端点并再次运行 `rds_tlog_backup_copy_to_S3` 存储过程。

存储过程	错误消息	问题	故障排除建议
rds_tlog_backup_copy_to_S3	只能提供这三个参数集中的一个。SET-1:(@backup_file_start_time, @backup_file_end_time) | SET-2:(@starting_lsn, @ending_lsn) | SET-3:(@rds_backup_starting_seq_id, @rds_backup_ending_seq_id)	提供了多个参数集。	您可以指定时间、lsn 或序列 ID 参数。这三组参数中的一组是必需的。有关所需参数的更多信息，请参阅复制事务日志备份。
rds_tlog_backup_copy_to_S3	不支持在事务内运行 rds_tlog_backup_copy_to_S3 存储过程。请验证会话没有未完成的事务，然后重试。	在事务中使用 `BEGIN` 和 `END` 尝试执行了此存储过程。	运行 `rds_tlog_backup_copy_to_S3` 存储过程时避免使用 `BEGIN` 和 `END`。

存储过程	错误消息	问题	故障排除建议
rds_tlog_backup_copy_to_S3	提供的参数超出了事务备份日志保留期。要列出可用的事务日志备份文件，请运行 rds_fn_list_tlog_backup_metadata 函数。	对于提供的输入参数，没有适合副本保留期限的可用事务日志备份。	使用一组有效的参数重试。有关所需参数的更多信息，请参阅复制事务日志备份。
rds_tlog_backup_copy_to_S3	处理请求时出现权限错误。确保桶与数据库实例位于相同的账户和区域中，并根据公有文档中的模板确认 S3 桶策略权限。	检测到所提供的 S3 桶或其策略权限存在问题。	确认访问事务日志备份的设置正确无误。有关 S3 桶的设置要求的更多信息，请参阅要求。
rds_tlog_backup_copy_to_S3	不允许在 RDS 只读副本实例上运行 `rds_tlog_backup_copy_to_S3` 存储过程。	已尝试在 RDS 只读副本实例上执行存储过程。	连接到 RDS 主数据库实例以运行 `rds_tlog_backup_copy_to_S3` 存储过程。

存储过程	错误消息	问题	故障排除建议
rds_tlog_backup_copy_to_S3	输入参数 @starting_lsn 的 LSN 必须小于 @ending_lsn 。	为输入参数 @starting_lsn 提供的值大于为输入参数 @ending_lsn 提供的值。	确保为输入参数 @starting_lsn 提供的值小于为输入参数 @ending_lsn 提供的值。
rds_tlog_backup_copy_to_S3	rds_tlog_backup_copy_to_S3 存储过程只能由源数据库中 db_owner 角色的成员执行。	尚未为尝试在提供的 db_name 上运行 rds_tlog_backup_copy_to_S3 存储过程的账户授予 db_owner 角色	确保运行存储过程的账户对于所提供的 db_name 获得 db_owner 角色的权限。
rds_tlog_backup_copy_to_S3	输入参数 @rds_backup_starting_seq_id 的序列 ID 必须小于或等于 @rds_backup_ending_seq_id 。	为输入参数 @rds_backup_starting_seq_id 提供的值大于为输入参数 @rds_backup_ending_seq_id 提供的值。	确保为输入参数 @rds_backup_starting_seq_id 提供的值小于为输入参数 @rds_backup_ending_seq_id 提供的值。

存储过程	错误消息	问题	故障排除建议
rds_tlog_backup_copy_to_S3	SQLSERVER_BACKUP_RESTORE 选项未启用或处于启用过程中。启用该选项或稍后重试。	SQLSERVER_BACKUP_RESTORE 选项未在数据库实例上启用，或者刚刚启用并等待内部激活。	按照“要求”部分指定的方式启用 SQLSERVER_BACKUP_RESTORE 选项。等待几分钟，然后再次运行 rds_tlog_backup_copy_to_S3 存储过程。
rds_tlog_backup_copy_to_S3	输入参数 @backup_file_start_time 的开始时间必须小于 @backup_file_end_time 。	为输入参数 @backup_file_start_time 提供的值大于为输入参数 @backup_file_end_time 提供的值。	确保为输入参数 @backup_file_start_time 提供的值小于为输入参数 @backup_file_end_time 提供的值。
rds_tlog_backup_copy_to_S3	由于缺乏访问权限，我们无法处理该请求。请检查您对该功能的设置和权限。	Amazon S3 桶权限可能存在问题，或者提供的 Amazon S3 桶位于其他账户或区域中。	确保 Amazon S3 桶策略权限允许 RDS 访问。确保 Amazon S3 桶与数据库实例位于同一账户和区域中。
rds_tlog_backup_copy_to_S3	对于未经过存储加密的实例，您不能将 KMS 密钥 ARN 作为输入参数提供给存储过程。	如果未在数据库实例上启用存储加密，则不应提供输入参数 @kms_key_arn 。	请勿为 @kms_key_arn 提供输入参数。

存储过程	错误消息	问题	故障排除建议
rds_tlog_backup_copy_to_S3	您必须向存储加密实例的存储过程提供 KMS 密钥 ARN 作为输入参数。	在数据库实例上启用存储加密时，必须提供输入参数 `@kms_key_arn` 。	为 `@kms_key_arn` 提供一个输入参数，其值与 Amazon S3 桶的 ARN 相匹配，用于事务日志备份。
rds_tlog_backup_copy_to_S3	在运行 `rds_tlog_backup_copy_to_S3` 存储过程之前，必须先运行 `rds_tlog_copy_setup` 存储过程并设置 `@target_s3_arn` 。	在尝试运行 `rds_tlog_backup_copy_to_S3` 存储过程之前，尚未完成对事务日志备份设置过程的访问。	在运行 `rds_tlog_copy_setup` 存储过程之前运行 `rds_tlog_backup_copy_to_S3` 存储过程。有关运行设置过程以访问事务日志备份的更多信息，请参阅设置访问事务日志备份。

适用于 Microsoft SQL Server 数据库引擎的选项

在本节中，您可以找到选项的说明，这些选项可供运行 Microsoft SQL Server 数据库引擎的 Amazon RDS 实例使用。要启用这些选项，可将它们添加到选项组中，然后将选项组关联到数据库实例。有关更多信息，请参阅“使用选项组”。

如果要查找未通过 RDS 选项组添加的可选功能（例如 SSL、Microsoft Windows 身份验证和 Amazon S3 集成），请参阅Amazon RDS 上的 Microsoft SQL Server 的其他功能。

Amazon RDS 支持 Microsoft SQL Server 数据库实例的以下选项。

选项	选项 ID	引擎版本
带有 Oracle OLEDB 的链接服务器	`OLEDB_ORACLE`	SQL Server 企业版 SQL Server 标准版
本机备份和还原	`SQLSERVER_BACKUP_RESTORE`	SQL Server 企业版 SQL Server 标准版 SQL Server Web Edition SQL Server Express Edition
透明数据加密	`TRANSPARENT_DATA_ENCRYPTION` （RDS 控制台） `TDE`（Amazon CLI 和 RDS API）	SQL Server 2016–2022 企业版 SQL Server 2022 标准版
SQL Server Audit	`SQLSERVER_AUDIT`	在 RDS 中，从 SQL Server 2016 开始，SQL Server 的所有版本都支持服务器

选项	选项 ID	引擎版本
		级审计，而企业版还支持数据库级审计。 从 SQL Server SQL Server 2016 (13.x) SP1 开始，所有版本均支持服务器级别和数据库级别审核。 有关更多信息，请参阅 SQL Server 文档中的 [SQL Server Audit（数据库引擎）](#)。
[SQL Server Analysis Services](#)	`SSAS`	SQL Server 企业版 SQL Server 标准版
[SQL Server Integration Services](#)	`SSIS`	SQL Server 企业版 SQL Server 标准版
[SQL Server Reporting Services](#)	`SSRS`	SQL Server 企业版 SQL Server 标准版
[Microsoft 分布式事务处理协调器](#)	`MSDTC`	在 RDS 中，从 SQL Server 2016 开始，SQL Server 的所有版本都支持分布式事务。

列出 SQL Server 版本的可用选项

您可以使用 `describe-option-group-options` Amazon CLI 命令列出 SQL Server 版本的可用选项以及这些选项的设置。

以下示例显示了 SQL Server 2019 企业版的选项和选项设置。`--engine-name` 选项是必需的。

```
aws rds describe-option-group-options --engine-name sqlserver-ee --major-engine-version
 15.00
```

输出与以下内容类似：


```
{
    "OptionGroupOptions": [
        {
            "Name": "MSDTC",
            "Description": "Microsoft Distributed Transaction Coordinator",
            "EngineName": "sqlserver-ee",
            "MajorEngineVersion": "15.00",
            "MinimumRequiredMinorEngineVersion": "4043.16.v1",
            "PortRequired": true,
            "DefaultPort": 5000,
            "OptionsDependedOn": [],
            "OptionsConflictsWith": [],
            "Persistent": false,
            "Permanent": false,
            "RequiresAutoMinorEngineVersionUpgrade": false,
            "VpcOnly": false,
            "OptionGroupOptionSettings": [
                {
                    "SettingName": "ENABLE_SNA_LU",
                    "SettingDescription": "Enable support for SNA LU protocol",
                    "DefaultValue": "true",
                    "ApplyType": "DYNAMIC",
                    "AllowedValues": "true,false",
                    "IsModifiable": true,
                    "IsRequired": false,
                    "MinimumEngineVersionPerAllowedValue": []
                },
        ...

        {
            "Name": "TDE",
            "Description": "SQL Server - Transparent Data Encryption",
            "EngineName": "sqlserver-ee",
            "MajorEngineVersion": "15.00",
            "MinimumRequiredMinorEngineVersion": "4043.16.v1",
            "PortRequired": false,
```

```
            "OptionsDependedOn": [],
            "OptionsConflictsWith": [],
            "Persistent": true,
            "Permanent": false,
            "RequiresAutoMinorEngineVersionUpgrade": false,
            "VpcOnly": false,
            "OptionGroupOptionSettings": []
        }
    ]
}
```

在 Amazon RDS for SQL Server 中支持带有 Oracle OLEDB 的链接服务器

RDS for SQL Server 上带有 Oracle Provider for OLEDB 的链接服务器可让您访问 Oracle 数据库上的外部数据来源。您可以从远程 Oracle 数据来源读取数据，并对您的 RDS for SQL Server 数据库实例之外的远程 Oracle 数据库服务器运行命令。使用带有 Oracle OLEDB 的链接服务器，您可以：

- 直接访问 SQL Server 以外的数据来源
- 使用相同的查询对不同的 Oracle 数据来源进行查询，无需移动数据
- 对整个企业生态系统中的数据来源发出分布式查询、更新、命令和事务
- 在 Microsoft 商业智能套件（SSIS、SSRS、SSAS）中集成与 Oracle 数据库的连接
- 从 Oracle 数据库迁移到 RDS for SQL Server

您可以在现有或新的 RDS for SQL Server 数据库实例上激活一个或多个适用于 Oracle 的链接服务器。然后，您可以将外部 Oracle 数据来源与您的数据库实例集成。

目录

支持的版本和区域

在以下版本上，对于 SQL Server 标准版和企业版，RDS for SQL Server 在所有区域中支持带有 Oracle OLEDB 的链接服务器：

- SQL Server 2022，所有版本
- SQL Server 2019，所有版本
- SQL Server 2017，所有版本

以下 Oracle Database 版本支持带有 Oracle OLEDB 的链接服务器：

- Oracle Database 21c，所有版本
- Oracle Database 19c，所有版本
- Oracle Database 18c，所有版本

限制和建议

请记住以下适用于带有 Oracle OLEDB 的链接服务器的限制和建议：

- 通过在每个 RDS for SQL Server 数据库实例的安全组中添加适用的 TCP 端口来允许网络流量。例如，如果您在 EC2 Oracle 数据库实例和 RDS for SQL Server 数据库实例之间配置链接服务器，则必须允许来自 EC2 Oracle 数据库实例的 IP 地址的流量。您还必须允许 SQL Server 用于侦听数据库通信的端口上的流量。有关安全组的更多信息，请参阅使用安全组控制访问权限。
- 在打开、关闭或修改选项组中的 `OLEDB_ORACLE` 选项后，重启 RDS for SQL Server 数据库实例。选项组状态对于这些事件显示 `pending_reboot` 且是必需的。对于启用了 AlwaysOn 或镜像选项的 RDS for SQL Server 多可用区实例，在新实例创建或还原后重启实例时，预计会进行失效转移。
- Oracle 数据来源的用户名和密码仅支持简单身份验证。
- 不支持开放式数据库连接（ODBC）驱动程序。仅支持最新版本的 OLEDB 驱动程序。
- 支持分布式事务（XA）。要激活分布式事务，请为您的数据库实例开启选项组中的 `MSDTC` 选项，并确保 XA 事务处于开启状态。有关更多信息，请参阅 RDS for SQL Server 中的 Microsoft 分布式事务处理协调器支持。
- 不支持创建用作连接字符串的快捷方式的数据来源名称（DSN）。
- 不支持 OLEDB 驱动程序跟踪。您可以使用 SQL Server 扩展事件来跟踪 OLEDB 事件。有关更多信息，请参阅在 RDS for SQL Server 中设置扩展事件。
- 不支持使用 SQL Server Management Studio（SSMS）访问 Oracle 链接服务器的目录文件夹。

激活适用于 Oracle 的链接服务器

通过在 RDS for SQL Server 数据库实例中添加 `OLEDB_ORACLE` 选项来激活适用于 Oracle 的链接服务器。使用以下过程：

1. 创建新的选项组或选择现有的选项组。
2. 将 `OLEDB_ORACLE` 选项添加到该选项组。

3. 选择要使用的 OLEDB 驱动程序的版本。
4. 将选项组与数据库实例相关联。
5. 重启数据库实例。

为 OLEDB_ORACLE 创建选项组

要使用适用于 Oracle 的链接服务器，请创建或修改与您计划使用的 SQL Server 版本和数据库实例版本对应的选项组。要完成此过程，请使用 Amazon Web Services Management Console或 Amazon CLI。

控制台

以下过程创建适用于 SQL Server 标准版 2019 的选项组。

创建选项组

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择选项组。
3. 选择创建群组。
4. 在创建选项组窗口中，执行以下操作：
   a. 对于名称，输入选项组的名称，该名称在 Amazon 账户内具有唯一性，例如 **`oracle-oledb-se-2019`**。此名称只能包含字母、数字和连字符。
   b. 对于描述，输入选项组的简要描述，例如 **`OLEDB_ORACLE option group for SQL Server SE 2019`**。此说明用于显示说明。
   c. 对于引擎，选择 sqlserver-se。
   d. 对于 Major engine version（主引擎版本），选择 15.00。
5. 选择创建。

CLI

以下过程创建适用于 SQL Server 标准版 2019 的选项组。

创建选项组

- 运行以下命令之一。

Example

对于 Linux、macOS 或 Unix：

```
aws rds create-option-group \
    --option-group-name oracle-oledb-se-2019 \
    --engine-name sqlserver-se \
    --major-engine-version 15.00 \
    --option-group-description "OLEDB_ORACLE option group for SQL Server SE 2019"
```

对于 Windows：

```
aws rds create-option-group ^
    --option-group-name oracle-oledb-se-2019 ^
    --engine-name sqlserver-se ^
    --major-engine-version 15.00 ^
    --option-group-description "OLEDB_ORACLE option group for SQL Server SE 2019"
```

将 **OLEDB_ORACLE** 选项添加到选项组

接下来，使用 Amazon Web Services Management Console或 Amazon CLI 将 `OLEDB_ORACLE` 选项添加到您的选项组。

控制台

添加 OLEDB_ORACLE 选项

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择选项组。
3. 选择您刚刚创建的选项组（本例中为 oracle-oledb-se-2019）。
4. 选择添加选项。
5. 在 Option details（选项详细信息）下，为 Option name（选项名称）选择 OLEDB_ORACLE。
6. 在 计划 (Scheduling) 下，选择是立即添加选项还是在下一个维护时段添加选项。
7. 选择添加选项。

CLI

添加 OLEDB_ORACLE 选项

- 将 OLEDB_ORACLE 选项添加到该选项组。

  Example

  对于 Linux、macOS 或 Unix：

```
aws rds add-option-to-option-group \
    --option-group-name oracle-oledb-se-2019 \
    --options OptionName=OLEDB_ORACLE \
    --apply-immediately
```

  对于 Windows：

```
aws rds add-option-to-option-group ^
    --option-group-name oracle-oledb-se-2019 ^
    --options OptionName=OLEDB_ORACLE ^
    --apply-immediately
```

将选项组与数据库实例关联

要将 OLEDB_ORACLE 选项组和参数组与数据库实例关联，请使用 Amazon Web Services Management Console或 Amazon CLI

控制台

要完成激活适用于 Oracle 的链接服务器，请将您的 OLEDB_ORACLE 选项组与新的或现有的数据库实例关联：

- 对于新数据库实例，请在启动实例时将其关联。有关更多信息，请参阅“创建 Amazon RDS 数据库实例”。
- 对于现有数据库实例，请通过修改实例将其关联。有关更多信息，请参阅 修改 Amazon RDS 数据库实例。

CLI

您可以将 OLEDB_ORACLE 选项组和参数组与新的或现有的数据库实例关联。

使用 **OLEDB_ORACLE** 选项组和参数组创建实例

- 指定创建选项组时使用的相同数据库引擎类型和主要版本。

  Example

  对于 Linux、macOS 或 Unix：

```
aws rds create-db-instance \
    --db-instance-identifier mytestsqlserveroracleoledbinstance \
    --db-instance-class db.m5.2xlarge \
    --engine sqlserver-se \
    --engine-version 15.0.4236.7.v1 \
    --allocated-storage 100 \
    --manage-master-user-password \
    --master-username admin \
    --storage-type gp2 \
    --license-model li \
    --domain-iam-role-name my-directory-iam-role \
    --domain my-domain-id \
    --option-group-name oracle-oledb-se-2019 \
    --db-parameter-group-name my-parameter-group-name
```

  对于 Windows：

```
aws rds create-db-instance ^
    --db-instance-identifier mytestsqlserveroracleoledbinstance ^
    --db-instance-class db.m5.2xlarge ^
    --engine sqlserver-se ^
    --engine-version 15.0.4236.7.v1 ^
    --allocated-storage 100 ^
    --manage-master-user-password ^
    --master-username admin ^
    --storage-type gp2 ^
    --license-model li ^
    --domain-iam-role-name my-directory-iam-role ^
    --domain my-domain-id ^
    --option-group-name oracle-oledb-se-2019 ^
    --db-parameter-group-name my-parameter-group-name
```

修改实例并关联 **OLEDB_ORACLE** 选项组

- 运行以下命令之一。

  Example

  对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier mytestsqlserveroracleoledbinstance \
    --option-group-name oracle-oledb-se-2019 \
    --db-parameter-group-name my-parameter-group-name \
    --apply-immediately
```

  对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier mytestsqlserveroracleoledbinstance ^
    --option-group-name oracle-oledb-se-2019 ^
    --db-parameter-group-name my-parameter-group-name ^
    --apply-immediately
```

修改 OLEDB 提供程序属性

您可以查看和更改 ODB 提供程序的属性。只有 `master` 用户才能执行此任务。在数据库实例上创建的所有适用于 Oracle 的链接服务器都使用该 OLEDB 提供程序的相同属性。调用 `sp_MSset_oledb_prop` 存储过程以更改 OLEDB 提供程序的属性。

更改 OLEDB 提供程序属性

```

USE [master]
GO
EXEC sp_MSset_oledb_prop N'OraOLEDB.Oracle', N'AllowInProcess', 1
EXEC sp_MSset_oledb_prop N'OraOLEDB.Oracle', N'DynamicParameters', 0
GO
```

可以修改以下属性：

属性名称	建议的值（1 = 开启，0 = 关闭）	描述
`Dynamic parameter`	1	在参数化查询中允许 SQL 占位符（以“?”表示）。
`Nested queries`	1	允许在 `FROM` 子句中嵌套 `SELECT` 语句，例如子查询。
`Level zero only`	0	仅针对此提供程序调用基级 OLEDB 接口。
`Allow inprocess`	1	如果开启，Microsoft SQL Server 允许将提供程序实例化为进程内服务器。将此属性设置为 1 可使用 Oracle 链接服务器。
`Non transacted updates`	0	如果不为零，则 SQL Server 允许更新。
`Index as access path`	False	如果不为零，则 SQL Server 会尝试使用提供程序的索引来获取数据。
`Disallow adhoc access`	False	如果已设置，则 SQL Server 不允许对 OLEDB 提供程序运行传递查询。虽然可以选中此选项，但有时运行传递查询是合适的。
`Supports LIKE operator`	1	表示提供程序支持使用 LIKE 关键字进行查询。

修改 OLEDB 驱动程序属性

在创建适用于 Oracle 的链接服务器时，可以查看和更改 OLEDB 驱动程序的属性。只有 `master` 用户才能执行此任务。驱动程序属性定义了在使用远程 Oracle 数据来源时 OLEDB 驱动程序如何处理数据。驱动程序属性特定于在数据库实例上创建的每个 Oracle 链接服务器。调用 `master.dbo.sp_addlinkedserver` 存储过程以更改 OLEDB 驱动程序的属性。

示例：创建链接服务器并更改 OLEDB 驱动程序 `FetchSize` 属性

```
EXEC master.dbo.sp_addlinkedserver
```

```
@server = N'Oracle_link2',
@srvproduct=N'Oracle',
@provider=N'OraOLEDB.Oracle',
@datasrc=N'my-oracle-test.cnetsipka.us-west-2.rds.amazonaws.com:1521/ORCL',
@provstr='FetchSize=200'
GO
```

```
EXEC master.dbo.sp_addlinkedsrvlogin
@rmtsrvname=N'Oracle_link2',
@useself=N'False',
@locallogin=NULL,
@rmtuser=N'master',
@rmtpassword='Test#1234'
GO
```

> **Note**
>
> 作为安全最佳实践，请指定除此处所示提示以外的密码。

停用适用于 Oracle 的链接服务器

要停用适用于 Oracle 的链接服务器，请从其选项组中删除 `OLEDB_ORACLE` 选项。

> **Important**
>
> 删除此选项不会删除数据库实例上现有的链接服务器配置。您必须手动删除它们，才能将其从数据库实例中删除。
>
> 删除后，您可以重新激活 `OLEDB_ORACLE` 选项，以重用先前在数据库实例上配置的链接服务器配置。

控制台

以下过程删除 `OLEDB_ORACLE` 选项。

从其选项组中删除 OLEDB_ORACLE 选项

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。

2. 在导航窗格中，选择选项组。
3. 选择具有 `OLEDB_ORACLE` 选项的选项组（在前面的示例中为 `oracle-oledb-se-2019`）。
4. 选择 Delete option (删除选项)。
5. 在 Deletion options（删除选项）下，为 Options to delete（待删除的选项）选择 OLEDB_ORACLE。
6. 在 Apply immediately（立即应用）下，选择 Yes（是）可立即删除选项，选择 No（否）可在下一个维护时段删除它。
7. 选择删除。

CLI

以下过程删除 `OLEDB_ORACLE` 选项。

从选项组中删除 OLEDB_ORACLE 选项

- 运行以下命令之一。

  Example

  对于 Linux、macOS 或 Unix：

```
aws rds remove-option-from-option-group \
    --option-group-name oracle-oledb-se-2019 \
    --options OLEDB_ORACLE \
    --apply-immediately
```

  对于 Windows：

```
aws rds remove-option-from-option-group ^
    --option-group-name oracle-oledb-se-2019 ^
    --options OLEDB_ORACLE ^
    --apply-immediately
```

RDS for SQL Server 中适用于 Teradata ODBC 的链接服务器

在 RDS for SQL Server 上通过 Teradata ODBC 驱动程序支持链接服务器，可让您访问 Teradata 数据库上的外部数据来源。您可以从 RDS for SQL Server 实例外部的远程 Teradata 数据库服务器读取数据并运行命令。将链接服务器与 Teradata ODBC 结合使用可启用以下功能：

- 直接访问 SQL Server 以外的数据来源。
- 使用相同的查询对不同的 Teradata 数据来源进行查询，而无需移动数据。
- 对整个企业生态系统中的数据来源发出分布式查询、更新、命令和事务。
- 从 Microsoft 商业智能套件（SSIS、SSRS、SSAS）中集成与 Teradata 数据库的连接。
- 从 Teradata 数据库迁移到 RDS for SQL Server。

您可以选择在现有或新的 RDS for SQL Server 数据库实例上激活一个或多个适用于 Teradata 的链接服务器。然后，您可以将外部 Teradata 数据来源与您的数据库实例集成。

主题

支持的版本和区域

在以下版本上，对于 SQL Server 标准版和企业版，RDS for SQL Server 在所有 Amazon Web Services 区域中支持适用于 Teradata ODBC 的链接服务器：

- SQL Server 2022，所有版本
- SQL Server 2019，所有版本
- SQL Server 2017，所有版本

以下 Teradata 数据库版本支持与 RDS for SQL Server 进行链接

- Teradata 17.20，所有版本

限制和建议

以下限制适用于面向 Teradata ODBC 的链接服务器：

- RDS for SQL Server 仅支持针对 Teradata 源使用用户名和密码进行简单身份验证。
- RDS for SQL Server 仅支持 Teradata ODBC 驱动程序版本 17.20.0.33。
- RDS for SQL Server 不支持创建用作连接字符串的快捷方式的数据来源名称（DSN）。
- RDS for SQL Server 不支持 ODBC 驱动程序跟踪。使用 SQL Server 扩展事件来跟踪 ODBC 事件。有关更多信息，请参阅在 RDS for SQL Server 中设置扩展事件。
- 使用 SQL Server Management Studio（SSMS）时，RDS for SQL Server 不支持访问 Teradata 链接服务器的目录文件夹。

将链接服务器与 Teradata ODBC 结合使用时，请考虑以下建议：

- 通过在每个 RDS for SQL Server 数据库实例的安全组中添加适用的 TCP 端口来允许网络流量。如果您在 EC2 Teradata 数据库实例和 RDS for SQL Server 数据库实例之间配置链接服务器，则必须支持来自 EC2 Teradata 数据库实例的 IP 地址的流量。您还必须支持 RDS for SQL Server 数据库实例用于侦听数据库通信的端口上的流量。有关安全组的更多信息，请参阅使用安全组控制访问权限。
- 支持分布式事务（XA）。要激活分布式事务，请为您的数据库实例开启选项组中的 `MSDTC` 选项，并确保 XA 事务处于开启状态。有关更多信息，请参阅 RDS for SQL Server 中的 Microsoft 分布式事务处理协调器支持。
- 只要在 Teradata 服务器上进行了配置，链接的 Teradata ODBC 就支持 SSL/TLS。有关更多信息，请参阅 Enable TLS Connectivity on Teradata Vantage。

多可用区部署的注意事项

RDS for SQL Server 目前不在多可用区部署中将链接服务器复制到镜像的数据库服务器（或 Always-On 可用性组辅助服务器）。如果在更改配置以添加镜像或 Always-On 之前添加了链接服务器，则会为现有的链接服务器复制这些链接服务器。

或者，您可以在主实例上创建链接服务器，失效转移到高可用性服务器实例，然后再次创建链接服务器，使它们位于这两个 RDS for SQL Server 实例上。

激活适用于 Teradata 的链接服务器

通过在 RDS for SQL Server 数据库实例中添加 `ODBC_TERADATA` 选项，激活适用于 Teradata 的链接服务器。使用以下过程：

主题

- 为 ODBC_TERADATA 创建选项组
- 将 ODBC_TERADATA 选项添加到选项组
- 将 ODBC_TERADATA 选项与数据库实例关联

为 **ODBC_TERADATA** 创建选项组

要使用适用于 Teradata 的链接服务器，请创建或修改与您计划使用的 SQL Server 版本和数据库实例版本对应的选项组。要完成此过程，请使用 Amazon Web Services Management Console或 Amazon CLI。

控制台

使用以下过程创建适用于 SQL Server 标准版 2019 的选项组。

创建选项组

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择选项组。
3. 选择创建群组。
4. 在创建选项组窗口中，执行以下操作：
   a. 对于 Name（名称），输入选项组的名称，该名称在 Amazon Web Services 账户内具有唯一性，例如 `teradata-odbc-se-2019`。此名称只能包含字母、数字和连字符。
   b. 对于描述，输入选项组的简要描述。
   c. 对于引擎，选择 sqlserver-se。
   d. 对于 Major engine version（主引擎版本），选择 15.00。
5. 选择创建。

Amazon CLI

以下过程创建适用于 SQL Server 标准版 2019 的选项组。

Example

对于 Linux、macOS 或 Unix：

```
aws rds create-option-group \
    --option-group-name teradata-odbc-se-2019 \
    --engine-name sqlserver-se \
    --major-engine-version 15.00 \
    --option-group-description "ODBC_TERADATA option group for SQL Server SE 2019"
```

Example

对于 Windows：

```
aws rds create-option-group ^
    --option-group-name teradata-odbc-se-2019 ^
    --engine-name sqlserver-se ^
    --major-engine-version 15.00 ^
    --option-group-description "ODBC_TERADATA option group for SQL Server SE 2019"
```

将 **ODBC_TERADATA** 选项添加到选项组

接下来，使用 Amazon Web Services Management Console或 Amazon CLI 将 `ODBC_Teradata` 选项添加到您的选项组。

控制台

使用以下过程创建适用于 SQL Server 标准版 2019 的选项组。

添加 **ODBC_TERADATA** 选项。

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择选项组。
3. 选择您的新选项组。
4. 选择添加选项。
5. 在选项详细信息下：
   a. 对于选项名称，选择 ODBC_TERADATA。
   b. 对于选项版本，选择 `17.20.33.00`。
6. 在“计划”下，选择是立即添加选项，还是在下一个维护时段添加选项。
7. 选择添加选项。

Amazon CLI

以下过程将 ODBC_TERADATA 选项添加到您的选项组。

Example

对于 Linux、macOS 或 Unix：

```
aws rds add-option-to-option-group \
    --option-group-name teradata-odbc-se-2019 \
    --options "OptionName=ODBC_TERADATA,OptionVersion=17.20.33.00" \
    --apply-immediately
```

Example

对于 Windows：

```
aws rds add-option-to-option-group ^
    --option-group-name teradata-odbc-se-2019 ^
    --options "OptionName=ODBC_TERADATA,OptionVersion=17.20.33.00" ^
    --apply-immediately
```

将 **ODBC_TERADATA** 选项与数据库实例关联

要将 ODBC_TERADATA 选项组与数据库实例关联，请使用 Amazon Web Services Management Console或 Amazon CLI。

控制台

要完成激活适用于 Teradata 的链接服务器，请将选项组与新的或现有的数据库实例关联：

- 对于新的数据库实例，请在启动实例时将其关联。有关更多信息，请参阅“创建 Amazon RDS 数据库实例”。
- 对于现有数据库实例，请通过修改实例将其关联。有关更多信息，请参阅 修改 Amazon RDS 数据库实例。

Amazon CLI

指定创建选项组时使用的相同数据库引擎类型和主要版本。

对于 Linux、macOS 或 Unix：

```
aws rds create-db-instance \
    --db-instance-identifier mytestsqlserverteradataodbcinstance \
    --db-instance-class db.m5.2xlarge \
    --engine sqlserver-se \
    --engine-version 15.00 \
    --license-model license-included \
    --allocated-storage 100 \
    --master-username admin \
    --master-user-password password \
    --storage-type gp2 \
    --option-group-name teradata-odbc-se-2019
```

对于 Windows：

```
aws rds create-db-instance ^
    --db-instance-identifier mytestsqlserverteradataodbcinstance ^
    --db-instance-class db.m5.2xlarge ^
    --engine sqlserver-se ^
    --engine-version 15.00 ^
    --license-model license-included ^
    --allocated-storage 100 ^
    --master-username admin ^
    --master-user-password password ^
    --storage-type gp2 ^
    --option-group-name teradata-odbc-se-2019
```

修改实例并关联新的选项组：

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier mytestsqlserverteradataodbcinstance \
    --option-group-name teradata-odbc-se-2019 \
    --apply-immediately
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier mytestsqlserverteradataodbcinstance ^
    --option-group-name teradata-odbc-se-2019 ^
```

```
    --apply-immediately
```

创建适用于 Teradata 的链接服务器

要创建适用于 Teradata 的链接服务器，请运行以下命令：

```
EXECUTE master.dbo.sp_addlinkedserver
    @server = N'LinkedServer_NAME',
    @srvproduct=N'',
    @provider=N'MSDASQL',
    @provstr=N'"PROVIDER=MSDASQL;DRIVER={Teradata Database ODBC Driver 17.20};
                DBCName=Server;UID=user_name;PWD=user_password;
                UseDataEncryption=YES/NO;SSLMODE=PREFER/ALLOW/DISABLE>;"',
    @catalog='database'
```

```
EXECUTE master.dbo.sp_addlinkedsrvlogin
    @rmtsrvname = N'LinkedServer_NAME',
    @locallogin = NULL ,
    @useself = N'False',
    @rmtuser = N'user_name',
    @rmtpassword = N'user_password'
```

上面的命令示例如下所示：

```
EXECUTE master.dbo.sp_addlinkedserver
    @server = N'LinkedServerToTeradata',
    @srvproduct=N'',
    @provider=N'MSDASQL',
    @provstr=N'"PROVIDER=MSDASQL;DRIVER={Teradata Database ODBC Driver 17.20};
                DBCName=my-teradata-test.cnetsipka.us-west-2.rds.amazonaws.com;
                UID=master;
                PWD=Test#1234;
                UseDataEncryption=YES;
                SSLMODE=PREFER;"',
    @catalog='MyTestTeradataDB'

EXECUTE master.dbo.sp_addlinkedsrvlogin
    @rmtsrvname = N'LinkedServerToTeradata',
    @locallogin = NULL ,
    @useself = N'False',
    @rmtuser = N'master',
```

```
    @rmtpassword = N'Test#1234'
```

Note

作为安全最佳实践，请指定除此处所示提示以外的密码。

停用链接到 Teradata 的服务器

要停用到 Teradata 的链接服务器，请从其选项组中移除 ODBC_TERADATA 选项。

Important

删除此选项不会删除数据库实例上的链接服务器配置。您必须手动删除它们，才能将其从数据库实例中删除。

移除后，您可以重新激活 ODBC_TERADATA，以重用先前在数据库实例上配置的链接服务器配置。

控制台

从选项组中移除 ODBC_TERADATA 选项

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择选项组。
3. 选择具有 ODBC_TERADATA 选项的选项组。
4. 选择删除。
5. 在删除选项下，在待删除的选项下选择 ODBC_TERADATA。
6. 在 Apply immediately（立即应用）下，选择 Yes（是）可立即删除选项，选择 No（否）可在下一个维护时段删除它。
7. 选择删除。

Amazon CLI

以下命令移除 ODBC_TERADATA 选项。

对于 Linux、macOS 或 Unix：

```
aws rds remove-option-from-option-group \
    --option-group-name teradata-odbc-se-2019 \
    --options ODBC_TERADATA \
    --apply-immediately
```

对于 Windows：

```
aws rds remove-option-from-option-group ^
    --option-group-name teradata-odbc-se-2019 ^
    --options ODBC_TERADATA ^
    --apply-immediately
```

SQL Server 中对本机备份和还原的支持

通过在 SQL Server 数据库中使用本机备份和还原，您可以创建本地数据库的差异备份或完整备份，并将备份文件存储在 Amazon S3 上。然后您可以将其恢复到运行 SQL Server 的现有 Amazon RDS 数据库实例。您也可以备份 RDS for SQL Server 数据库，将其存储在 Amazon S3 中，然后在其他位置恢复它。此外，您还可以将备份还原到本地服务器或运行 SQL Server 的不同 Amazon RDS 数据库实例。有关更多信息，请参阅"使用本机备份和还原导入和导出 SQL Server 数据库"。

Amazon RDS 支持使用差异备份或完整备份文件（.bak 文件）对 Microsoft SQL Server 数据库进行本机备份和还原。

添加本机备份和还原选项

将本机备份和还原选项添加到数据库实例的一般过程如下所示：

1. 创建新的选项组，或者复制或修改现有选项组。
2. 将 `SQLSERVER_BACKUP_RESTORE` 选项添加到该选项组。
3. 将一个 Amazon Identity and Access Management (IAM) 角色与该选项相关联。该 IAM 角色必须有权访问 S3 存储桶以存储数据库备份。

   也就是说，该选项必须将使用 `arn:aws:iam::`*`account-id`*`:role/`*`role-name`* 格式的有效 Amazon Resource Name (ARN) 作为其选项设置。有关更多信息，请参阅《Amazon Web Services 一般参考》中的 Amazon 资源名称（ARN）。

   IAM 角色还必须有信任关系和附加的权限策略。信任关系允许 RDS 代入角色，权限策略定义了角色可以执行的操作。有关更多信息，请参阅"为本机备份和还原手动创建 IAM 角色"。
4. 将选项组与数据库实例相关联。

在添加本机备份和还原选项后，您无需重新启动数据库实例。一旦激活此选项组，您就可以立即开始备份和还原。

控制台

添加本机备份和还原选项

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择选项组。

3. 创建新的选项组或使用现有的选项组。有关如何创建自定义数据库选项组的信息，请参阅创建选项组。

   要使用现有的选项组，请跳到下一步。
4. 将 SQLSERVER_BACKUP_RESTORE 选项添加到选项组中。有关添加选项的更多信息，请参阅将选项添加到选项组。
5. 请执行下列操作之一：
   - 要使用现有的 IAM 角色和 Amazon S3 设置，请为 IAM 角色选择一个现有的 IAM 角色。如果使用现有的 IAM 角色，则 RDS 使用为该角色配置的 Amazon S3 设置。
   - 要创建新的角色并配置 Amazon S3 设置，请执行以下操作：
     1. 对于 IAM Role (IAM 角色)，选择 Create a new role (创建新角色)。
     2. 对于 S3 bucket（S3 存储桶），请从列表中选择一个 S3 存储桶。
     3. 对于 S3 prefix (optional)（S3 前缀（可选）），请指定要用于 Amazon S3 存储桶中存储的文件的前缀。

        该前缀可以包含文件路径，但不是必需的。如果提供一个前缀，则 RDS 将该前缀附加到所有备份文件。然后，RDS 在还原期间使用该前缀识别相关的文件，并忽略不相关的文件。例如，除了保存备份文件以外，您还可以将 S3 存储桶用于其他用途。在这种情况下，您可以使用前缀让 RDS 仅在特定文件夹及其子文件夹上执行本机备份和还原。

        如果将前缀保留空白，则 RDS 不会使用前缀标识备份文件或要还原的文件。因此，在多文件还原期间，RDS 尝试还原 S3 存储桶的每个文件夹中的每个文件。
6. 选择添加选项。
7. 将选项组应用到新的或现有的数据库实例：
   - 对于新数据库实例，请在启动实例时应用选项组。有关更多信息，请参阅“创建 Amazon RDS 数据库实例”。
   - 对于现有数据库实例，请修改实例并附加新的选项组以应用选项组。有关更多信息，请参阅 修改 Amazon RDS 数据库实例。

CLI

此过程作出以下假设：

- 您正在将 SQLSERVER_BACKUP_RESTORE 选项添加到已存在的选项组。有关添加选项的更多信息，请参阅 将选项添加到选项组。

- 您正在将选项与已存在且有权访问 S3 存储桶以存储备份的 IAM 角色关联。
- 您正在将选项组应用于已存在的数据库实例。有关更多信息，请参阅“修改 Amazon RDS 数据库实例”。

添加本机备份和还原选项

1. 将 SQLSERVER_BACKUP_RESTORE 选项添加到该选项组。

   Example

   对于 Linux、macOS 或 Unix：

```
aws rds add-option-to-option-group \
 --apply-immediately \
 --option-group-name mybackupgroup \
 --options "OptionName=SQLSERVER_BACKUP_RESTORE, \
   OptionSettings=[{Name=IAM_ROLE_ARN,Value=arn:aws:iam::account-id:role/role-
name}]"
```

   对于 Windows：

```
aws rds add-option-to-option-group ^
 --option-group-name mybackupgroup ^
 --options "[{\"OptionName\": \"SQLSERVER_BACKUP_RESTORE\", ^
 \"OptionSettings\": [{\"Name\": \"IAM_ROLE_ARN\", ^
 \"Value\": \"arn:aws:iam::account-id:role/role-name"}]}]" ^
 --apply-immediately
```

   Note

   使用 Windows 命令提示符时，必须在 JSON 代码中转义双引号 (")，方法是使用反斜杠 (\) 作为其前缀。

2. 将选项组应用于数据库实例。

   Example

   对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
```

```
 --db-instance-identifier mydbinstance \
 --option-group-name mybackupgroup \
 --apply-immediately
```

对于 Windows：

```
aws rds modify-db-instance ^
 --db-instance-identifier mydbinstance ^
 --option-group-name mybackupgroup ^
 --apply-immediately
```

修改本机备份和还原选项设置

在启用本机备份和还原选项后，您可以修改该选项的设置。有关如何修改选项设置的更多信息，请参阅修改选项设置。

删除本机备份和还原选项

您可以从数据库实例中删除该选项以禁用本机备份和还原。在删除本机备份和还原选项后，您无需重新启动数据库实例。

要从数据库实例中删除本机备份和还原选项，请执行以下操作之一：

- 从 选项所属的选项组中删除该选项。此更改会影响使用该选项组的所有数据库实例。有关更多信息，请参阅"从选项组中删除选项"。
- 修改数据库实例，并指定不包含本机备份和还原选项的不同选项组。此更改会影响单个数据库实例。您可以指定默认 (空) 选项组，或指定其他自定义选项组。有关更多信息，请参阅"修改 Amazon RDS 数据库实例"。

SQL Server 中的透明数据加密支持

Amazon RDS 支持使用透明数据加密 (TDE) 来加密运行 Microsoft SQL Server 的数据库实例中存储的数据。TDE 会在数据写入存储前自动加密这些数据，并在从存储中读取时自动解密这些数据。

Amazon RDS 支持在以下 SQL Server 版本中使用 TDE：

- SQL Server 2022 标准版和企业版
- SQL Server 2019 标准版和企业版
- SQL Server 2017 企业版
- SQL Server 2016 企业版

适用于 SQL Server 的透明数据加密使用双层密钥体系结构提供加密密钥管理。系统会从数据库主密钥中生成证书，用于保护数据加密密钥。数据库加密密钥对用户数据库上的数据执行实际的加密与解密操作。Amazon RDS 备份和管理数据库主密钥和 TDE 证书。

透明数据加密在您需要加密敏感数据的场景中使用。例如，您可能想要将数据文件和备份提供给第三方，或者想要解决有关安全的合规性问题。您不能加密 SQL Server 的系统数据库，例如 `model` 数据库或 `master` 数据库。

详细讨论透明数据加密超出了本指南的范围，但确保您了解每个加密算法和密钥的安全优缺点。有关适用于 SQL Server 的透明数据加密的信息，请参阅 Microsoft 文档中的透明数据加密 (TDE)。

主题

为 RDS for SQL Server 开启 TDE

要为 RDS for SQL Server 数据库实例开启透明数据加密，请在与该数据库实例关联的 RDS 选项组中指定 TDE 选项。

1. 确定您的数据库实例是否已与具有 TDE 选项的选项组关联。要查看与数据库实例关联的选项组，请使用 RDS 控制台、Amazon CLI 命令 describe-db-instance 或 API 操作 DescribeDBInstances。

2. 如果数据库实例未与已开启 TDE 的选项组关联，则您有两个选择。您可以创建选项组并添加 TDE 选项，或您可以修改关联选项组以添加它。

> **Note**
> 在 RDS 控制台中，该选项名为 TRANSPARENT_DATA_ENCRYPTION。在 Amazon CLI 和 RDS API 中，它名为 TDE。

   有关创建或者修改选项组的信息，请参阅 使用选项组。有关向选项组添加选项的信息，请参阅 将选项添加到选项组。

3. 将数据库实例与具有 TDE 选项的选项组相关联。有关将数据库实例与选项组关联的信息，请参阅 修改 Amazon RDS 数据库实例。

选项组注意事项

TDE 选项是一种持久性选项。除非所有数据库实例和备份都不再与选项组关联，否则无法将此选项从选项组中删除。将 TDE 选项添加到选项组之后，该选项组只能与使用 TDE 的数据库实例关联。有关选项组中持久性选项的详细信息，请参阅选项组概述。

由于 TDE 选项是一个持久性选项，因此在选项组与关联的数据库实例之间可能发生冲突。在以下情况下，可能会发生冲突：

- 当前选项组具有 TDE 选项，而您将它替换为没有 TDE 选项的选项组。
- 从数据库快照还原到新的数据库实例，但该实例不具有包含 TDE 选项的选项组。有关此方案的更多信息，请参阅选项组的注意事项。

SQL Server 性能注意事项

使用透明数据加密可能会影响 SQL Server 数据库实例的性能。

如果数据库实例上的数据库拥有至少一个加密的数据库，那么，也可能会降低未加密数据库的性能。因此，我们建议您将加密数据库和未加密数据库保存在不同的数据库实例上。

在 RDS for SQL Server 上加密数据

将 TDE 选项添加到选项组后，Amazon RDS 将生成在加密过程中使用的证书。然后，您就可以使用该证书运行为数据库实例上的数据库加密数据的 SQL 语句。

下例使用 RDS 创建的、名为 `RDSTDECertificateName` 的证书加密名为 `myDatabase` 的数据库。

```
---------- Turning on TDE -------------

-- Find an RDS TDE certificate to use
USE [master]
GO
SELECT name FROM sys.certificates WHERE name LIKE 'RDSTDECertificate%'
GO

USE [myDatabase]
GO
-- Create a database encryption key (DEK) using one of the certificates from the
 previous step
CREATE DATABASE ENCRYPTION KEY WITH ALGORITHM = AES_256
ENCRYPTION BY SERVER CERTIFICATE [RDSTDECertificateName]
GO

-- Turn on encryption for the database
ALTER DATABASE [myDatabase] SET ENCRYPTION ON
GO

-- Verify that the database is encrypted
USE [master]
GO
SELECT name FROM sys.databases WHERE is_encrypted = 1
GO
SELECT db_name(database_id) as DatabaseName, * FROM sys.dm_database_encryption_keys
GO
```

使用 TDE 加密 SQL Server 数据库所花的时间取决于几个因素。其中包括数据库实例的大小、实例是否使用预调配 IOPS、数据量及其他因素。

在 RDS for SQL Server 上备份和还原 TDE 证书

RDS for SQL Server 提供了用于备份、还原和删除 TDE 证书的存储过程。RDS for SQL Server 还提供了用于查看已还原的用户 TDE 证书的功能。

用户 TDE 证书用于将数据库还原到 RDS for SQL Server，这些数据库位于本地并已开启 TDE。这些证书具有前缀 `UserTDECertificate_`。还原数据库之后，在使其可供使用之前，RDS 会修改已开启 TDE 的数据库以使用 RDS 生成的 TDE 证书。这些证书具有前缀 `RDSTDECertificate`。

用户 TDE 证书保留在 RDS for SQL Server 数据库实例上，除非您使用 `rds_drop_tde_certificate` 存储过程删除它们。有关更多信息，请参阅 删除还原的 TDE 证书。

您可以使用用户 TDE 证书从源数据库实例还原其他数据库。要还原的数据库必须使用相同的 TDE 证书并开启 TDE。您不必再次导入（还原）同一证书。

主题

先决条件

在 RDS for SQL Server 上备份或还原 TDE 证书之前，请确保执行以下任务。针对本机备份和还原进行设置中介绍了前三个任务。

1. 创建 Amazon S3 桶以存储要备份和还原的文件。

   我们建议您使用单独的桶进行数据库备份和 TDE 证书备份。
2. 创建用于备份和还原文件的 IAM 角色。

   IAM 角色必须既是 Amazon KMS key 的用户又是其管理员。

   除了 SQL Server 本机备份和还原所需的权限外，IAM 角色还需要以下权限：
   - 针对 S3 桶资源的 `s3:GetBucketAcl`、`s3:GetBucketLocation` 和 `s3:ListBucket`
   - 针对 * 资源的 `s3:ListAllMyBuckets`
3. 将 `SQLSERVER_BACKUP_RESTORE` 选项添加到数据库实例上的选项组。

   这是对 `TRANSPARENT_DATA_ENCRYPTION` (`TDE`) 选项的补充。
4. 请确保您有对称的加密 KMS 密钥。您有以下选项：
   - 如果您的账户中有现有的 KMS 密钥，则可以使用它。无需进一步操作。
   - 如果您的账户中没有对称的加密 KMS 密钥，请按照《Amazon Key Management Service 开发人员指南》中的创建密钥中的说明创建一个 KMS 密钥。

5. 启用 Amazon S3 集成以在数据库实例和 Amazon S3 之间传输文件。

   有关启用 Amazon S3 集成的更多信息，请参阅将 Amazon RDS for SQL Server 数据库实例与 Amazon S3 集成。

限制

使用存储过程备份和还原 TDE 证书具有以下限制：

- 必须将 SQLSERVER_BACKUP_RESTORE 和 TRANSPARENT_DATA_ENCRYPTION (TDE) 这两个选项添加到与数据库实例关联的数据库选项组。
- 多可用区数据库实例不支持 TDE 证书备份和还原。
- 不支持取消 TDE 证书备份和还原任务。
- 您不能使用用户 TDE 证书对 RDS for SQL Server 数据库实例上的任何其他数据库进行 TDE 加密。您可以使用它从源数据库实例中仅还原已启用 TDE 并且使用相同 TDE 证书的其他数据库。
- 您只能删除用户 TDE 证书。
- RDS 上支持的最大用户 TDE 证书数量为 10 个。如果此数字超过 10，请删除未使用的 TDE 证书，然后重试。
- 证书名称不能为空或为 null。
- 还原证书时，证书名称不能包含 RDSTDECERTIFICATE 关键字，并且必须以 UserTDECertificate_ 前缀开头。
- @certificate_name 参数只能包含以下字符：a-z、0-9、@、$、# 和下划线 (_)。
- @certificate_file_s3_arn 的文件扩展名必须为 .cer（不区分大小写）。
- @private_key_file_s3_arn 的文件扩展名必须为 .pvk（不区分大小写）。
- 私有密钥文件的 S3 元数据必须包括 x-amz-meta-rds-tde-pwd 标签。有关更多信息，请参阅 备份和还原本地数据库的 TDE 证书。

备份 TDE 证书

要备份 TDE 证书，请使用 rds_backup_tde_certificate 存储过程。它具有以下语法。

```
EXECUTE msdb.dbo.rds_backup_tde_certificate
    @certificate_name='UserTDECertificate_certificate_name |
 RDSTDECertificatetimestamp',
    @certificate_file_s3_arn='arn:aws:s3:::bucket_name/certificate_file_name.cer',
```

```
    @private_key_file_s3_arn='arn:aws:s3:::bucket_name/key_file_name.pvk',
    @kms_password_key_arn='arn:aws:kms:region:account-id:key/key-id',
    [@overwrite_s3_files=0|1];
```

以下参数为必需参数：

- `@certificate_name` – 要备份的 TDE 证书的名称。
- `@certificate_file_s3_arn` – Amazon S3 中证书备份文件的目标 Amazon 资源名称 (ARN)。
- `@private_key_file_s3_arn` – 保护 TDE 证书的私有密钥文件的目标 S3 ARN。
- `@kms_password_key_arn` – 用于加密私有密钥密码的对称 KMS 密钥的 ARN。

以下参数为可选参数：

- `@overwrite_s3_files` – 指示是否覆盖 S3 中的现有证书和私有密钥文件：
  - `0` – 不覆盖现有文件。此值是默认值。

    如果文件已存在，则将 `@overwrite_s3_files` 设置为 0 会返回错误。
  - 1 – 覆盖具有指定名称的现有文件，即使该文件不是备份文件也是如此。

Example 备份 TDE 证书

```
EXECUTE msdb.dbo.rds_backup_tde_certificate
    @certificate_name='RDSTDECertificate20211115T185333',
    @certificate_file_s3_arn='arn:aws:s3:::TDE_certs/mycertfile.cer',
    @private_key_file_s3_arn='arn:aws:s3:::TDE_certs/mykeyfile.pvk',
    @kms_password_key_arn='arn:aws:kms:us-
west-2:123456789012:key/AKIAIOSFODNN7EXAMPLE',
    @overwrite_s3_files=1;
```

还原 TDE 证书

可以使用 `rds_restore_tde_certificate` 存储过程来还原（导入）用户 TDE 证书。它具有以下语法。

```
EXECUTE msdb.dbo.rds_restore_tde_certificate
    @certificate_name='UserTDECertificate_certificate_name',
    @certificate_file_s3_arn='arn:aws:s3:::bucket_name/certificate_file_name.cer',
    @private_key_file_s3_arn='arn:aws:s3:::bucket_name/key_file_name.pvk',
```

```
    @kms_password_key_arn='arn:aws:kms:region:account-id:key/key-id';
```

以下参数为必需参数：

- @certificate_name –要还原的 TDE 证书的名称。名称必须以 UserTDECertificate_ 前缀开头。
- @certificate_file_s3_arn – 用于还原 TDE 证书的备份文件的 S3 ARN。
- @private_key_file_s3_arn – 要还原的 TDE 证书的私有密钥备份文件的 S3 ARN。
- @kms_password_key_arn – 用于加密私有密钥密码的对称 KMS 密钥的 ARN。

Example 还原 TDE 证书

```
EXECUTE msdb.dbo.rds_restore_tde_certificate
    @certificate_name='UserTDECertificate_myTDEcertificate',
    @certificate_file_s3_arn='arn:aws:s3:::TDE_certs/mycertfile.cer',
    @private_key_file_s3_arn='arn:aws:s3:::TDE_certs/mykeyfile.pvk',
    @kms_password_key_arn='arn:aws:kms:us-
west-2:123456789012:key/AKIAIOSFODNN7EXAMPLE';
```

查看还原的 TDE 证书

可以使用 rds_fn_list_user_tde_certificates 函数来查看已还原（导入）的用户 TDE 证书。它具有以下语法。

```
SELECT * FROM msdb.dbo.rds_fn_list_user_tde_certificates();
```

输出与以下内容类似。此处并未显示所有列。

name	certificate_id	principal_id	pvt_key_encryption_type_desc	issuer_name	cert_serial_number	thumbprint	subject	start_date	expiry_date	pvt_key_last_backup_date
UserTDECertificate_tde_c	343	1	ENCRYPTED_BY_MASTER_KEY	AnyCompany Shippi	79 3e 57	0x6BB2341103 80B	AnyCompany Shippi	2022-05	2023-05	NULL

					a3 69 fd 1d 9e 47 2c 32 67 1d 9c ca af	FE1BA2 C69509 5B5		19:49: 000000	19:49: 000000	

删除还原的 TDE 证书

要删除您未使用的已还原（导入）的用户 TDE 证书，请使用 `rds_drop_tde_certificate` 存储过程。它具有以下语法。

```
EXECUTE msdb.dbo.rds_drop_tde_certificate
 @certificate_name='UserTDECertificate_certificate_name';
```

以下参数是必需参数：

- `@certificate_name` – 要删除的 TDE 证书的名称。

您只能删除已还原（导入）的 TDE 证书。您不能删除 RDS 创建的证书。

Example 删除 TDE 证书

```
EXECUTE msdb.dbo.rds_drop_tde_certificate
 @certificate_name='UserTDECertificate_myTDEcertificate';
```

备份和还原本地数据库的 TDE 证书

您可以备份本地数据库的 TDE 证书，然后将其还原到 RDS for SQL Server。您还可以将 RDS for SQL Server TDE 证书还原到本地数据库实例。

以下过程备份 TDE 证书和私有密钥。私有密钥使用从对称加密 KMS 密钥生成的数据密钥进行加密。

备份本地 TDE 证书

1. 使用 Amazon CLI 命令 [generate-data-key](https://docs.aws.amazon.com/cli/latest/reference/kms/generate-data-key.html) 生成数据密钥。

```
aws kms generate-data-key \
    --key-id my_KMS_key_ID \
    --key-spec AES_256
```

   输出与以下内容类似。

```
{
"CiphertextBlob": "AQIDAHimL2NEoAlOY6Bn7LJfnxi/OZe9kTQo/
XQXduug1rmerwGiL7g5ux4av9GfZLxYTDATAAAAfjB8BgkqhkiG9w0B
BwagbzBtAgEAMGgGCSqGSIb3DQEHATAeBglghkgBZQMEAS4wEQQMyCxLMi7GRZgKqD65AgEQgDtjvZLJo2cQ31Vetng
2RezQy3sAS6ZHrCjfnfn0c65bFdhsXxjSMnudIY7AKw==",
"Plaintext": "U/fpGtmzGCYBi8A2+0/9qcRQRK2zmG/aOn939ZnKi/0=",
"KeyId": "arn:aws:kms:us-west-2:123456789012:key/1234abcd-00ee-99ff-88dd-
aa11bb22cc33"
}
```

   您可以在下一步中使用纯文本输出作为私有密钥密码。

2. 备份 TDE 证书，如以下示例所示。

```
BACKUP CERTIFICATE myOnPremTDEcertificate TO FILE = 'D:\tde-cert-backup.cer'
WITH PRIVATE KEY (
FILE = 'C:\Program Files\Microsoft SQL Server\MSSQL14.MSSQLSERVER\MSSQL\DATA\cert-
backup-key.pvk',
ENCRYPTION BY PASSWORD = 'U/fpGtmzGCYBi8A2+0/9qcRQRK2zmG/aOn939ZnKi/0=');
```

3. 将证书备份文件保存到 Amazon S3 证书桶。

4. 将私有密钥备份文件保存到 S3 证书桶，并在文件的元数据中使用以下标签：

   • 密钥 – `x-amz-meta-rds-tde-pwd`

   • 值 – `CiphertextBlob` 值来自生成数据密钥过程，如以下示例所示。

```
AQIDAHimL2NEoAlOY6Bn7LJfnxi/OZe9kTQo/
XQXduug1rmerwGiL7g5ux4av9GfZLxYTDATAAAAfjB8BgkqhkiG9w0B
BwagbzBtAgEAMGgGCSqGSIb3DQEHATAeBglghkgBZQMEAS4wEQQMyCxLMi7GRZgKqD65AgEQgDtjvZLJo2cQ31Vet
```

```
2RezQy3sAS6ZHrCjfnfn0c65bFdhsXxjSMnudIY7AKw==
```

以下过程将 RDS for SQL Server TDE 证书还原到本地数据库实例。您可以使用证书备份、相应的私有密钥文件和数据密钥在目标数据库实例上复制和还原 TDE 证书。还原的证书由新服务器的数据库主密钥加密。

还原 TDE 证书

1. 将 TDE 证书备份文件和私有密钥文件从 Amazon S3 复制到目标实例。有关从 Amazon S3 中复制文件的更多信息，请参阅在 RDS for SQL Server 和 Amazon S3 之间传输文件。
2. 使用 KMS 密钥解密输出密码文本，以检索数据密钥的纯文本。密码文本位于私有密钥备份文件的 S3 元数据中。

   ```
   aws kms decrypt \
       --key-id my_KMS_key_ID \
       --ciphertext-blob fileb://exampleCiphertextFile | base64 -d \
       --output text \
       --query Plaintext
   ```

   您可以在下一步中使用纯文本输出作为私有密钥密码。
3. 使用以下 SQL 命令还原 TDE 证书。

   ```
   CREATE CERTIFICATE myOnPremTDEcertificate FROM FILE='D:\tde-cert-backup.cer'
   WITH PRIVATE KEY (FILE = N'D:\tde-cert-key.pvk',
   DECRYPTION BY PASSWORD = 'plain_text_output');
   ```

有关 KMS 解密的更多信息，请参阅《Amazon CLI 命令参考》的 KMS 部分中的解密。

在目标数据库实例上还原 TDE 证书后，您可以使用该证书还原加密的数据库。

> **Note**
>
> 您可以使用同一 TDE 证书对源数据库实例上的多个 SQL Server 数据库进行加密。要将多个数据库迁移到目标实例，请将与它们关联的 TDE 证书仅复制到目标实例（仅限一次）。

为 RDS for SQL Server 关闭 TDE

要为 RDS for SQL Server 数据库实例关闭 TDE，首先应确保数据库实例上没有留下加密的对象。为此，请解密对象或删除它们。如果该数据库实例中存在任何加密的对象，则您不能为该数据库实例关闭 TDE。当您使用控制台从选项组中删除 TDE 选项时，控制台会指明它正在处理。此外，如果选项组与加密数据库实例或数据库快照关联，则会创建一个错误事件。

下例从名为 `customerDatabase` 的数据库中删除 TDE 加密。

```
------------- Removing TDE ----------------

USE [customerDatabase]
GO

-- Turn off encryption of the database
ALTER DATABASE [customerDatabase]
SET ENCRYPTION OFF
GO

-- Wait until the encryption state of the database becomes 1. The state is 5
 (Decryption in progress) for a while
SELECT db_name(database_id) as DatabaseName, * FROM sys.dm_database_encryption_keys
GO

-- Drop the DEK used for encryption
DROP DATABASE ENCRYPTION KEY
GO

-- Alter to SIMPLE Recovery mode so that your encrypted log gets truncated
USE [master]
GO
ALTER DATABASE [customerDatabase] SET RECOVERY SIMPLE
GO
```

解密所有对象后，您有两个选项：

1. 您可以修改数据库实例，使其与没有 TDE 选项的选项组关联。
2. 您可以从选项组中删除 TDE 选项。

SQL Server Audit

在 Amazon RDS 中，您可以使用内置 SQL Server 审核机制来审核 Microsoft SQL Server 数据库。您可以按照为本地数据库服务器创建审核和审核规范的相同方式来创建它们。

RDS 使用您提供的 IAM 角色将完成的审核日志上传到 S3 存储桶。如果您启用保留，则 RDS 会在配置的时间段内将您的审核日志保留在您的数据库实例中。

有关更多信息，请参阅 Microsoft SQL Server 文档中的 SQL Server Audit（数据库引擎）。

SQL Server 审计与数据库活动流

您可以使用 RDS 的数据库活动流，将 SQL Server 审计事件与 Imperva、McAfee 和 IBM 中的数据库活动监控工具集成。有关使用 RDS SQL Server 的数据库活动流进行审计的详细信息，请参阅Microsoft SQL Server 中的审计

主题

SQL Server Audit 支持

在 Amazon RDS 中，从 SQL Server 2016 开始，SQL Server 的所有版本都支持服务器级审计，而企业版还支持数据库级审计。从 SQL Server 2016 (13.x) SP1 开始，所有版本支持服务器级别和数据库级别审核。有关更多信息，请参阅 SQL Server 文档中的 SQL Server Audit（数据库引擎）。

RDS 支持为 SQL Server Audit 配置以下选项设置。

选项设置	有效值	描述
`IAM_ROLE_ARN`	采用 `arn:aws:iam::` *`account-i`*	IAM 角色的 ARN，该角色对您想要用于存储审核日志的 S3 存储桶授予访问权。有关

选项设置	有效值	描述
	d :role/*role-name* 格式的有效 Amazon Resource Name (ARN)。	更多信息，请参阅《Amazon Web Services 一般参考》中的 Amazon 资源名称（ARN）。
S3_BUCKET_ARN	有效 ARN 为 arn:aws:s3::: *amzn-s3-demo-bucket* 或 arn:aws:s3::: *amzn-s3-demo-bucket* /key-prefix 格式	您想要用于存储审核日志的 S3 存储桶的 ARN。
ENABLE_COMPRESSION	true 或者 false	控制审核日志压缩。默认情况下，启用压缩（设置为 true）。
RETENTION_TIME	0 到 840	SQL Server 审核记录在您的 RDS 实例上保留的时间（按小时计）。默认情况下，禁用保留。

将 SQL Server Audit 添加到数据库实例选项

启用 SQL Server Audit 需要两个步骤：在数据库实例上启用选项，然后在 SQL Server 内启用功能。将 SQL Server Audit 选项添加到数据库实例的流程如下：

1. 创建新的选项组，或者复制或修改现有选项组。
2. 添加和配置所有必需的选项。
3. 将选项组与数据库实例相关联。

添加 SQL Server Audit 选项后，您无需重新启动数据库实例。只要选项组处于活动状态，您就可以创建审核并在 S3 存储桶中存储审核日志。

若要在数据库实例的选项组中添加和配置 SQL Server Audit

1. 选择以下选项之一：

   - 使用现有选项组。

- 创建自定义数据库选项组并使用该选项组。有关更多信息，请参阅“创建选项组”。

2. 将 SQLSERVER_AUDIT 选项添加到选项组，然后配置选项设置。有关添加选项的更多信息，请参阅 将选项添加到选项组。

    - 对于 IAM 角色，如果已经有具备必需策略的 IAM 角色，则可选择该角色。要创建新的 IAM 角色，请选择创建新角色。有关必需策略的信息，请参阅 手动为 SQL Server Audit 创建 IAM 角色。
    - 对于选择 S3 目标，如果已经有要使用的 S3 存储桶，则可选择该存储桶。要创建 S3 存储桶，请选择 Create a New S3 Bucket（创建新的 S3 存储桶）。
    - 对于启用压缩，请将此选项保留为选中状态以压缩审核文件。默认情况下，压缩已启用。若要禁用压缩，请取消选中 Enable Compression (启用压缩)。
    - 对于审核日志保留，如需在数据库实例上保留审核记录，请选择此选项。指定保留时间，按小时计。最长保留时间为 35 天。

3. 将选项组应用到新的或现有的数据库实例。选择以下选项之一：

    - 如果您正在创建新的数据库实例，请在启动实例时应用选项组。
    - 在现有数据库实例上，请通过修改实例并附加新的选项组来应用选项组。有关更多信息，请参阅“修改 Amazon RDS 数据库实例”。

修改 SQL Server Audit 选项

启用 SQL Server Audit 选项后，您可以修改各项设置。有关如何修改选项设置的信息，请参阅修改选项设置。

从数据库实例选项中删除 SQL Server Audit

您可以通过禁用审核并删除选项来关闭 SQL Server Audit 功能。

若要删除审核

1. 禁用 SQL Server 内的所有审核设置。若要了解审核在何处运行，请查询 SQL Server 安全性目录视图。有关更多信息，请参阅 Microsoft SQL Server 文档中的安全性目录视图。
2. 从数据库实例中删除 SQL Server Audit 选项。选择以下选项之一：

    - 从数据库实例使用的选项组中删除 SQL Server Audit 选项。此更改会影响使用相同选项组的所有数据库实例。有关更多信息，请参阅“从选项组中删除选项”。

  - 修改数据库实例，然后选择一个没有 SQL Server Audit 选项的选项组。此更改仅影响您修改的数据库实例。您可以指定默认 (空) 选项组，或指定其他自定义选项组。有关更多信息，请参阅“修改 Amazon RDS 数据库实例”。
3. 从数据库实例中删除 SQL Server Audit 选项之后，不需要重新启动实例。从您的 S3 存储桶中删除不需要的审核文件。

使用 SQL Server Audit

您可以按照与控制本地数据库服务器相同的方式控制服务器审核、服务器审核规范和数据库审核规范。

创建审核

按照为本地数据库服务器创建服务器审核的相同方式来创建它们。有关如何创建服务器审核的信息，请参阅 Microsoft SQL Server 文档中的创建服务器审核。

为避免错误，请遵循以下限制：

- 不要超过每个实例支持的服务器审核最大数目：50。
- 指示 SQL Server 将数据写入二进制文件。
- 请勿在服务器审核名称中使用 `RDS_` 作为前缀。
- 对于 `FILEPATH`，请指定 `D:\rdsdbdata\SQLAudit`。
- 对于 `MAXSIZE`，请指定 2 MB 至 50 MB 之间的大小。
- 请勿配置 `MAX_ROLLOVER_FILES` 或 `MAX_FILES`。
- 请勿将 SQL Server 配置为在数据库实例未能写入审核记录时将其关闭。

创建审核规范

按照为本地数据库服务器创建服务器审核规范和数据库审核规范相同的方式来创建它们。有关创建审核规范的信息，请参阅 Microsoft SQL Server 文档中的创建服务器审核规范和创建数据库审核规范。

为避免错误，请勿在数据库审核规范或服务器审核规范的名称中使用 `RDS_` 作为前缀。

查看审核日志

审核日志存储在 `D:\rdsdbdata\SQLAudit` 中。

SQL Server 完成写入到审核日志文件之后（当文件达到其大小限制时），Amazon RDS 会将文件上传到您的 S3 存储桶。如果启用了保留，Amazon RDS 会将文件移入到保留文件夹：D:\rdsdbdata\SQLAudit\transmitted。

有关配置保留的信息，请参阅将 SQL Server Audit 添加到数据库实例选项。

在上传审核日志文件之前，审核记录会一直保留在数据库实例上。您可以通过运行以下命令来查看审核记录。

```
SELECT   *
 FROM     msdb.dbo.rds_fn_get_audit_file
          ('D:\rdsdbdata\SQLAudit\*.sqlaudit'
          , default
          , default )
```

您可以使用相同的命令来查看保留文件夹中的审核记录，只需将筛选条件更改为 D:\rdsdbdata\SQLAudit\transmitted*.sqlaudit。

```
SELECT   *
 FROM     msdb.dbo.rds_fn_get_audit_file
          ('D:\rdsdbdata\SQLAudit\transmitted\*.sqlaudit'
          , default
          , default )
```

将 SQL Server Audit 与多可用区实例结合使用

对于多可用区实例，将审核日志文件发送到 Amazon S3 的过程与单可用区实例类似。但是，二者之间存在一些重要区别：

- 数据库审核规范对象复制到所有节点。
- 服务器审核和服务审核规范不会复制到辅助节点。相反，您必须手动创建或修改它们。

若要从两个节点捕获服务器审核或服务器审核规范：

1. 在主节点上创建服务器审核或服务器审核规范。
2. 故障转移到辅助节点，并在辅助节点上使用相同名称和 GUID 创建服务器审核或服务器审核规范。使用 AUDIT_GUID 参数指定 GUID。

配置 S3 存储桶

审核日志文件会自动从数据库实例上传到您的 S3 存储桶。以下限制适用于您用作审核文件目标的 S3 存储桶。

- 它必须与数据库实例位于同一 Amazon 区域。
- 它不得对公众开放。
- 存储桶拥有者也必须是 IAM 角色拥有者。
- IAM 角色必须对与 S3 存储桶服务器端加密关联的客户自主管理型 KMS 密钥拥有权限。

用于存储数据的目标键遵循以下命名模式：`amzn-s3-demo-bucket/key-prefix/instance-name/audit-name/node_file-name.ext`

> **Note**
>
> 您使用 (S3_BUCKET_ARN) 选项设置来设定存储桶名称和键前缀值。

模式由以下元素组成：

- ***amzn-s3-demo-bucket*** – S3 存储桶的名称。
- **key-prefix** – 您想要为审核日志使用的自定义键前缀。
- **instance-name** – 您的 Amazon RDS 实例的名称。
- **audit-name** – 审核的名称。
- **node** – 审核日志来源节点（node1 或 node2）的标识符。单可用区实例有一个节点，多可用区实例有两个复制节点。这些不是主要和辅助节点，因为主要和辅助节点的角色会随时间而变化。相反，节点标识符是一个简单的标签。
  - **node1** – 第一个复制节点（单可用区只有一个节点）。
  - **node2** – 第二个复制节点（多可用区有两个节点）。
- **file-name** – 目标文件名称。文件名按原样从 SQL Server 获取。
- **ext** – 文件的扩展名（zip 或 sqlaudit）：
  - **zip** – 是否已启用压缩（默认）。
  - **sqlaudit** – 是否已禁用压缩。

手动为 SQL Server Audit 创建 IAM 角色

通常，当您创建新选项时，Amazon Web Services Management Console会为您创建 IAM 角色和 IAM 信任策略。但是，您可以手动创建与 SQL Server Audit 一起使用的新 IAM 角色，以便您可以根据自己的任何附加需求对其进行定制。为此，创建一个 IAM 角色并委托权限，以便 Amazon RDS 服务可使用您的 Amazon S3 存储桶。创建这个 IAM 角色时，附加信任和权限策略。信任策略允许 Amazon RDS 担任此角色。权限策略定义此角色可以执行的操作。有关更多信息，请参阅 Amazon Identity and Access Management 用户指南中的创建角色以向 Amazon 服务委托权限。

您可以使用本节中的示例来创建所需的信任关系和权限策略。

以下示例显示了 SQL Server Audit 的信任关系。该关系使用服务主体 `rds.amazonaws.com` 来允许 RDS 写入到 S3 存储桶。服务委托人是一个标识符，用于向服务授予权限。任何时候您通过这种方式允许访问 `rds.amazonaws.com`，即表示您允许 RDS 代表您执行操作。有关服务委托人的更多信息，请参阅 Amazon JSON 策略元素：Principal。

Example SQL Server Audit 的信任关系

```
{
     "Version": "2012-10-17",
     "Statement": [
         {
             "Effect": "Allow",
             "Principal": {
                 "Service": "rds.amazonaws.com"
             },
             "Action": "sts:AssumeRole"
         }
     ]
 }
```

我们建议在基于资源的信任关系中使用 `aws:SourceArn` 和 `aws:SourceAccount` 全局条件上下文键，以此限制服务对特定资源的权限。这是防范混淆代理问题最有效的方法。

您可以使用这两个全局条件上下文键并让 `aws:SourceArn` 值包含账户 ID。在这种情况下，当 `aws:SourceAccount` 值和 `aws:SourceArn` 值中的账户使用相同策略语句时，确保二者使用相同的账户 ID。

- 如果您想对单个资源进行跨服务访问，请使用 `aws:SourceArn`。
- 如果您想允许该账户中的任何资源与跨服务使用操作相关联，请使用 `aws:SourceAccount`。

在信任关系中，请务必使用 `aws:SourceArn` 全局条件上下文键和访问角色资源的完整 Amazon Resource Name (ARN)。对于 SQL Server Audit，请确保同时包含数据库选项组和数据库实例，如以下示例所示。

Example 与 SQL Server Audit 的全局条件上下文键的信任关系

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "rds.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceArn": [
                        "arn:aws:rds:Region:my_account_ID:db:db_instance_identifier",
                        "arn:aws:rds:Region:my_account_ID:og:option_group_name"
                    ]
                }
            }
        }
    ]
}
```

在以下 SQL Server Audit 权限策略示例中，我们为 Amazon S3 存储桶指定 ARN。您可以使用 ARN 来识别您想要为其授予访问权的特定账户、用户或角色。有关使用 ARN 的更多信息，请参阅 [Amazon Resource Name (ARN)]。

Example SQL Server Audit 的权限策略

```
{
     "Version": "2012-10-17",
     "Statement": [
         {
             "Effect": "Allow",
             "Action": "s3:ListAllMyBuckets",
             "Resource": "*"
         },
         {
```

```
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket",
                "s3:GetBucketAcl",
                "s3:GetBucketLocation"
            ],
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:ListMultipartUploadParts",
                "s3:AbortMultipartUpload"
            ],
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/key_prefix/*"
        }
    ]
}
```

> **Note**
>
> 要验证同一 `s3:ListAllMyBuckets` 账户是否同时拥有 S3 存储桶和 SQL Server 数据库实例，需要执行 Amazon 操作。该操作会列出该账户中的存储桶的名称。
> S3 存储桶命名空间是全局的。如果您意外删除了您的存储桶，则另一用户可以在其他账户中创建具有相同名称的存储桶。然后将 SQL Server Audit 数据写入新的存储桶。

Amazon RDS for SQL Server 中对 SQL Server Analysis Services 的支持

Microsoft SQL Server Analysis Services (SSAS) 属于 Microsoft Business Intelligence (MSBI) 套件。SSAS 是一个在线分析处理 (OLAP) 和数据挖掘工具，安装在 SQL Server 中。您可以使用 SSAS 分析数据来帮助制定业务决策。SSAS 不同于 SQL Server 关系数据库，因为 SSAS 针对商业智能环境中常见的查询和计算进行了优化。

您可以为现有或新数据库实例启用 SSAS。它安装在与您的数据库引擎相同的数据库实例上。有关 SSAS 的更多信息，请参阅 Microsoft 分析服务文档。

Amazon RDS 支持在 SQL Server 标准版和企业版的以下版中使用 SSAS：

- 表格模式：
  - SQL Server 2019，版本 15.00.4043.16.v1 和更高版本
  - SQL Server 2017，版本 14.00.3223.3.v1 和更高版本
  - SQL Server 2016，版本 13.00.5426.0.v1 和更高版本
- 多维模式：
  - SQL Server 2019，版本 15.00.4153.1.v1 和更高版本
  - SQL Server 2017，版本 14.00.3381.3.v1 和更高版本
  - SQL Server 2016，版本 13.00.5882.1.v1 和更高版本

目录

限制

以下限制适用于在 RDS for SQL Server 上使用 SSAS：

- RDS for SQL Server 支持在表格模式或多维模式下运行 SSAS。有关更多信息，请参阅 Microsoft 文档中的比较表格和多维解决方案。
- 一次只能使用一个 SSAS 模式。在更改模式之前，请确保删除所有 SSAS 数据库。

  有关更多信息，请参阅 更改 SSAS 模式。
- 不支持多可用区实例。
- 实例必须使用自行管理的 Active Directory 或 Amazon Directory Service for Microsoft Active Directory 才能进行 SSAS 身份验证。有关更多信息，请参阅 将 Active Directory 用于 RDS for SQL Server。
- 用户不会被授予 SSAS 服务器管理员访问权限，但可以向他们授予数据库级管理员访问权限。
- 唯一支持访问 SSAS 的端口是 2383。
- 您不能直接部署项目。我们提供 RDS 存储过程来执行部署。有关更多信息，请参阅“在 Amazon RDS 上部署 SSAS 项目”。
- 部署期间不支持进行处理。
- 不支持使用 .xmla 文件进行部署。
- SSAS 项目输入文件和数据库备份输出文件只能位于数据库实例的 `D:\S3` 文件夹中。

启用 SSAS

使用以下过程为数据库实例启用 SSAS：

1. 创建新的选项组或选择现有的选项组。
2. 将 SSAS 选项添加到该选项组。
3. 将选项组与数据库实例相关联。
4. 允许对 SSAS 侦听器端口的 Virtual Private Cloud (VPC) 安全组进行入站访问。
5. 启用 Amazon S3 集成。

为 SSAS 创建选项组

使用 Amazon Web Services Management Console 或 Amazon CLI 创建对应于计划使用的 SQL Server 引擎和数据库实例版本的选项组。

> **Note**
> 如果现有选项组针对的是正确的 SQL Server 引擎和版本，也可以使用现有选项组。

控制台

以下控制台过程创建适用于 SQL Server 标准版 2017 的选项组。

创建选项组

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择选项组。
3. 选择创建组。
4. 在创建选项组窗格中，执行以下操作：
   a. 对于名称，输入选项组的名称，该名称在 Amazon 账户内具有唯一性，例如 **ssas-se-2017**。此名称只能包含字母、数字和连字符。
   b. 对于描述，输入选项组的简要描述，例如 **SSAS option group for SQL Server SE 2017**。此说明用于显示说明。
   c. 对于引擎，选择 sqlserver-se。

   d. 对于主引擎版本，选择 14.00。

5. 选择创建。

CLI

以下 CLI 示例为 SQL Server 标准版 2017 创建一个选项组。

创建选项组

- 使用以下命令之一。

  Example

  对于 Linux、macOS 或 Unix：

```
aws rds create-option-group \
    --option-group-name ssas-se-2017 \
    --engine-name sqlserver-se \
    --major-engine-version 14.00 \
    --option-group-description "SSAS option group for SQL Server SE 2017"
```

  对于 Windows：

```
aws rds create-option-group ^
    --option-group-name ssas-se-2017 ^
    --engine-name sqlserver-se ^
    --major-engine-version 14.00 ^
    --option-group-description "SSAS option group for SQL Server SE 2017"
```

将 SSAS 选项添加到选项组

接下来，使用 Amazon Web Services Management Console或 Amazon CLI 将 SSAS 选项添加到选项组。

控制台

添加 SSAS 选项

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。

2. 在导航窗格中，选择选项组。
3. 选择刚刚创建的选项组。
4. 选择添加选项。
5. 在 Option details (选项详细信息) 下，为 Option name (选项名称) 选择 SSAS。
6. 在选项设置下，执行以下操作：

    a. 在 Max memory（最大内存）中，输入 10–80 这个范围内的一个值。

       Max memory (最大内存) 指定上限阈值，超过该阈值后，SSAS 将开始更积极地释放内存，以便为正在运行的请求以及新的高优先级请求释放空间。该数字是占数据库实例总内存的百分比。允许的值是 10–80；默认值为 45。

    b. 在 Mode（模式）中选择 SSAS 服务器模式：Tabular（表格）或 Multidimensional（多维）。

       如果没有看到 Mode（模式）选项设置，则表示 Amazon 区域不支持多维模式。有关更多信息，请参阅 限制。

       默认为 Tabular（表格）模式。

    c. 对于 Security groups (安全组)，请选择要与选项关联的 VPC 安全组。

> **Note**
>
> 用于访问 SSAS 的端口 2383 已预先填充。

7. 在 计划 (Scheduling) 下，选择是立即添加选项还是在下一个维护时段添加选项。
8. 选择添加选项。

CLI

添加 SSAS 选项

1. 使用以下参数创建 JSON 文件 ，例如 `ssas-option.json`：

    • `OptionGroupName` – 您之前创建或选择的选项组的名称（以下示例中的 `ssas-se-2017`）。

    • `Port` – 用于访问 SSAS 的端口。唯一支持的端口是 2383。

    • `VpcSecurityGroupMemberships`：RDS 数据库实例的 VPC 安全组成员。

- MAX_MEMORY – 上限阈值，超过该阈值后，SSAS 将开始更积极地释放内存，以便为正在运行的请求以及新的高优先级请求释放空间。该数字是占数据库实例总内存的百分比。允许的值是 10–80；默认值为 45。
- MODE：SSAS 服务器模式，即 Tabular 和 Multidimensional，默认为 Tabular。

  如果收到的错误消息显示 MODE 选项设置无效，则表示 Amazon 区域不支持多维模式。有关更多信息，请参阅 限制。

以下是具有 SSAS 选项设置的 JSON 文件示例。

```
{
"OptionGroupName": "ssas-se-2017",
"OptionsToInclude": [
 {
 "OptionName": "SSAS",
 "Port": 2383,
 "VpcSecurityGroupMemberships": ["sg-0abcdef123"],
 "OptionSettings": [{"Name":"MAX_MEMORY","Value":"60"},
{"Name":"MODE","Value":"Multidimensional"}]
 }],
"ApplyImmediately": true
}
```

2. 将 SSAS 选项添加到该选项组。

   Example

   对于 Linux、macOS 或 Unix：

```
aws rds add-option-to-option-group \
    --cli-input-json file://ssas-option.json \
    --apply-immediately
```

   对于 Windows：

```
aws rds add-option-to-option-group ^
    --cli-input-json file://ssas-option.json ^
    --apply-immediately
```

将选项组与数据库实例关联

您可以使用控制台或 CLI 将选项组与数据库实例关联。

控制台

将您的选项组与新的或现有的数据库实例关联：

- 对于新数据库实例，请在启动实例时将选项组与数据库实例关联。有关更多信息，请参阅“创建 Amazon RDS 数据库实例”。
- 对于现有数据库实例，请修改实例并将新选项组与其关联。有关更多信息，请参阅 修改 Amazon RDS 数据库实例。

> **Note**
> 如果您使用现有实例，则该实例必须已关联 Active Directory 域和 Amazon Identity and Access Management (IAM) 角色。如果您创建新实例，请指定现有的 Active Directory 域和 IAM 角色。有关更多信息，请参阅 将 Active Directory 用于 RDS for SQL Server。

CLI

您可将您的选项组与新的或现有的数据库实例关联。

> **Note**
> 如果您使用现有实例，则该实例必须已经关联了 Active Directory 域和 IAM 角色。如果您创建新实例，请指定现有的 Active Directory 域和 IAM 角色。有关更多信息，请参阅 将 Active Directory 用于 RDS for SQL Server。

创建使用选项组的数据库实例

- 指定创建选项组时使用的相同数据库引擎类型和主要版本。

  Example

  对于 Linux、macOS 或 Unix：

```
aws rds create-db-instance \
    --db-instance-identifier myssasinstance \
```

```
    --db-instance-class db.m5.2xlarge \
    --engine sqlserver-se \
    --engine-version 14.00.3223.3.v1 \
    --allocated-storage 100 \
    --manage-master-user-password \
    --master-username admin \
    --storage-type gp2 \
    --license-model li \
    --domain-iam-role-name my-directory-iam-role \
    --domain my-domain-id \
    --option-group-name ssas-se-2017
```

对于 Windows：

```
aws rds create-db-instance ^
    --db-instance-identifier myssasinstance ^
    --db-instance-class db.m5.2xlarge ^
    --engine sqlserver-se ^
    --engine-version 14.00.3223.3.v1 ^
    --allocated-storage 100 ^
    --manage-master-user-password ^
    --master-username admin ^
    --storage-type gp2 ^
    --license-model li ^
    --domain-iam-role-name my-directory-iam-role ^
    --domain my-domain-id ^
    --option-group-name ssas-se-2017
```

修改数据库实例以关联选项组

- 使用以下命令之一。

  Example

  对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier myssasinstance \
    --option-group-name ssas-se-2017 \
    --apply-immediately
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier myssasinstance ^
    --option-group-name ssas-se-2017 ^
    --apply-immediately
```

允许对 VPC 安全组的入站访问

在与数据库实例关联的 VPC 安全组中，为指定的 SSAS 侦听器端口创建入站规则。有关设置安全组的更多信息，请参阅通过创建安全组提供对 VPC 中的数据库实例的访问。

启用 Amazon S3 集成

要将模型配置文件下载到主机进行部署，请使用 Amazon S3 集成。有关更多信息，请参阅 将 Amazon RDS for SQL Server 数据库实例与 Amazon S3 集成。

在 Amazon RDS 上部署 SSAS 项目

在 RDS 上，您不能使用 SQL Server Management Studio (SSMS) 直接部署 SSAS 项目。要部署项目，请使用 RDS 存储过程。

> **Note**
>
> 不支持使用 .xmla 文件进行部署。

在部署项目之前，请确保满足以下条件：

- Amazon S3 集成已启用。有关更多信息，请参阅 将 Amazon RDS for SQL Server 数据库实例与 Amazon S3 集成。
- `Processing Option` 配置设置已设置为 `Do Not Process`。此设置意味着部署后不会进行任何处理。
- 您有 *`myssasproject`*`.asdatabase` 和 *`myssasproject`*`.deploymentoptions` 文件。在您构建 SSAS 项目时，这些文件会自动生成。

在 RDS 上部署 SSAS 项目

1. 将 .asdatabase（SSAS 模型）文件从 S3 存储桶下载到数据库实例，如以下示例所示。有关下载参数的更多信息，请参阅将文件从 Amazon S3 存储桶下载到 SQL Server 数据库实例。

```
exec msdb.dbo.rds_download_from_s3
@s3_arn_of_file='arn:aws:s3:::bucket_name/myssasproject.asdatabase',
[@rds_file_path='D:\S3\myssasproject.asdatabase'],
[@overwrite_file=1];
```

2. 将 .deploymentoptions 文件从 S3 存储桶下载到数据库实例。

```
exec msdb.dbo.rds_download_from_s3
@s3_arn_of_file='arn:aws:s3:::bucket_name/myssasproject.deploymentoptions',
[@rds_file_path='D:\S3\myssasproject.deploymentoptions'],
[@overwrite_file=1];
```

3. 部署项目。

```
exec msdb.dbo.rds_msbi_task
@task_type='SSAS_DEPLOY_PROJECT',
@file_path='D:\S3\myssasproject.asdatabase';
```

监控部署任务的状态

要跟踪部署（或下载）任务的状态，请调用 rds_fn_task_status 函数。它获取两个参数。第一个参数应该始终为 NULL，因为它不适用于 SSAS。第二个参数接受任务 ID。

要查看所有任务的列表，请将第一个参数设置为 NULL，将第二个参数设置为 0，如以下示例所示。

```
SELECT * FROM msdb.dbo.rds_fn_task_status(NULL,0);
```

要获取特定任务，请将第一个参数设置为 NULL，将第二个参数设置为任务 ID，如以下示例所示。

```
SELECT * FROM msdb.dbo.rds_fn_task_status(NULL,42);
```

rds_fn_task_status 函数将返回以下信息。

输出参数	描述
task_id	任务的 ID。
task_type	对于 SSAS，任务可以具有以下任务类型： • SSAS_DEPLOY_PROJECT • SSAS_ADD_DB_ADMIN_MEMBER • SSAS_BACKUP_DB • SSAS_RESTORE_DB
database_name	不适用于 SSAS 任务。
% complete	用百分比表示的任务进度。
duration (mins)	在任务上花费的时间 (以分钟为单位)。
lifecycle	任务的状态。有以下可能状态： • CREATED – 调用 SSAS 存储过程之一后，将创建一个任务并将状态设置为 CREATED。 • IN_PROGRESS – 任务开始后，状态将设置为 IN_PROGRESS 。要将状态从 CREATED 更改为 IN_PROGRESS 可能需要最多 5 分钟的时间。 • SUCCESS – 任务完成后，状态将设置为 SUCCESS。 • ERROR – 如果任务失败，状态将设置为 ERROR。有关错误的更多信息，请参阅 task_info 列。 • CANCEL_REQUESTED – 调用 rds_cancel_task 后，任务的状态将设置为 CANCEL_REQUESTED 。

输出参数	描述
	• `CANCELLED` – 在成功取消任务后，任务的状态将设置为 `CANCELLED` 。
`task_info`	有关任务的其他信息。如果在处理过程中发生错误，则此列包含有关错误的信息。 有关更多信息，请参阅 解决 SSAS 问题。
`last_updated`	上次更新任务状态的日期和时间。
`created_at`	任务的创建日期和时间。
`S3_object_arn`	不适用于 SSAS 任务。
`overwrite_S3_backup_file`	不适用于 SSAS 任务。
`KMS_master_key_arn`	不适用于 SSAS 任务。
`filepath`	不适用于 SSAS 任务。
`overwrite_file`	不适用于 SSAS 任务。
`task_metadata`	与 SSAS 任务关联的元数据。

在 Amazon RDS 上使用 SSAS

部署 SSAS 项目后，您可以直接在 SSMS 上处理 OLAP 数据库。

在 RDS 上使用 SSAS

1. 在 SSMS 中，使用 Active Directory 域的用户名和密码连接到 SSAS。
2. 展开 Databases (数据库)。此时将显示新部署的 SSAS 数据库。
3. 找到连接字符串，再更新用户名和密码，以便有权访问源 SQL 数据库。处理 SSAS 对象必须要执行此操作。

a. 针对表格模式执行以下操作：

   1. 展开 Connections（连接）选项卡。
   2. 打开连接对象的上下文（右键单击）菜单，然后选择 Properties（属性）。
   3. 更新连接字符串中的用户名和密码。

b. 针对多维模式执行以下操作：

   1. 展开 Data Sources（数据源）选项卡。
   2. 打开数据源对象的上下文（右键单击）菜单，然后选择 Properties（属性）。
   3. 更新连接字符串中的用户名和密码。

4. 打开您创建的 SSAS 数据库的上下文（右键单击）菜单，然后选择 Process Database (处理数据库)。

   根据输入数据的大小，处理操作可能需要几分钟时间才能完成。

主题

- 为 SSAS 设置经 Windows 身份验证的用户
- 添加域用户作为数据库管理员
- 创建 SSAS 代理
- 使用 SQL Server Agent 安排 SSAS 数据库处理
- 从代理撤消 SSAS 访问权限

为 SSAS 设置经 Windows 身份验证的用户

主管理员用户（有时也被称为主用户）可以使用以下代码示例设置经 Windows 身份验证的登录并授予所需的过程权限。执行此操作可向域用户授予权限以运行 SSAS 客户任务、使用 S3 文件传输过程、创建凭证以及使用 SQL Server Agent 代理。有关更多信息，请参阅 Microsoft 文档中的凭证（数据库引擎）和创建 SQL Server Agent 代理。

您可以根据需要向经 Windows 身份验证的用户授予以下部分或所有权限。

Example

```
-- Create a server-level domain user login, if it doesn't already exist
USE [master]
GO
```

```
CREATE LOGIN [mydomain\user_name] FROM WINDOWS
GO

-- Create domain user, if it doesn't already exist
USE [msdb]
GO
CREATE USER [mydomain\user_name] FOR LOGIN [mydomain\user_name]
GO

-- Grant necessary privileges to the domain user
USE [master]
GO
GRANT ALTER ANY CREDENTIAL TO [mydomain\user_name]
GO

USE [msdb]
GO
GRANT EXEC ON msdb.dbo.rds_msbi_task TO [mydomain\user_name] with grant option
GRANT SELECT ON msdb.dbo.rds_fn_task_status TO [mydomain\user_name] with grant option
GRANT EXEC ON msdb.dbo.rds_task_status TO [mydomain\user_name] with grant option
GRANT EXEC ON msdb.dbo.rds_cancel_task TO [mydomain\user_name] with grant option
GRANT EXEC ON msdb.dbo.rds_download_from_s3 TO [mydomain\user_name] with grant option
GRANT EXEC ON msdb.dbo.rds_upload_to_s3 TO [mydomain\user_name] with grant option
GRANT EXEC ON msdb.dbo.rds_delete_from_filesystem TO [mydomain\user_name] with grant
 option
GRANT EXEC ON msdb.dbo.rds_gather_file_details TO [mydomain\user_name] with grant
 option
GRANT EXEC ON msdb.dbo.sp_add_proxy TO [mydomain\user_name] with grant option
GRANT EXEC ON msdb.dbo.sp_update_proxy TO [mydomain\user_name] with grant option
GRANT EXEC ON msdb.dbo.sp_grant_login_to_proxy TO [mydomain\user_name] with grant
 option
GRANT EXEC ON msdb.dbo.sp_revoke_login_from_proxy TO [mydomain\user_name] with grant
 option
GRANT EXEC ON msdb.dbo.sp_delete_proxy TO [mydomain\user_name] with grant option
GRANT EXEC ON msdb.dbo.sp_enum_login_for_proxy to [mydomain\user_name] with grant
 option
GRANT EXEC ON msdb.dbo.sp_enum_proxy_for_subsystem TO [mydomain\user_name] with grant
 option
GRANT EXEC ON msdb.dbo.rds_sqlagent_proxy TO [mydomain\user_name] with grant option
ALTER ROLE [SQLAgentUserRole] ADD MEMBER [mydomain\user_name]
GO
```

添加域用户作为数据库管理员

您可以通过以下方式，将域用户添加为 SSAS 数据库管理员：

- 数据库管理员可以使用 SSMS 创建具有 admin 权限的角色，然后将用户添加到该角色。
- 您可以使用以下存储过程。

```
exec msdb.dbo.rds_msbi_task
@task_type='SSAS_ADD_DB_ADMIN_MEMBER',
@database_name='myssasdb',
@ssas_role_name='exampleRole',
@ssas_role_member='domain_name\domain_user_name';
```

  以下参数为必需参数：
  - `@task_type` – MSBI 任务的类型，在此例中为 `SSAS_ADD_DB_ADMIN_MEMBER`。
  - `@database_name` – 要向其授予管理员特权的 SSAS 数据库的名称。
  - `@ssas_role_name` – SSAS 数据库管理员角色名称。如果角色尚不存在，则会创建它。
  - `@ssas_role_member` – 要添加到管理员角色的 SSAS 数据库用户。

创建 SSAS 代理

要先创建 SSAS 凭证和 SSAS 代理，才能使用 SQL Server Agent 来安排 SSAS 数据库处理。以经 Windows 身份验证的用户身份运行这些过程。

创建 SSAS 凭证

- 为代理创建凭证。为此，您可以使用 SSMS 或以下 SQL 语句。

```
USE [master]
GO
CREATE CREDENTIAL [SSAS_Credential] WITH IDENTITY = N'mydomain\user_name', SECRET =
 N'mysecret'
GO
```

> **Note**
>
> `IDENTITY` 必须是经域身份验证的登录名。将 *`mysecret`* 替换为经域身份验证的登录的密码。

创建 SSAS 代理

1. 使用以下 SQL 语句创建代理。

```
USE [msdb]
GO
EXEC msdb.dbo.sp_add_proxy
 @proxy_name=N'SSAS_Proxy',@credential_name=N'SSAS_Credential',@description=N''
GO
```

2. 使用以下 SQL 语句向其他用户授予对代理的访问权限。

```
USE [msdb]
GO
EXEC msdb.dbo.sp_grant_login_to_proxy
 @proxy_name=N'SSAS_Proxy',@login_name=N'mydomain\user_name'
GO
```

3. 使用以下 SQL 语句向代理授予 SSAS 子系统访问权限。

```
USE [msdb]
GO
EXEC msdb.dbo.rds_sqlagent_proxy
 @task_type='GRANT_SUBSYSTEM_ACCESS',@proxy_name='SSAS_Proxy',@proxy_subsystem='SSAS'
GO
```

查看代理和代理上的授权

1. 使用以下 SQL 语句查看代理的被授权者。

```
USE [msdb]
GO
EXEC sp_help_proxy
GO
```

2. 使用以下 SQL 语句查看子系统授权。

```
USE [msdb]
GO
EXEC msdb.dbo.sp_enum_proxy_for_subsystem
GO
```

使用 SQL Server Agent 安排 SSAS 数据库处理

创建凭证和代理并向 SSAS 授予对代理的访问权限后，您可以创建 SQL Server Agent 任务来安排 SSAS 数据库处理。

安排 SSAS 数据库处理

- 使用 SSMS 或 T-SQL 创建 SQL Server Agent 任务。以下示例使用 T-SQL。您可以通过 SSMS 或 T-SQL 进一步配置其任务计划。
  - `@command` 参数概述了将由 SQL Server Agent 任务运行的 XML for Analysis (XMLA) 命令。此示例配置了 SSAS 多维数据库处理。
  - `@server` 参数概述了 SQL Server Agent 任务的目标 SSAS 服务器名称。

    要在 SQL Server Agent 任务所在的同一 RDS 数据库实例中调用 SSAS 服务，请使用 `localhost:2383`。

    要从 RDS 数据库实例外部调用 SSAS 服务，请使用 RDS 端点。如果 RDS 数据库实例是由同一个域联接，则还可以使用 Kerberos Active Directory (AD) 端点 (*`your-DB-instance-name`*.*`your-AD-domain-name`*)。对于外部数据库实例，请确保正确配置与 RDS 数据库实例关联的 VPC 安全组，以便实现安全连接。

  您可以进一步编辑查询来支持各种 XMLA 操作。可以直接修改 T-SQL 查询进行编辑，也可以在创建 SQL Server Agent 任务后使用 SSMS UI 进行编辑。

```
USE [msdb]
GO
DECLARE @jobId BINARY(16)
EXEC msdb.dbo.sp_add_job @job_name=N'SSAS_Job',
    @enabled=1,
    @notify_level_eventlog=0,
    @notify_level_email=0,
    @notify_level_netsend=0,
    @notify_level_page=0,
    @delete_level=0,
    @category_name=N'[Uncategorized (Local)]',
    @job_id = @jobId OUTPUT
GO
EXEC msdb.dbo.sp_add_jobserver
    @job_name=N'SSAS_Job',
    @server_name = N'(local)'
```

```
GO
EXEC msdb.dbo.sp_add_jobstep @job_name=N'SSAS_Job',
 @step_name=N'Process_SSAS_Object',
    @step_id=1,
    @cmdexec_success_code=0,
    @on_success_action=1,
    @on_success_step_id=0,
    @on_fail_action=2,
    @on_fail_step_id=0,
    @retry_attempts=0,
    @retry_interval=0,
    @os_run_priority=0, @subsystem=N'ANALYSISCOMMAND',
    @command=N'<Batch xmlns="http://schemas.microsoft.com/analysisservices/2003/
engine">
        <Parallel>
            <Process xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                xmlns:ddl2="http://schemas.microsoft.com/analysisservices/2003/
engine/2" xmlns:ddl2_2="http://schemas.microsoft.com/analysisservices/2003/
engine/2/2"
                xmlns:ddl100_100="http://schemas.microsoft.com/
analysisservices/2008/engine/100/100" xmlns:ddl200="http://schemas.microsoft.com/
analysisservices/2010/engine/200"
                xmlns:ddl200_200="http://schemas.microsoft.com/
analysisservices/2010/engine/200/200" xmlns:ddl300="http://schemas.microsoft.com/
analysisservices/2011/engine/300"
                xmlns:ddl300_300="http://schemas.microsoft.com/
analysisservices/2011/engine/300/300" xmlns:ddl400="http://schemas.microsoft.com/
analysisservices/2012/engine/400"
                xmlns:ddl400_400="http://schemas.microsoft.com/
analysisservices/2012/engine/400/400" xmlns:ddl500="http://schemas.microsoft.com/
analysisservices/2013/engine/500"
                xmlns:ddl500_500="http://schemas.microsoft.com/
analysisservices/2013/engine/500/500">
                <Object>
                    <DatabaseID>Your_SSAS_Database_ID</DatabaseID>
                </Object>
                <Type>ProcessFull</Type>
                <WriteBackTableCreation>UseExisting</WriteBackTableCreation>
            </Process>
        </Parallel>
    </Batch>',
    @server=N'localhost:2383',
    @database_name=N'master',
```

```
    @flags=0,
    @proxy_name=N'SSAS_Proxy'
GO
```

从代理撤消 SSAS 访问权限

您可以使用以下存储过程撤消对 SSAS 子系统的访问权限并删除 SSAS 代理。

撤消访问权限并删除代理

1. 撤消子系统访问权限。

```
USE [msdb]
GO
EXEC msdb.dbo.rds_sqlagent_proxy
 @task_type='REVOKE_SUBSYSTEM_ACCESS',@proxy_name='SSAS_Proxy',@proxy_subsystem='SSAS'
GO
```

2. 撤消代理上的授权。

```
USE [msdb]
GO
EXEC msdb.dbo.sp_revoke_login_from_proxy
 @proxy_name=N'SSAS_Proxy',@name=N'mydomain\user_name'
GO
```

3. 删除代理。

```
USE [msdb]
GO
EXEC dbo.sp_delete_proxy @proxy_name = N'SSAS_Proxy'
GO
```

备份 SSAS 数据库

您只能在数据库实例的 `D:\S3` 文件夹中创建 SSAS 数据库备份文件。要将备份文件移动到 S3 存储桶，请使用 Amazon S3。

您可以如下所示备份 SSAS 数据库：

- 特定数据库具有 `admin` 角色的域用户可以使用 SSMS 将数据库备份到 `D:\S3` 文件夹。

有关更多信息，请参阅“添加域用户作为数据库管理员”。

- 您可以使用以下存储过程。此存储过程不支持加密。

```
exec msdb.dbo.rds_msbi_task
@task_type='SSAS_BACKUP_DB',
@database_name='myssasdb',
@file_path='D:\S3\ssas_db_backup.abf',
[@ssas_apply_compression=1],
[@ssas_overwrite_file=1];
```

以下参数为必需参数：

- `@task_type` – MSBI 任务的类型，在此例中为 `SSAS_BACKUP_DB`。
- `@database_name` – 要备份的 SSAS 数据库的名称。
- `@file_path` – SSAS 备份文件的路径。`.abf` 扩展名必需。

以下参数为可选参数：

- `@ssas_apply_compression` – 是否应用 SSAS 备份压缩。有效值为 1（是）和 0（否）。
- `@ssas_overwrite_file` – 是否覆盖 SSAS 备份文件。有效值为 1（是）和 0（否）。

还原 SSAS 数据库

使用以下存储过程从备份还原 SSAS 数据库。

如果存在同名的现有 SSAS 数据库，则无法还原数据库。用于还原的存储过程不支持加密的备份文件。

```
exec msdb.dbo.rds_msbi_task
@task_type='SSAS_RESTORE_DB',
@database_name='mynewssasdb',
@file_path='D:\S3\ssas_db_backup.abf';
```

以下参数为必需参数：

- `@task_type` – MSBI 任务的类型，在此例中为 `SSAS_RESTORE_DB`。
- `@database_name` – 您要还原到的新 SSAS 数据库的名称。
- `@file_path` – SSAS 备份文件的路径。

将数据库实例还原到指定时间

时间点恢复 (PITR) 不适用于 SSAS 数据库。如果您执行 PITR，则在还原的实例上，只有在请求时间之前的最后一个快照中的 SSAS 数据可用。

在还原的数据库实例上拥有最新的 SSAS 数据库

1. 将您的 SSAS 数据库备份到源实例上的 `D:\S3` 文件夹。
2. 将备份文件传输到 S3 存储桶。
3. 将备份文件从 S3 存储桶传输到还原后的实例上的 `D:\S3` 文件夹。
4. 运行存储过程，将 SSAS 数据库还原到还原后的实例上。

   您还可以重新处理 SSAS 项目以还原数据库。

更改 SSAS 模式

您可以更改 SSAS 的运行模式，无论其采用的是表格模式还是多维模式。若要更改模式，请使用 Amazon Web Services Management Console 或 Amazon CLI 修改 SSAS 选项中的选项设置。

> **Important**
>
> 一次只能使用一个 SSAS 模式。更改模式之前，请务必删除所有 SSAS 数据库，否则会收到错误消息。

控制台

以下 Amazon RDS 控制台程序可将 SSAS 模式更改为表格模式，然后将 `MAX_MEMORY` 参数设置为 70%。

修改 SSAS 选项

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择选项组。
3. 选择具有要修改的 SSAS 选项（前面示例中的 `ssas-se-2017`）的选项组。
4. 选择 Modify option（修改选项）。
5. 更改选项设置：

a. 在 Max memory（最大内存）中输入 **70**。

b. 在Mode（模式）中选择 Tabular（表格）。

6. 选择 Modify option（修改选项）。

Amazon CLI

以下 Amazon CLI 示例会将 SSAS 模式更改为表格模式，然后将 `MAX_MEMORY` 参数设置为 70%。

为了使 CLI 命令正常运行，请确保命令中包含所有必要参数，即使未对其进行修改。

修改 SSAS 选项

- 使用以下命令之一。

  Example

  对于 Linux、macOS 或 Unix：

```
aws rds add-option-to-option-group \
    --option-group-name ssas-se-2017 \
    --options
 "OptionName=SSAS,VpcSecurityGroupMemberships=sg-12345e67,OptionSettings=[{Name=MAX_MEMORY,
{Name=MODE,Value=Tabular}]" \
    --apply-immediately
```

  对于 Windows：

```
aws rds add-option-to-option-group ^
    --option-group-name ssas-se-2017 ^
    --options
 OptionName=SSAS,VpcSecurityGroupMemberships=sg-12345e67,OptionSettings=[{Name=MAX_MEMORY,V
{Name=MODE,Value=Tabular}] ^
    --apply-immediately
```

禁用 SSAS

若要禁用 SSAS，请从选项组中删除 SSAS 选项。

> **Important**
>
> 删除 SSAS 选项之前，请先删除您的 SSAS 数据库。
>
> 我们强烈建议您先备份 SSAS 数据库，然后再删除这些数据库并删除 SSAS 选项。

控制台

从其选项组中删除 SSAS 选项

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择选项组。
3. 选择具有要删除的 SSAS 选项（前面示例中的 `ssas-se-2017`）的选项组。
4. 选择 Delete option (删除选项)。
5. 在 Delete option (删除选项) 下，为 Options to delete (要删除的选项) 选择 SSAS。
6. 在 Apply immediately (立即应用) 下，选择 Yes (是) 可立即删除选项，选择 No (否) 可在下次维护时段时删除。
7. 选择删除。

Amazon CLI

从其选项组中删除 SSAS 选项

- 使用以下命令之一。

  Example

  对于 Linux、macOS 或 Unix：

  ```
  aws rds remove-option-from-option-group \
      --option-group-name ssas-se-2017 \
      --options SSAS \
      --apply-immediately
  ```

  对于 Windows：

  ```
  aws rds remove-option-from-option-group ^
  ```

```
    --option-group-name ssas-se-2017 ^
    --options SSAS ^
    --apply-immediately
```

解决 SSAS 问题

您可能会在使用 SSAS 时遇到以下问题。

问题	类型	故障排除建议
无法配置 SSAS 选项。请求的 SSAS 模式是 *new_mode*，但当前数据库实例有 *number* 个 *current_mode* 数据库。在切换到 *new_mode* 模式之前，请删除现有数据库。要重新获得数据库删除 *current_mode* 模式的访问权限，可以更新当前数据库选项组，也可以附加新选项组（将 %s 作为 SSAS 选项的模式选项设置值）。	RDS 事件	如果仍有 SSAS 数据库使用当前模式，则无法更改 SSAS 模式。请删除 SSAS 数据库，然后重试。
无法删除 SSAS 选项，因为存在 *number* 个现有 *mode* 数据库。删除所有 SSAS 数据库后，才能删除 SSAS 选项。再次添加 SSAS 选项，删除所有 SSAS 数据库，然后重试。	RDS 事件	如果仍然有 SSAS 数据库，则无法禁用 SSAS。请删除 SSAS 数据库，然后重试。
SSAS 选项未启用或处于启用过程中。请稍后重试。	RDS 存储过程	选项被禁用或正在启用时，您无法运行 SSAS 存储过程。
SSAS 选项配置不正确。确保选项组成员资格状态为“同步中”，然后查看 RDS 事件日志以获取相关 SSAS 配置错误消息。在完成这些调查后重试一次。如果仍会出现错误，请联系 Amazon Support。	RDS 存储过程	如果选项组成员资格不是 `in-sync` 状态，则无法运行 SSAS 存储过程。这会让 SSAS 选项处于不正确的配置状态。 如果选项组成员资格状态因 SSAS 选项修改而变为 `failed`，则有两个可能原因： 1. 已删除 SSAS 选项，但未删除 SSAS 数据库。

问题	类型	故障排除建议
		2. SSAS 模式已从表格模式更新为多维模式或从多维模式更新为表格模式，但未删除现有的 SSAS 数据库。 重新配置 SSAS 选项，因为 RDS 一次只允许一个 SSAS 模式，并且不支持在存在 SSAS 数据库的情况下删除 SSAS 选项。 检查 RDS 事件日志中 SSAS 实例的配置错误，再相应地解决问题。
部署失败。只能在以 *deployment_file_mode* 模式运行的服务器上部署更改。当前服务器模式为 *current_mode* 。	RDS 存储过程	无法将表格数据库部署到多维服务器，也无法将多维数据库部署到表格服务器。 确保使用的是具有正确模式的文件，然后验证 `MODE` 选项设置是否设置为合适的值。
还原失败。只能在以 *restore_file_mode* 模式运行的服务器上恢复备份文件。当前服务器模式为 *current_mode* 。	RDS 存储过程	无法将表格数据库还原到多维服务器，也无法将多维数据库还原到表格服务器。 确保使用的是具有正确模式的文件，然后验证 `MODE` 选项设置是否设置为合适的值。
还原失败。备份文件与 RDS 数据库实例版本不兼容。	RDS 存储过程	无法使用与 SQL Server 实例版本不兼容的版本还原 SSAS 数据库。 有关更多信息，请参阅 Microsoft 文档中的[表格模型的兼容性级别]和[多维数据库的兼容性级别]。
还原失败。还原操作中指定的备份文件已损坏或不是 SSAS 备份文件。确保 @rds_file_path 格式正确。	RDS 存储过程	无法使用受损文件还原 SSAS 数据库。 确保文件未遭破坏或损坏。 此错误也会在 `@rds_file_path` 格式不正确时发生。（例如 `D:\S3\\incorrect_format.abf` 中存在两条反斜杠）。

<table>
<tr><th>问题</th><th>类型</th><th>故障排除建议</th></tr>
<tr><td>还原失败。还原的数据库名称不能包含任何保留字或无效字符（. , ; ' ` : / \\ * | ? \" & % $! + = ( ) [ ] { } <>），长度不能超过 100 个字符。</td><td>RDS 存储过程</td><td>还原的数据库名称不能包含任何保留字或无效字符，长度不能超过 100 个字符。

有关 SSAS 对象命名惯例，请参阅 Microsoft 文档中的对象命名规则。</td></tr>
<tr><td>提供的角色名称无效。角色名称不能包含任何保留字符串。</td><td>RDS 存储过程</td><td>角色名称不能包含任何保留字符串。

有关 SSAS 对象命名惯例，请参阅 Microsoft 文档中的对象命名规则。</td></tr>
<tr><td>提供的角色名称无效。角色名称不能包含以下任何保留字符：. , ; ' ` : / \\ * | ? \" & % $! + = ( ) [ ] { } <></td><td>RDS 存储过程</td><td>角色名称不能包含任何保留字符。

有关 SSAS 对象命名惯例，请参阅 Microsoft 文档中的对象命名规则。</td></tr>
</table>

支持 Amazon RDS for SQL Server 中的 SQL Server Integration Services

Microsoft SQL Server Integration Services (SSIS) 是一个组件，可用于执行一系列广泛的数据迁移任务。SSIS 是一个用于数据集成和工作流应用程序的平台。它具有用于数据提取、转换和加载 (ETL) 的数据仓库工具。您还可以使用此工具自动维护 SQL Server 数据库以及对多维数据集数据的更新。

SSIS 项目将整理到另存为基于 XML 的 .dtsx 文件的包中。包可以包含控制流和数据流。您可以使用数据流来表示 ETL 操作。部署后，包存储在 SQL Server 中的 SSISDB 数据库中。SSISDB 是处于完全恢复模式的联机事务处理 (OLTP) 数据库。

Amazon RDS for SQL Server 支持直接在 RDS 数据库实例上运行 SSIS。您可以在现有的或新的数据库实例上启用 SSIS。SSIS 安装在与您的数据库引擎相同的数据库实例上。

RDS 在以下版本上对于 SQL Server 标准版和企业版支持 SSIS：

- SQL Server 2022，所有版本
- SQL Server 2019，版本 15.00.4043.16.v1 和更高版本
- SQL Server 2017，版本 14.00.3223.3.v1 和更高版本
- SQL Server 2016，版本 13.00.5426.0.v1 和更高版本

目录

  - 为 SSIS 项目设置数据库连接管理器
  - 创建 SSIS 代理
  - 使用 SQL Server Agent 安排 SSIS 程序包
  - 从代理撤消 SSIS 访问权限
- 禁用并删除 SSIS 数据库
  - 禁用 SSIS
  - 删除 SSISDB 数据库

限制和建议

以下限制和建议适用于在 RDS for SQL Server 上运行 SSIS：

- 数据库实例必须具有 `clr enabled` 参数设置为 1 的关联参数组。有关更多信息，请参阅 修改 SSIS 的参数。

> **Note**
>
> 如果在 SQL Server 2017 或 2019 上启用 `clr enabled` 参数，则无法在数据库实例上使用公共语言运行时 (CLR)。有关更多信息，请参阅 不支持的功能以及有限支持的功能。

- 支持以下控制流任务：
  - 分析服务执行 DDL 任务
  - 分析服务处理任务
  - 批量插入任务
  - 检查数据库完整性任务
  - 数据流任务
  - 数据挖掘查询任务
  - 数据分析任务
  - 执行包任务
  - 执行 SQL Server Agent 作业任务
  - 执行 SQL 任务
  - 执行 T-SQL 语句任务
  - 通知操作员任务

  - 重建索引任务
  - 重新组织索引任务
  - 收缩数据库任务
  - 传输数据库任务
  - 传输作业任务
  - 传输登录名任务
  - 传输 SQL Server 对象任务
  - 更新统计数据任务
- 仅支持项目部署。
- 支持使用 SQL Server Agent 运行 SSIS 程序包。
- SSIS 日志记录只能插入用户创建的数据库中。
- 仅使用 `D:\S3` 文件夹来处理文件。放置在任何其他目录中的文件将被删除。请注意其他几个文件位置的详细信息：
  - 将 SSIS 项目输入和输出文件置于 `D:\S3` 文件夹中。
  - 对于数据流任务，将 `BLOBTempStoragePath` 和 `BufferTempStoragePath` 的位置更改为 `D:\S3` 文件夹中的文件。文件路径必须以 `D:\S3\` 开头。
  - 确保用于文件连接的所有参数、变量和表达式都指向 `D:\S3` 文件夹。
  - 在多可用区实例上，SSIS 在 `D:\S3` 文件夹中创建的文件将在故障转移后被删除。有关更多信息，请参阅"S3 集成的多可用区限制"。
  - 将 SSIS 在 `D:\S3` 文件夹中创建的文件上传到您的 Amazon S3 存储桶以使其持久存在。
- 不支持导入列和导出列转换以及数据流任务上的脚本组件。
- 无法对正在运行的 SSIS 程序包启用转储，也无法在 SSIS 程序包上添加数据分接头。
- 不支持 SSIS 扩展功能。
- 您不能直接部署项目。我们提供 RDS 存储过程来执行此操作。有关更多信息，请参阅"部署 SSIS 项目"。
- 使用 `DoNotSavePasswords` 保护模式构建 SSIS 项目 (.ispac) 文件以便在 RDS 上进行部署。
- 具有只读副本的"始终打开"实例不支持 SSIS。
- 您无法备份与 `SSIS` 选项关联的 SSISDB 数据库。
- 不支持从 SSIS 的其他实例导入和还原 SSISDB 数据库。

- 您可以连接到其他 SQL Server 数据库实例或 Oracle 数据来源。RDS for SQL Server 上的 SSIS 不支持连接到其他数据库引擎（例如 MySQL 或 PostgreSQL）。有关连接到 Oracle 数据来源的更多信息，请参阅 带有 Oracle OLEDB 的链接服务器。
- SSIS 不支持与本地域建立传出信任的已加入域的实例。在使用传出信任时，请从本地 Amazon 域中的账户运行 SSIS 作业。

启用 SSIS

您可以通过向数据库实例添加 SSIS 选项来启用 SSIS。使用以下过程：

1. 创建新的选项组或选择现有的选项组。
2. 将 `SSIS` 选项添加到该选项组。
3. 创建新的参数组，或选择现有参数组。
4. 修改参数组以将 `clr enabled` 参数设置为 1。
5. 将选项组和参数组与数据库实例关联。
6. 启用 Amazon S3 集成。

> **Note**
> 如果数据库实例上已存在带有名称 SSISDB 或预留的 SSIS 登录名的数据库，则无法在实例上启用 SSIS。

为 SSIS 创建选项组

要使用 SSIS，请创建或修改与您计划使用的 SQL Server 版本和数据库实例版本对应的选项组。为此，请使用 Amazon Web Services Management Console或 Amazon CLI。

控制台

以下控制台过程创建适用于 SQL Server 标准版 2016 的选项组。

创建选项组

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择选项组。

3. 选择创建群组。
4. 在创建选项组窗口中，执行以下操作：
   a. 对于名称，输入选项组的名称，该名称在 Amazon 账户内具有唯一性，例如 **ssis-se-2016**。此名称只能包含字母、数字和连字符。
   b. 对于描述，输入选项组的简要描述，例如 **SSIS option group for SQL Server SE 2016**。此说明用于显示说明。
   c. 对于引擎，选择 sqlserver-se。
   d. 对于主要引擎版本，选择 13.00。
5. 选择创建。

CLI

以下控制台过程创建适用于 SQL Server 标准版 2016 的选项组。

创建选项组

- 运行以下命令之一。

  Example

  对于 Linux、macOS 或 Unix：

```
aws rds create-option-group \
    --option-group-name ssis-se-2016 \
    --engine-name sqlserver-se \
    --major-engine-version 13.00 \
    --option-group-description "SSIS option group for SQL Server SE 2016"
```

  对于 Windows：

```
aws rds create-option-group ^
    --option-group-name ssis-se-2016 ^
    --engine-name sqlserver-se ^
    --major-engine-version 13.00 ^
    --option-group-description "SSIS option group for SQL Server SE 2016"
```

将 SSIS 选项添加到选项组

接下来，使用 Amazon Web Services Management Console或 Amazon CLI 将 SSIS 选项添加到您的选项组。

控制台

添加 SSIS 选项

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择选项组。
3. 选择您刚刚创建的选项组（在此示例中为 ssis-se-2016）。
4. 选择添加选项。
5. 在选项详细信息下，为选项名称选择 SSIS。
6. 在 计划 (Scheduling) 下，选择是立即添加选项还是在下一个维护时段添加选项。
7. 选择添加选项。

CLI

添加 SSIS 选项

- 将 SSIS 选项添加到该选项组。

  Example

  对于 Linux、macOS 或 Unix：

```
aws rds add-option-to-option-group \
    --option-group-name ssis-se-2016 \
    --options OptionName=SSIS \
    --apply-immediately
```

  对于 Windows：

```
aws rds add-option-to-option-group ^
    --option-group-name ssis-se-2016 ^
    --options OptionName=SSIS ^
    --apply-immediately
```

为 SSIS 创建参数组

为与您计划用于 SSIS 的 SQL Server 版本和数据库实例版本对应的 `clr enabled` 参数创建或修改参数组。

控制台

以下过程为 SQL Server 标准版 2016 创建一个参数组。

创建参数组

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择参数组。
3. 选择创建参数组。
4. 在创建参数组窗格中，执行以下操作：
   a. 对于参数组系列，选择 sqlserver-se-13.0。
   b. 对于组名称，输入参数组的标识符，如 **ssis-sqlserver-se-13**。
   c. 对于描述，输入 **clr enabled parameter group**。
5. 选择创建。

CLI

以下过程为 SQL Server 标准版 2016 创建一个参数组。

创建参数组

- 运行以下命令之一。

  Example

  对于 Linux、macOS 或 Unix：

  ```
  aws rds create-db-parameter-group \
      --db-parameter-group-name ssis-sqlserver-se-13 \
      --db-parameter-group-family "sqlserver-se-13.0" \
      --description "clr enabled parameter group"
  ```

  对于 Windows：

```
aws rds create-db-parameter-group ^
    --db-parameter-group-name ssis-sqlserver-se-13 ^
    --db-parameter-group-family "sqlserver-se-13.0" ^
    --description "clr enabled parameter group"
```

修改 SSIS 的参数

修改与 SQL Server 版本和数据库实例版本对应的参数组中的 `clr enabled` 参数。对于 SSIS，将 `clr enabled` 参数设置为 1。

控制台

以下过程修改您为 SQL Server 标准版 2016 创建的参数组。

修改参数组

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择参数组。
3. 选择参数组，例如 ssis-sqlserver-se-13。
4. 在参数下，从参数列表中筛选 **`clr`**。
5. 选择 clr enabled (启用 clr)。
6. 选择编辑参数。
7. 从值中选择 1。
8. 选择保存更改。

CLI

以下过程修改您为 SQL Server 标准版 2016 创建的参数组。

修改参数组

- 运行以下命令之一。

  Example

  对于 Linux、macOS 或 Unix：

```
aws rds modify-db-parameter-group \
    --db-parameter-group-name ssis-sqlserver-se-13 \
    --parameters "ParameterName='clr
 enabled',ParameterValue=1,ApplyMethod=immediate"
```

对于 Windows：

```
aws rds modify-db-parameter-group ^
    --db-parameter-group-name ssis-sqlserver-se-13 ^
    --parameters "ParameterName='clr
 enabled',ParameterValue=1,ApplyMethod=immediate"
```

将选项组和参数组与数据库实例关联

要将 SSIS 选项组和参数组与数据库实例关联，请使用 Amazon Web Services Management Console 或 Amazon CLI

> **Note**
> 如果您使用现有实例，则该实例必须已关联 Active Directory 域和 Amazon Identity and Access Management (IAM) 角色。如果您创建新实例，请指定现有的 Active Directory 域和 IAM 角色。有关更多信息，请参阅 将 Active Directory 用于 RDS for SQL Server。

控制台

要完成对 SSIS 的启用，请将 SSIS 选项组和参数组与新的或现有的数据库实例关联：

- 对于新数据库实例，请在启动实例时将其关联。有关更多信息，请参阅"创建 Amazon RDS 数据库实例"。
- 对于现有数据库实例，请通过修改实例将其关联。有关更多信息，请参阅 修改 Amazon RDS 数据库实例。

CLI

您可以将 SSIS 选项组和参数组与新的或现有的数据库实例关联。

使用 SSIS 选项组和参数组创建实例

- 指定创建选项组时使用的相同数据库引擎类型和主要版本。

  Example

  对于 Linux、macOS 或 Unix：

```
aws rds create-db-instance \
    --db-instance-identifier myssisinstance \
    --db-instance-class db.m5.2xlarge \
    --engine sqlserver-se \
    --engine-version 13.00.5426.0.v1 \
    --allocated-storage 100 \
    --manage-master-user-password \
    --master-username admin \
    --storage-type gp2 \
    --license-model li \
    --domain-iam-role-name my-directory-iam-role \
    --domain my-domain-id \
    --option-group-name ssis-se-2016 \
    --db-parameter-group-name ssis-sqlserver-se-13
```

  对于 Windows：

```
aws rds create-db-instance ^
    --db-instance-identifier myssisinstance ^
    --db-instance-class db.m5.2xlarge ^
    --engine sqlserver-se ^
    --engine-version 13.00.5426.0.v1 ^
    --allocated-storage 100 ^
    --manage-master-user-password ^
    --master-username admin ^
    --storage-type gp2 ^
    --license-model li ^
    --domain-iam-role-name my-directory-iam-role ^
    --domain my-domain-id ^
    --option-group-name ssis-se-2016 ^
    --db-parameter-group-name ssis-sqlserver-se-13
```

修改实例并关联 SSIS 选项组和参数组

- 运行以下命令之一。

  Example

  对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier myssisinstance \
    --option-group-name ssis-se-2016 \
    --db-parameter-group-name ssis-sqlserver-se-13 \
    --apply-immediately
```

  对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier myssisinstance ^
    --option-group-name ssis-se-2016 ^
    --db-parameter-group-name ssis-sqlserver-se-13 ^
    --apply-immediately
```

启用 S3 集成

要将 SSIS 项目 (.ispac) 文件下载到主机以进行部署，请使用 S3 文件集成。有关更多信息，请参阅 将 Amazon RDS for SQL Server 数据库实例与 Amazon S3 集成。

SSISDB 上的管理权限

如果使用 SSIS 选项创建或修改实例，则结果是一个 SSISDB 数据库，其中包含向主用户授予的 ssis_admin 和 ssis_logreader 角色。在 SSISDB 中，主用户有权执行以下操作：

- 更改 ssis_admin 角色
- 更改 ssis_logreader 角色
- 更改任何用户

由于主用户是经 SQL 身份验证的用户，因此，您不能使用主用户执行 SSIS 程序包。主用户可以使用这些权限创建新的 SSISDB 用户并将这些用户添加到 ssis_admin 和 ssis_logreader 角色。这样做对于向域用户授予访问权限以使用 SSIS 非常有用。

为 SSIS 设置经 Windows 身份验证的用户

主用户可以使用以下代码示例在 SSISDB 中设置经 Windows 身份验证的登录并授予所需的过程权限。执行此操作可向域用户授予权限以部署和运行 SSIS 程序包、使用 S3 文件传输过程、创建凭证以及使用 SQL Server Agent 代理。有关更多信息，请参阅 Microsoft 文档中的凭证（数据库引擎）和创建 SQL Server Agent 代理。

> **Note**
>
> 您可以根据需要向经 Windows 身份验证的用户授予以下部分或所有权限。

Example

```
-- Create a server-level SQL login for the domain user, if it doesn't already exist
USE [master]
GO
CREATE LOGIN [mydomain\user_name] FROM WINDOWS
GO

-- Create a database-level account for the domain user, if it doesn't already exist

USE [SSISDB]
GO
CREATE USER [mydomain\user_name] FOR LOGIN [mydomain\user_name]

-- Add SSIS role membership to the domain user
ALTER ROLE [ssis_admin] ADD MEMBER [mydomain\user_name]
ALTER ROLE [ssis_logreader] ADD MEMBER [mydomain\user_name]
GO

-- Add MSDB role membership to the domain user
USE [msdb]
GO
CREATE USER [mydomain\user_name] FOR LOGIN [mydomain\user_name]

-- Grant MSDB stored procedure privileges to the domain user
GRANT EXEC ON msdb.dbo.rds_msbi_task TO [mydomain\user_name] with grant option
GRANT SELECT ON msdb.dbo.rds_fn_task_status TO [mydomain\user_name] with grant option
GRANT EXEC ON msdb.dbo.rds_task_status TO [mydomain\user_name] with grant option
GRANT EXEC ON msdb.dbo.rds_cancel_task TO [mydomain\user_name] with grant option
GRANT EXEC ON msdb.dbo.rds_download_from_s3 TO [mydomain\user_name] with grant option
```

```
GRANT EXEC ON msdb.dbo.rds_upload_to_s3 TO [mydomain\user_name] with grant option
GRANT EXEC ON msdb.dbo.rds_delete_from_filesystem TO [mydomain\user_name] with grant
 option
GRANT EXEC ON msdb.dbo.rds_gather_file_details TO [mydomain\user_name] with grant
 option
GRANT EXEC ON msdb.dbo.sp_add_proxy TO [mydomain\user_name] with grant option
GRANT EXEC ON msdb.dbo.sp_update_proxy TO [mydomain\user_name] with grant option
GRANT EXEC ON msdb.dbo.sp_grant_login_to_proxy TO [mydomain\user_name] with grant
 option
GRANT EXEC ON msdb.dbo.sp_revoke_login_from_proxy TO [mydomain\user_name] with grant
 option
GRANT EXEC ON msdb.dbo.sp_delete_proxy TO [mydomain\user_name] with grant option
GRANT EXEC ON msdb.dbo.sp_enum_login_for_proxy to [mydomain\user_name] with grant
 option
GRANT EXEC ON msdb.dbo.sp_enum_proxy_for_subsystem TO [mydomain\user_name]  with grant
 option
GRANT EXEC ON msdb.dbo.rds_sqlagent_proxy TO [mydomain\user_name] WITH GRANT OPTION


-- Add the SQLAgentUserRole privilege to the domain user
USE [msdb]
GO
ALTER ROLE [SQLAgentUserRole] ADD MEMBER [mydomain\user_name]
GO

-- Grant the ALTER ANY CREDENTIAL privilege to the domain user
USE [master]
GO
GRANT ALTER ANY CREDENTIAL TO [mydomain\user_name]
GO
```

部署 SSIS 项目

在 RDS 上，您不能使用 SQL Server Management Studio (SSMS) 或 SSIS 过程直接部署 SSIS 项目。要从 Amazon S3 中下载项目文件并在随后进行部署，请使用 RDS 存储过程。

要运行存储过程，请以您已向其授予运行存储过程的权限的任何用户身份登录。有关更多信息，请参阅“为 SSIS 设置经 Windows 身份验证的用户”。

部署 SSIS 项目

1. 下载项目 (.ispac) 文件。

```
exec msdb.dbo.rds_download_from_s3
@s3_arn_of_file='arn:aws:s3:::bucket_name/ssisproject.ispac',
@rds_file_path='D:\S3\ssisproject.ispac',
@overwrite_file=1;
```

2. 提交部署任务并确保：

   - 该文件夹存在于 SSIS 目录中。
   - 该项目名称与您在开发 SSIS 项目时使用的项目名称匹配。

```
exec msdb.dbo.rds_msbi_task
@task_type='SSIS_DEPLOY_PROJECT',
@folder_name='DEMO',
@project_name='ssisproject',
@file_path='D:\S3\ssisproject.ispac';
```

监控部署任务的状态

要跟踪部署任务的状态，请调用 `rds_fn_task_status` 函数。它获取两个参数。第一个参数应该始终为 NULL，因为它不适用于 SSIS。第二个参数接受任务 ID。

要查看所有任务的列表，请将第一个参数设置为 NULL，将第二个参数设置为 `0`，如以下示例所示。

```
SELECT * FROM msdb.dbo.rds_fn_task_status(NULL,0);
```

要获取特定任务，请将第一个参数设置为 NULL，将第二个参数设置为任务 ID，如以下示例所示。

```
SELECT * FROM msdb.dbo.rds_fn_task_status(NULL,42);
```

`rds_fn_task_status` 函数将返回以下信息。

输出参数	描述
`task_id`	任务的 ID。
`task_type`	`SSIS_DEPLOY_PROJECT`
`database_name`	不适用于 SSIS 任务。

输出参数	描述
% complete	用百分比表示的任务进度。
duration (mins)	在任务上花费的时间 (以分钟为单位)。
lifecycle	任务的状态。有以下可能状态： • CREATED – 调用 msdb.dbo.rds_msbi_task 存储过程后，将创建一个任务并将状态设置为 CREATED。 • IN_PROGRESS – 任务开始后，状态将设置为 IN_PROGRESS 。要将状态从 CREATED 更改为 IN_PROGRESS 可能需要最多 5 分钟的时间。 • SUCCESS – 任务完成后，状态将设置为 SUCCESS。 • ERROR – 如果任务失败，状态将设置为 ERROR。有关错误的更多信息，请参阅 task_info 列。 • CANCEL_REQUESTED – 调用 rds_cancel_task 后，任务的状态将设置为 CANCEL_REQUESTED 。 • CANCELLED – 在成功取消任务后，任务的状态将设置为 CANCELLED 。
task_info	有关任务的其他信息。如果在处理过程中发生错误，则此列包含有关错误的信息。
last_updated	上次更新任务状态的日期和时间。
created_at	任务的创建日期和时间。

输出参数	描述
S3_object_arn	不适用于 SSIS 任务。
overwrite_S3_backup_file	不适用于 SSIS 任务。
KMS_master_key_arn	不适用于 SSIS 任务。
filepath	不适用于 SSIS 任务。
overwrite_file	不适用于 SSIS 任务。
task_metadata	与 SSIS 任务关联的元数据。

使用 SSIS

将 SSIS 项目部署到 SSIS 目录中后，您可以直接从 SSMS 运行程序包或使用 SQL Server Agent 安排这些程序包。您必须使用经 Windows 身份验证的登录名来执行 SSIS 程序包。有关更多信息，请参阅“为 SSIS 设置经 Windows 身份验证的用户”。

主题

- 为 SSIS 项目设置数据库连接管理器
- 创建 SSIS 代理
- 使用 SQL Server Agent 安排 SSIS 程序包
- 从代理撤消 SSIS 访问权限

为 SSIS 项目设置数据库连接管理器

在使用连接管理器时，可以使用以下类型的身份验证：

- 对于使用 Amazon Managed Active Directory 的本地数据库连接，您可以使用 SQL 身份验证或 Windows 身份验证。对于 Windows 身份验证，请使用 *DB_instance_name*.*fully_qualified_domain_name* 作为连接字符串的服务器名称。

以 `myssisinstance.corp-ad.example.com` 为例，其中 `myssisinstance` 是数据库实例名称，`corp-ad.example.com` 是完全限定域名。

- 对于远程连接，请始终使用 SQL 身份验证。
- 对于使用自托管式 Active Directory 的本地数据库连接，您可以使用 SQL 身份验证或 Windows 身份验证。对于 Windows 身份验证，请使用 *.* 或 *LocalHost* 作为连接字符串的服务器名称。

创建 SSIS 代理

您需要创建 SSIS 凭证和 SSIS 代理，才能使用 SQL Server Agent 安排 SSIS 程序包。以经 Windows 身份验证的用户身份运行这些过程。

创建 SSIS 凭证

- 为代理创建凭证。为此，您可以使用 SSMS 或以下 SQL 语句。

```
USE [master]
GO
CREATE CREDENTIAL [SSIS_Credential] WITH IDENTITY = N'mydomain\user_name', SECRET =
 N'mysecret'
GO
```

> **Note**
>
> IDENTITY 必须是经域身份验证的登录名。将 *mysecret* 替换为经域身份验证的登录的密码。
> 每当更改 SSISDB 主机时，请更改 SSIS 代理凭证以允许新主机访问它们。

创建 SSIS 代理

1. 使用以下 SQL 语句创建代理。

```
USE [msdb]
GO
EXEC msdb.dbo.sp_add_proxy
 @proxy_name=N'SSIS_Proxy',@credential_name=N'SSIS_Credential',@description=N''
GO
```

2. 使用以下 SQL 语句向其他用户授予对代理的访问权限。

```
USE [msdb]
GO
EXEC msdb.dbo.sp_grant_login_to_proxy
 @proxy_name=N'SSIS_Proxy',@login_name=N'mydomain\user_name'
GO
```

3. 使用以下 SQL 语句向代理授予 SSIS 子系统访问权限。

```
USE [msdb]
GO
EXEC msdb.dbo.rds_sqlagent_proxy
 @task_type='GRANT_SUBSYSTEM_ACCESS',@proxy_name='SSIS_Proxy',@proxy_subsystem='SSIS'
GO
```

查看代理和代理上的授权

1. 使用以下 SQL 语句查看代理的被授权者。

```
USE [msdb]
GO
EXEC sp_help_proxy
GO
```

2. 使用以下 SQL 语句查看子系统授权。

```
USE [msdb]
GO
EXEC msdb.dbo.sp_enum_proxy_for_subsystem
GO
```

使用 SQL Server Agent 安排 SSIS 程序包

创建凭证和代理并向 SSIS 授予对代理的访问权限后，您可以创建 SQL Server Agent 作业来安排 SSIS 程序包。

安排 SSIS 程序包

- 您可以使用 SSMS 或 T-SQL 创建 SQL Server Agent 作业。以下示例使用 T-SQL。

```
USE [msdb]
GO
DECLARE @jobId BINARY(16)
EXEC msdb.dbo.sp_add_job @job_name=N'MYSSISJob',
@enabled=1,
@notify_level_eventlog=0,
@notify_level_email=2,
@notify_level_page=2,
@delete_level=0,
@category_name=N'[Uncategorized (Local)]',
@job_id = @jobId OUTPUT
GO
EXEC msdb.dbo.sp_add_jobserver @job_name=N'MYSSISJob',@server_name=N'(local)'
GO
EXEC msdb.dbo.sp_add_jobstep
 @job_name=N'MYSSISJob',@step_name=N'ExecuteSSISPackage',
@step_id=1,
@cmdexec_success_code=0,
@on_success_action=1,
@on_fail_action=2,
@retry_attempts=0,
@retry_interval=0,
@os_run_priority=0,
@subsystem=N'SSIS',
@command=N'/ISSERVER "\"\SSISDB\MySSISFolder\MySSISProject\MySSISPackage.dtsx\"" /
SERVER "\"my-rds-ssis-instance.corp-ad.company.com/\""
/Par "\"$ServerOption::LOGGING_LEVEL(Int16)\"";1 /Par
 "\"$ServerOption::SYNCHRONIZED(Boolean)\"";True /CALLERINFO SQLAGENT /REPORTING
 E',
@database_name=N'master',
@flags=0,
@proxy_name=N'SSIS_Proxy'
GO
```

从代理撤消 SSIS 访问权限

您可以使用以下存储过程撤消对 SSIS 子系统的访问权限并删除 SSIS 代理。

撤消访问权限并删除代理

1. 撤消子系统访问权限。

```
USE [msdb]
GO
EXEC msdb.dbo.rds_sqlagent_proxy
 @task_type='REVOKE_SUBSYSTEM_ACCESS',@proxy_name='SSIS_Proxy',@proxy_subsystem='SSIS'
GO
```

2. 撤消代理上的授权。

```
USE [msdb]
GO
EXEC msdb.dbo.sp_revoke_login_from_proxy
 @proxy_name=N'SSIS_Proxy',@name=N'mydomain\user_name'
GO
```

3. 删除代理。

```
USE [msdb]
GO
EXEC dbo.sp_delete_proxy @proxy_name = N'SSIS_Proxy'
GO
```

禁用并删除 SSIS 数据库

使用以下步骤禁用或删除 SSIS 数据库：

主题

- 禁用 SSIS
- 删除 SSISDB 数据库

禁用 SSIS

要禁用 SSIS，请从其选项组中删除 SSIS 选项。

Important

删除该选项不会删除 SSISDB 数据库，因此，您可以安全地删除该选项，而不会丢失 SSIS 项目。

您可以在删除后重新启用 SSIS 选项，以重用先前部署到 SSIS 目录的 SSIS 项目。

控制台

以下过程删除 SSIS 选项。

从其选项组中删除 SSIS 选项

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择选项组。
3. 选择具有 SSIS 选项的选项组（在前面的示例中为 `ssis-se-2016`）。
4. 选择 Delete option (删除选项)。
5. 在删除选项下，为待删除的选项选择 SSIS。
6. 在 Apply immediately (立即应用) 下，选择 Yes (是) 可立即删除选项，选择 No (否) 可在下次维护时段时删除。
7. 选择删除。

CLI

以下过程删除 SSIS 选项。

从其选项组中删除 SSIS 选项

- 运行以下命令之一。

  Example

  对于 Linux、macOS 或 Unix：

  ```
  aws rds remove-option-from-option-group \
      --option-group-name ssis-se-2016 \
      --options SSIS \
      --apply-immediately
  ```

  对于 Windows：

  ```
  aws rds remove-option-from-option-group ^
      --option-group-name ssis-se-2016 ^
      --options SSIS ^
      --apply-immediately
  ```

删除 SSISDB 数据库

删除 SSIS 选项后并不会删除 SSISDB 数据库。要删除 SSISDB 数据库，请在删除 SSIS 选项后使用 `rds_drop_ssis_database` 存储过程。

删除 SSIS 数据库

- 使用以下存储过程。

```
USE [msdb]
GO
EXEC dbo.rds_drop_ssis_database
GO
```

删除 SSISDB 数据库后，如果您重新启用 SSIS 选项，则会获得一个新的 SSISDB 目录。

Amazon RDS for SQL Server 中的 SQL Server Reporting Services 支持

Microsoft SQL Server Reporting Services (SSRS) 是一个基于服务器的应用程序，用于生成和分发报告。它是 SQL Server 服务套件的一部分，其中还包括 SQL Server Analysis Services (SSAS) 和 SQL Server Integration Services (SSIS)。SSRS 是在 SQL Server 之上构建的服务。您可以使用它从各种数据源收集数据，并以易于理解和直接可供分析的方式呈现。

Amazon RDS for SQL Server 支持直接在 RDS 数据库实例上运行 SSRS。您可以将 SSR 用于现有或新的数据库实例。

RDS 在以下版本上对于 SQL Server 标准版和企业版支持 SSRS：

- SQL Server 2022，所有版本
- SQL Server 2019，版本 15.00.4043.16.v1 和更高版本
- SQL Server 2017，版本 14.00.3223.3.v1 和更高版本

目录

限制和建议

以下限制和建议适用于在 RDS for SQL Server 上运行 SSRS：

- 您不能在具有只读副本的数据库实例上使用 SSRS。
- 实例必须使用自行管理的 Active Directory 或 Amazon Directory Service for Microsoft Active Directory，才能进行 SSRS Web 门户和 Web 服务器身份验证。有关更多信息，请参阅 将 Active Directory 用于 RDS for SQL Server。
- 您无法备份使用 SSRS 选项创建的报表服务器数据库。
- 不支持从 SSRS 的其他实例导入和恢复报告服务器数据库。有关更多信息，请参阅 报告服务器数据库。
- 您无法将 SSRS 配置为侦听默认 SSL 端口 (443)。允许的值为 1150–49511，但不包括 1234、1434、3260、3343、3389 和 47001。
- 不支持通过 Microsoft Windows 文件共享进行订阅。
- 不支持使用报告服务配置管理器。
- 不支持创建和修改角色。
- 不支持修改报告服务器属性。
- 未授予系统管理员和系统用户角色。
- 无法通过 Web 门户编辑系统级角色分配。

开启 SSRS

使用以下过程为数据库实例开启 SSRS：

1. 创建新的选项组或选择现有的选项组。
2. 将 SSRS 选项添加到该选项组。
3. 将选项组与数据库实例相关联。
4. 允许对 SSRS 侦听器端口的 Virtual Private Cloud (VPC) 安全组进行入站访问。

为 SSRS 创建选项组

要使用 SSRS，请创建对应于计划使用的 SQL Server 引擎和数据库实例版本的选项组。为此，请使用 Amazon Web Services Management Console或 Amazon CLI。

> **Note**
> 如果现有选项组针对的是正确的 SQL Server 引擎和版本，也可以使用现有选项组。

控制台

以下过程创建适用于 SQL Server 标准版 2017 的选项组。

创建选项组

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择选项组。
3. 选择创建组。
4. 在创建选项组窗格中，执行以下操作：
   a. 对于 Name（名称），输入选项组的名称，该名称在 Amazon Web Services 账户内具有唯一性，例如 **`ssrs-se-2017`**。此名称只能包含字母、数字和连字符。
   b. 对于描述，输入选项组的简要描述，例如 **`SSRS option group for SQL Server SE 2017`**。此说明用于显示说明。
   c. 对于引擎，选择 sqlserver-se。
   d. 对于主引擎版本，选择 14.00。
5. 选择创建。

CLI

以下过程创建适用于 SQL Server 标准版 2017 的选项组。

创建选项组

- 运行以下命令之一。

Example

对于 Linux、macOS 或 Unix：

```
aws rds create-option-group \
    --option-group-name ssrs-se-2017 \
    --engine-name sqlserver-se \
    --major-engine-version 14.00 \
    --option-group-description "SSRS option group for SQL Server SE 2017"
```

对于 Windows：

```
aws rds create-option-group ^
    --option-group-name ssrs-se-2017 ^
    --engine-name sqlserver-se ^
    --major-engine-version 14.00 ^
    --option-group-description "SSRS option group for SQL Server SE 2017"
```

将 SSRS 选项添加到选项组

接下来，使用 Amazon Web Services Management Console或 Amazon CLI 将 SSRS 选项添加到您的选项组。

控制台

添加 SSRS 选项

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择选项组。
3. 选择刚创建的选项组，然后选择 Add Option（添加选项）。
4. 在选项详细信息下，为选项名称选择 SSRS。
5. 在选项设置下，执行以下操作：

    a. 输入 SSRS 服务要侦听的端口。默认值为 8443。有关允许值的列表，请参阅 限制和建议。
    b. 输入最大内存的值。

        最大内存指定阈值上限，超过该阈值后，不会允许报告服务器应用程序的新内存分配请求。该数字是占数据库实例总内存的百分比。允许的值为 10–80。

   c. 对于 Security groups (安全组)，请选择要与选项关联的 VPC 安全组。使用与您数据库实例关联的相同安全组。

6. 要使用 SSRS 电子邮件发送报告，选择 Email delivery in reporting services（报告服务中的电子邮件递送）下的 Configure email delivery options（配置电子邮件递送选项）复选框，然后执行以下操作：

   a. 对于 Sender email address（发件人电子邮件地址），在 SSRS 电子邮件发送的邮件的 From（发件人）字段中输入要使用的电子邮件地址。

      指定有权从 SMTP 服务器发送邮件的用户账户。

   b. 对于 SMTP server（SMTP 服务器），请指定要使用的 SMTP 服务器或网关。

      它可以是 IP 地址、公司内部网中计算机的 NetBIOS 名称或完全限定域名。

   c. 对于 SMTP port（SMTP 端口），输入要用于连接到邮件服务器的端口。默认值为 25。

   d. 要使用身份验证：

      i. 选择 Use authentication（使用身份验证）复选框。

      ii. 对于 Secret Amazon Resource Name (ARN) [密钥 Amazon 资源名称（ARN）]，输入用户凭证的 Amazon Secrets Manager ARN。

          使用以下格式：

          arn:aws:secretsmanager:*Region*:*AccountId*:secret:*SecretName*-*6RandomChara*

          例如：

          arn:aws:secretsmanager:*us-west-2*:*123456789012*:secret:*MySecret-a1b2c3*

          有关创建密钥的更多信息，请参阅使用 SSRS 电子邮件发送报告。

   e. 选中 Use Secure Sockets Layer (SSL) [使用安全套接字层 (SSL)] 复选框可使用 SSL 加密电子邮件。

7. 在 计划 (Scheduling) 下，选择是立即添加选项还是在下一个维护时段添加选项。

8. 选择添加选项。

CLI

添加 SSRS 选项

1. 创建 JSON 文件，例如：`ssrs-option.json`。

   a. 设置以下必需参数：

      - `OptionGroupName` – 您之前创建或选择的选项组的名称（以下示例中的 `ssrs-se-2017`）。
      - `Port` – SSRS 服务要侦听的端口。默认值为 8443。有关允许值的列表，请参阅 限制和建议。
      - `VpcSecurityGroupMemberships` – RDS 数据库实例的 VPC 安全组成员。
      - `MAX_MEMORY` – 阈值上限，超过该阈值后，不会允许报告服务器应用程序的新内存分配请求。该数字是占数据库实例总内存的百分比。允许的值为 10–80。

   b. （可选）设置以下参数以使用 SSRS 电子邮件：

      - `SMTP_ENABLE_EMAIL` – 设置为 `true` 以使用 SSRS 电子邮件。默认为 `false`。
      - `SMTP_SENDER_EMAIL_ADDRESS` – SSRS 电子邮件发送的邮件的 From（发件人）字段中要使用的电子邮件地址。指定有权从 SMTP 服务器发送邮件的用户账户。
      - `SMTP_SERVER` – 要使用的 SMTP 服务器或网关。它可以是 IP 地址、公司内部网中计算机的 NetBIOS 名称或完全限定域名。
      - `SMTP_PORT` – 用于连接到邮件服务器的端口。默认值为 25。
      - `SMTP_USE_SSL` – 设置为 `true` 以使用 SSL 加密电子邮件。默认为 `true`。
      - `SMTP_EMAIL_CREDENTIALS_SECRET_ARN` – 用于保存用户凭证的 Secrets Manager ARN。采用以下格式：

        `arn:aws:secretsmanager:`*`Region`*`:`*`AccountId`*`:secret:`*`SecretName`*`-`*`6RandomCharact`*

        有关创建密钥的更多信息，请参阅使用 SSRS 电子邮件发送报告。
      - `SMTP_USE_ANONYMOUS_AUTHENTICATION` – 设置为 `true`，如果不想使用身份验证，则不包括 `SMTP_EMAIL_CREDENTIALS_SECRET_ARN`。

        当 `SMTP_ENABLE_EMAIL` 为 `true` 时，原定设置值为 `false`。

   以下示例包含使用密钥 ARN 的 SSRS 电子邮件参数。

```
{
"OptionGroupName": "ssrs-se-2017",
"OptionsToInclude": [
 {
 "OptionName": "SSRS",
 "Port": 8443,
 "VpcSecurityGroupMemberships": ["sg-0abcdef123"],
 "OptionSettings": [
            {"Name": "MAX_MEMORY","Value": "60"},
            {"Name": "SMTP_ENABLE_EMAIL","Value": "true"}
            {"Name": "SMTP_SENDER_EMAIL_ADDRESS","Value": "nobody@example.com"},
            {"Name": "SMTP_SERVER","Value": "email-smtp.us-west-2.amazonaws.com"},
            {"Name": "SMTP_PORT","Value": "25"},
            {"Name": "SMTP_USE_SSL","Value": "true"},
            {"Name": "SMTP_EMAIL_CREDENTIALS_SECRET_ARN","Value":
 "arn:aws:secretsmanager:us-west-2:123456789012:secret:MySecret-a1b2c3"}
            ]
 }],
"ApplyImmediately": true
}
```

2. 将 SSRS 选项添加到该选项组。

   Example

   对于 Linux、macOS 或 Unix：

```
aws rds add-option-to-option-group \
    --cli-input-json file://ssrs-option.json \
    --apply-immediately
```

   对于 Windows：

```
aws rds add-option-to-option-group ^
    --cli-input-json file://ssrs-option.json ^
    --apply-immediately
```

将选项组与数据库实例关联

使用 Amazon Web Services Management Console 或 Amazon CLI 将您的选项组与数据库实例关联。

如果您使用现有数据库实例，则该实例必须已经关联了 Active Directory 域和 Amazon Identity and Access Management (IAM) 角色。如果您创建新实例，请指定现有的 Active Directory 域和 IAM 角色。有关更多信息，请参阅 将 Active Directory 用于 RDS for SQL Server。

控制台

您可将您的选项组与新的或现有的数据库实例关联：

- 对于新数据库实例，请在启动实例时关联选项组。有关更多信息，请参阅“创建 Amazon RDS 数据库实例”。
- 对于现有数据库实例，请修改实例并关联新选项组。有关更多信息，请参阅 修改 Amazon RDS 数据库实例。

CLI

您可将您的选项组与新的或现有的数据库实例关联。

创建使用选项组的数据库实例

- 指定创建选项组时使用的相同数据库引擎类型和主要版本。

  Example

  对于 Linux、macOS 或 Unix：

  ```
  aws rds create-db-instance \
      --db-instance-identifier myssrsinstance \
      --db-instance-class db.m5.2xlarge \
      --engine sqlserver-se \
      --engine-version 14.00.3223.3.v1 \
      --allocated-storage 100 \
      --manage-master-user-password  \
      --master-username admin \
      --storage-type gp2 \
      --license-model li \
      --domain-iam-role-name my-directory-iam-role \
      --domain my-domain-id \
      --option-group-name ssrs-se-2017
  ```

  对于 Windows：

```
aws rds create-db-instance ^
    --db-instance-identifier myssrsinstance ^
    --db-instance-class db.m5.2xlarge ^
    --engine sqlserver-se ^
    --engine-version 14.00.3223.3.v1 ^
    --allocated-storage 100 ^
    --manage-master-user-password ^
    --master-username admin ^
    --storage-type gp2 ^
    --license-model li ^
    --domain-iam-role-name my-directory-iam-role ^
    --domain my-domain-id ^
    --option-group-name ssrs-se-2017
```

修改数据库实例以使用选项组

- 运行以下命令之一。

  Example

  对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier myssrsinstance \
    --option-group-name ssrs-se-2017 \
    --apply-immediately
```

  对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier myssrsinstance ^
    --option-group-name ssrs-se-2017 ^
    --apply-immediately
```

允许对 VPC 安全组的入站访问

要允许对与您数据库实例关联的 VPC 安全组的入站访问，请为指定的 SSRS 侦听器端口创建入站规则。有关设置安全组的更多信息，请参阅通过创建安全组提供对 VPC 中的数据库实例的访问。

报告服务器数据库

当数据库实例与 SSRS 选项关联时，将在数据库实例上创建两个新的数据库：

- `rdsadmin_ReportServer`
- `rdsadmin_ReportServerTempDB`

这些数据库充当 ReportServer 和 ReportServerTempDB 数据库。SSRS 将其数据存储在 ReportServer 数据库中，并将其数据缓存在 ReportServerTempDB 数据库中。有关更多信息，请参阅 Microsoft 文档中的报表服务器数据库。

RDS 拥有和管理这些数据库，因此不允许对它们进行 ALTER 和 DROP 等数据库操作。不允许访问 `rdsadmin_ReportServerTempDB` 数据库。但是，您可以对 `rdsadmin_ReportServer` 数据库执行读取操作。

SSRS 日志文件

您可以列出、查看并下载 SSRS 日志文件。SSRS 日志文件遵循 ReportServerService_*timestamp*.log 的命名约定。这些报表服务器日志位于 `D:\rdsdbdata\Log\SSRS` 目录中。（`D:\rdsdbdata\Log` 目录也是错误日志和 SQL Server 代理日志的父目录。）有关更多信息，请参阅 查看和列出数据库日志文件。

对于现有 SSRS 实例，可能需要重启 SSRS 服务才能访问报表服务器日志。可以通过更新 SSRS 选项来重启服务。

有关更多信息，请参阅 使用 Amazon RDS for Microsoft SQL Server 日志。

访问 SSRS Web 门户

使用以下过程访问 SSRS Web 门户：

1. 开启安全套接字层（SSL）。
2. 将访问权限授予域用户。
3. 使用浏览器和域用户凭证访问 Web 门户。

在 RDS 上使用 SSL

SSRS 为其连接使用 HTTPS SSL 协议。若要使用此协议，请将 SSL 证书导入客户端计算机上的 Microsoft Windows 操作系统。

有关 SSL 证书的更多信息，请参阅使用 SSL/TLS 加密与数据库实例或集群的连接。有关将 SSL 与 SQL Server 一起使用的更多信息，请参阅将 SSL 与 Microsoft SQL Server 数据库实例结合使用。

将访问权限授予域用户

在新的 SSRS 激活中，SSRS 中没有角色分配。为了授予域用户或用户组访问 Web 门户的访问权限，RDS 提供了一个存储过程。

向 Web 门户上的域用户授予访问权限

- 使用以下存储过程。

```
exec msdb.dbo.rds_msbi_task
@task_type='SSRS_GRANT_PORTAL_PERMISSION',
@ssrs_group_or_username=N'AD_domain\user';
```

向域用户或用户组授予 `RDS_SSRS_ROLE` 系统角色。此角色授予了以下系统级任务：

- 运行报告
- 管理作业。
- 管理共享计划
- 查看共享计划

还授予根文件夹上 `Content Manager` 的项目级角色。

访问 Web 门户

`SSRS_GRANT_PORTAL_PERMISSION` 任务成功完成后，您具有使用 Web 浏览器访问门户的权限。Web 门户 URL 的格式如下。

```
https://rds_endpoint:port/Reports
```

在此格式中，以下项适用：

- *`rds_endpoint`* – 与 SSRS 一起使用的 RDS 数据库实例的终端节点。

  您可以在数据库实例的连接和安全性选项卡上找到终端节点。有关更多信息，请参阅“连接到 Microsoft SQL Server 数据库实例”。

- *port* – 您在 SSRS 选项中设置的 SSRS 侦听器端口。

访问 Web 门户

1. 在浏览器中输入 Web 门户网址。

```
https://myssrsinstance.cg034itsfake.us-east-1.rds.amazonaws.com:8443/Reports
```

2. 使用您通过 `SSRS_GRANT_PORTAL_PERMISSION` 任务授予访问权限的域用户的凭证登录。

部署报告和配置报告数据来源

使用以下过程将报告部署到 SSRS 并配置报告数据来源：

主题

- 将报告部署到 SSRS
- 配置报告数据来源。

将报告部署到 SSRS

访问 Web 门户后，您可以将报告部署到其中。您可以使用 Web 门户中的上传工具来上传报告，或直接从 SQL Server Data Tools (SSDT) 部署。从 SSDT 部署时，请确保以下各项：

- 启动 SSDT 的用户可以访问 SSRS Web 门户。
- SSRS 项目属性中的 `TargetServerURL` 值设置为带 `ReportServer` 后缀的 RDS 数据库实例的 HTTPS 终端节点，例如：

```
https://myssrsinstance.cg034itsfake.us-east-1.rds.amazonaws.com:8443/ReportServer
```

配置报告数据来源。

将报告部署到 SSRS 后，应配置报告数据来源。配置报告数据来源时，请确保满足以下条件：

- 对于已加入 Amazon Directory Service for Microsoft Active Directory 的 RDS for SQL Server 数据库实例，使用完全限定域名（FQDN）作为连接字符串的数据来源名称。以 *myssrsinstance.corp-ad.example.com* 为例，其中 *myssrsinstance* 是数据库实例名称，*corp-ad.example.com* 是完全限定域名。

- 对于已加入自行管理的 Active Directory 的 RDS for SQL Server 数据库实例，请使用 `.` 或 *`LocalHost`* 作为连接字符串的数据来源名称。

使用 SSRS 电子邮件发送报告

SSRS 包括 SSRS 电子邮件扩展，您可以使用它向用户发送报告。

要配置 SSRS 电子邮件，请使用 SSRS 选项设置。有关更多信息，请参阅 将 SSRS 选项添加到选项组。

配置 SSRS 电子邮件后，您可以在报告服务器上订阅报表。有关更多信息，请参阅 Microsoft 文档中的报告服务中的电子邮件递送。

与 Amazon Secrets Manager 集成是 SSRS 电子邮件在 RDS 上运行所必需的。要与 Secrets Manager 集成，您需要创建密钥。

> **Note**
>
> 如果您稍后更改密钥，则还必须更新选项组中的 SSRS 选项。

为 SSRS 电子邮件创建密钥

1. 按照《Amazon Secrets Manager 用户指南》中创建密钥的步骤操作。

   a. 对于选择密钥类型，选择其他密钥类型。

   b. 对于 Key/value pairs（键值对），输入以下内容：

      - **SMTP_USERNAME** – 输入有权从 SMTP 服务器发送邮件的用户。
      - **SMTP_PASSWORD** – 输入 SMTP 用户的密码。

   c. 对于 Encryption key（加密密钥），不要使用原定设置 Amazon KMS key。使用您自己的现有密钥，或创建新密钥。

      KMS 密钥策略必须允许 `kms:Decrypt` 操作，例如：

```
{
    "Sid": "Allow use of the key",
    "Effect": "Allow",
    "Principal": {
```

```
            "Service": [
                "rds.amazonaws.com"
            ]
        },
        "Action": [
            "kms:Decrypt"
        ],
        "Resource": "*"
    }
```

2. 按照《Amazon Secrets Manager 用户指南》中的将权限策略附加到密钥中的步骤进行操作。权限策略将 `secretsmanager:GetSecretValue` 操作提供给 `rds.amazonaws.com` 服务主体。

   我们建议您在策略中使用 `aws:sourceAccount` 和 `aws:sourceArn` 条件，以避免出现混淆代理人问题。将您的 Amazon Web Services 账户用于 `aws:sourceAccount`，并将选项组 ARN 用于 `aws:sourceArn`。有关更多信息，请参阅 防范跨服务混淆代理问题。

   下面显示了一个示例权限策略。

```
{
  "Version" : "2012-10-17",
  "Statement" : [ {
    "Effect" : "Allow",
    "Principal" : {
      "Service" : "rds.amazonaws.com"
    },
    "Action" : "secretsmanager:GetSecretValue",
    "Resource" : "*",
    "Condition" : {
      "StringEquals" : {
        "aws:sourceAccount" : "123456789012"
      },
      "ArnLike" : {
        "aws:sourceArn" : "arn:aws:rds:us-west-2:123456789012:og:ssrs-se-2017"
      }
    }
  } ]
}
```

   有关更多示例，请参阅《Amazon Secrets Manager 用户指南》中的 Amazon Secrets Manager 的权限策略示例。

撤销系统级权限

RDS_SSRS_ROLE 系统角色没有足够的权限来删除系统级角色分配。要从 RDS_SSRS_ROLE 中删除用户或用户组，请使用在授予角色时使用的相同存储过程，但使用 SSRS_REVOKE_PORTAL_PERMISSION 任务类型。

撤消域用户对 Web 门户的访问权限

- 使用以下存储过程。

```
exec msdb.dbo.rds_msbi_task
@task_type='SSRS_REVOKE_PORTAL_PERMISSION',
@ssrs_group_or_username=N'AD_domain\user';
```

执行此操作将从 RDS_SSRS_ROLE 系统角色中删除用户。如果用户有 Content Manager 项目级角色，则还会从其中删除该用户。

监控任务的状态

要跟踪您的授予或撤消任务的状态，请调用 rds_fn_task_status 函数。它获取两个参数。第一个参数应该始终为 NULL，因为它不适用于 SSRS。第二个参数接受任务 ID。

要查看所有任务的列表，请将第一个参数设置为 NULL，将第二个参数设置为 0，如以下示例所示。

```
SELECT * FROM msdb.dbo.rds_fn_task_status(NULL,0);
```

要获取特定任务，请将第一个参数设置为 NULL，将第二个参数设置为任务 ID，如以下示例所示。

```
SELECT * FROM msdb.dbo.rds_fn_task_status(NULL,42);
```

rds_fn_task_status 函数将返回以下信息。

输出参数	描述
task_id	任务的 ID。
task_type	对于 SSRS，任务可以具有以下任务类型： • SSRS_GRANT_PORTAL_PERMISSION

输出参数	描述
	• SSRS_REVOKE_PORTAL_PERMISSION
database_name	不适用于 SSRS 任务。
% complete	用百分比表示的任务进度。
duration (mins)	在任务上花费的时间 (以分钟为单位)。
lifecycle	任务的状态。有以下可能状态： • CREATED 调用 SSRS 存储过程之一后，将创建一个任务并将状态设置为 CREATED。 • IN_PROGRESS – 任务开始后，状态将设置为 IN_PROGRESS 。要将状态从 CREATED 更改为 IN_PROGRESS 可能需要最多 5 分钟的时间。 • SUCCESS – 任务完成后，状态将设置为 SUCCESS。 • ERROR – 如果任务失败，状态将设置为 ERROR。有关错误的更多信息，请参阅 task_info 列。 • CANCEL_REQUESTED – 在您调用 rds_cancel_task 存储过程后，任务的状态将设置为 CANCEL_REQUESTED 。 • CANCELLED – 在成功取消任务后，任务的状态将设置为 CANCELLED 。
task_info	有关任务的其他信息。如果在处理过程中发生错误，则此列包含有关错误的信息。
last_updated	上次更新任务状态的日期和时间。

输出参数	描述
created_at	任务的创建日期和时间。
S3_object_arn	不适用于 SSRS 任务。
overwrite_S3_backup_file	不适用于 SSRS 任务。
KMS_master_key_arn	不适用于 SSRS 任务。
filepath	不适用于 SSRS 任务。
overwrite_file	不适用于 SSRS 任务。
task_metadata	与 SSRS 任务关联的元数据。

禁用和删除 SSRS 数据库

使用以下过程禁用 SSRS 并删除 SSRS 数据库：

主题

- 关闭 SSRS
- 删除 SSRS 数据库

关闭 SSRS

要关闭 SSRS，请从其选项组中删除 SSRS 选项。删除该选项不会删除 SSRS 数据库。有关更多信息，请参阅 删除 SSRS 数据库。

您可以通过重新添加 SSRS 选项以再次开启 SSRS。如果您还删除了 SSRS 数据库，则在同一数据库实例上重新添加此选项会创建新的报告服务器数据库。

控制台

从其选项组中删除 SSRS 选项

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择选项组。

3. 选择具有 SSRS 选项的选项组（在前面的示例中为 `ssrs-se-2017`）。
4. 选择 Delete option (删除选项)。
5. 在删除选项下，为待删除的选项选择 SSRS。
6. 在 Apply immediately (立即应用) 下，选择 Yes (是) 可立即删除选项，选择 No (否) 可在下次维护时段时删除。
7. 选择删除。

CLI

从其选项组中删除 SSRS 选项

- 运行以下命令之一。

  Example

  对于 Linux、macOS 或 Unix：

```
aws rds remove-option-from-option-group \
    --option-group-name ssrs-se-2017 \
    --options SSRS \
    --apply-immediately
```

  对于 Windows：

```
aws rds remove-option-from-option-group ^
    --option-group-name ssrs-se-2017 ^
    --options SSRS ^
    --apply-immediately
```

删除 SSRS 数据库

删除 SSRS 选项不会删除报告服务器数据库。要删除数据库，请使用以下存储过程。

要删除报表服务器数据库，请务必先删除 SSRS 选项。

删除 SSRS 数据库

- 使用以下存储过程。

```
exec msdb.dbo.rds_drop_ssrs_databases
```

RDS for SQL Server 中的 Microsoft 分布式事务处理协调器支持

分布式事务是涉及两个或多个网络主机的数据库事务。RDS for SQL Server 支持在主机之间存在分布式事务，其中一个主机可以是以下之一：

- RDS for SQL Server DB 实例
- 本地 SQL Server 主机
- 安装了 SQL Server 的 Amazon EC2 主机
- 具有支持分布式事务的数据库引擎的任何其他 EC2 主机或 RDS 数据库实例

在 RDS 中，从 SQL Server 2012（版本 11.00.5058.0.v1 及更高版本）开始，RDS for SQL Server 的所有版本都支持分布式事务。该支持使用 Microsoft 分布式事务处理协调器 (MSDTC) 提供。有关 MSDTC 的详细信息，请参阅 Microsoft 文档中的分布式事务处理协调器。

目录

限制

以下限制适用于在 RDS for SQL Server 上使用 MSDTC：

- 使用 SQL Server 数据库镜像的实例上不支持 MSDTC。有关详细信息，请参阅事务 – 可用性组和数据库镜像。
- `in-doubt xact resolution` 参数必须设置为 1 或 2。有关更多信息，请参阅 修改 MSDTC 的参数。
- MSDTC 要求参与分布式事务的所有主机都可以使用其主机名进行解析。RDS 会自动为加入域的实例维护此功能。但是，对于独立实例，请确保手动配置 DNS 服务器。
- SQL Server 2017 版本 14.00.3223.3 及更高版本和 SQL Server 2019 支持 Java Database Connectivity (JDBC) XA 事务。
- 依赖于 RDS 实例上的客户端动态链接库 (DLL) 的分布式事务不受支持。
- 不支持使用自定义 XA 动态链接库。

启用 MSDTC

使用以下过程为数据库实例启用 MSDTC：

1. 创建新的选项组或选择现有的选项组。
2. 将 `MSDTC` 选项添加到该选项组。
3. 创建新的参数组，或选择现有参数组。
4. 修改参数组以将 `in-doubt xact resolution` 参数设置为 1 或 2。
5. 将选项组和参数组与数据库实例关联。

为 MSDTC 创建选项组

使用 Amazon Web Services Management Console 或 Amazon CLI 创建对应于 SQL Server 引擎和数据库实例版本的选项组。

> **Note**
> 如果现有选项组针对的是正确的 SQL Server 引擎和版本，也可以使用现有选项组。

控制台

以下控制台过程创建适用于 SQL Server 标准版 2016 的选项组。

创建选项组

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择选项组。
3. 选择创建组。
4. 在创建选项组窗格中，执行以下操作：

   a. 对于名称，输入选项组的名称，该名称在 Amazon 账户内具有唯一性，例如 **msdtc-se-2016**。此名称只能包含字母、数字和连字符。

   b. 对于描述，输入选项组的简要描述，例如 **MSDTC option group for SQL Server SE 2016**。此说明用于显示说明。

   c. 对于引擎，选择 sqlserver-se。

   d. 对于主要引擎版本，选择 13.00。
5. 选择创建。

CLI

以下示例为 SQL Server 标准版 2016 创建一个选项组。

创建选项组

- 使用以下命令之一。

  Example

  对于 Linux、macOS 或 Unix：

```
aws rds create-option-group \
    --option-group-name msdtc-se-2016 \
    --engine-name sqlserver-se \
    --major-engine-version 13.00 \
    --option-group-description "MSDTC option group for SQL Server SE 2016"
```

  对于 Windows：

```
aws rds create-option-group ^
    --option-group-name msdtc-se-2016 ^
```

```
    --engine-name sqlserver-se ^
    --major-engine-version 13.00 ^
    --option-group-description "MSDTC option group for SQL Server SE 2016"
```

将 MSDTC 选项添加到选项组

接下来，使用 Amazon Web Services Management Console或 Amazon CLI 将 MSDTC 选项添加到选项组。

以下选项设置为必需：

- 端口 – 用于访问 MSDTC 的端口。允许的值为 1150–49151，但不包括 1234、1434、3260、3343、3389 和 47001。默认值是 5000。

  确保在防火墙规则中启用了要使用的端口。此外，请确保根据需要，在与数据库实例关联的安全组的入站和出站规则中启用了此端口。有关更多信息，请参阅 无法连接到 Amazon RDS 数据库实例。
- Security groups（安全组）– RDS 数据库实例的 VPC 安全组成员。
- 身份验证类型 – 主机之间的身份验证模式。支持以下身份验证类型：
  - 双向 – RDS 实例使用集成身份验证双向进行身份验证。如果选择此选项，则与此选项组关联的所有实例都必须加入域。
  - 无 – 主机之间不进行身份验证。我们建议不要在生产环境中使用此模式。
- 事务日志大小 – MSDTC 事务日志的大小。允许的值为 4–1024 MB。默认大小为 4 MB。

以下选项设置可选：

- 启用入站连接 – 是否允许指向与此选项组关联的实例的入站 MSDTC 连接。
- 启用出站连接 – 是否允许来自与此选项组关联的实例的出站 MSDTC 连接。
- 启用 XA – 是否允许 XA 事务。有关 XA 协议的更多信息，请参阅 XA 规范。
- 启用 SNA LU – 是否允许将 SNA LU 协议用于分布式事务。有关 SNA LU 协议支持的更多信息，请参阅 Microsoft 文档中的管理 IBM CICS LU 6.2 事务。

控制台

添加 MSDTC 选项

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。

2. 在导航窗格中，选择选项组。
3. 选择刚刚创建的选项组。
4. 选择添加选项。
5. 在选项详细信息下，为选项名称 选择 MSDTC。
6. 在选项设置下：

   a. 对于端口，输入访问 MSDTC 的端口号。默认值为 5000。
   b. 对于 Security groups (安全组)，请选择要与选项关联的 VPC 安全组。
   c. 对于身份验证类型，请选择双向或无。
   d. 对于事务日志大小，请输入 4–1024 之间的值。默认值为 4。
7. 在其他配置下，执行以下操作：

   a. 对于连接，根据需要选择启用入站连接和启用出站连接。
   b. 对于允许的协议，根据需要选择启用 XA 和启用 SNA LU。
8. 在 计划 (Scheduling) 下，选择是立即添加选项还是在下一个维护时段添加选项。
9. 选择添加选项。

   添加此选项不需要重新启动。

CLI

添加 MSDTC 选项

1. 使用以下必需参数创建 JSON 文件，例如 `msdtc-option.json`：

```
{
"OptionGroupName":"msdtc-se-2016",
"OptionsToInclude": [
 {
 "OptionName":"MSDTC",
 "Port":5000,
 "VpcSecurityGroupMemberships":["sg-0abcdef123"],
 "OptionSettings":[{"Name":"AUTHENTICATION","Value":"MUTUAL"},
{"Name":"TRANSACTION_LOG_SIZE","Value":"4"}]
 }],
"ApplyImmediately": true
}
```

2. 将 MSDTC 选项添加到该选项组。

   Example

   对于 Linux、macOS 或 Unix：

   ```
   aws rds add-option-to-option-group \
       --cli-input-json file://msdtc-option.json \
       --apply-immediately
   ```

   对于 Windows：

   ```
   aws rds add-option-to-option-group ^
       --cli-input-json file://msdtc-option.json ^
       --apply-immediately
   ```

   无需重新启动。

为 MSDTC 创建参数组

为与 SQL Server 版本和数据库实例版本对应的 `in-doubt xact resolution` 参数创建或修改参数组。

控制台

以下示例为 SQL Server 标准版 2016 创建一个参数组。

创建参数组

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择参数组。
3. 选择创建参数组。
4. 在创建参数组窗格中，执行以下操作：

   a. 对于参数组系列，选择 sqlserver-se-13.0。
   b. 对于组名称，输入参数组的标识符，如 **msdtc-sqlserver-se-13**。
   c. 对于描述，输入 **in-doubt xact resolution**。
5. 选择创建。

CLI

以下示例为 SQL Server 标准版 2016 创建一个参数组。

创建参数组

- 使用以下命令之一。

  Example

  对于 Linux、macOS 或 Unix：

```
aws rds create-db-parameter-group \
    --db-parameter-group-name msdtc-sqlserver-se-13 \
    --db-parameter-group-family "sqlserver-se-13.0" \
    --description "in-doubt xact resolution"
```

  对于 Windows：

```
aws rds create-db-parameter-group ^
    --db-parameter-group-name msdtc-sqlserver-se-13 ^
    --db-parameter-group-family "sqlserver-se-13.0" ^
    --description "in-doubt xact resolution"
```

修改 MSDTC 的参数

修改与 SQL Server 版本和数据库实例版本对应的参数组中的 `in-doubt xact resolution` 参数。

对于 MSDTC，将 `in-doubt xact resolution` 参数设置为以下选项之一：

- 1 – `Presume commit`。任何 MSDTC 有疑问的事务都假定为已经提交。
- 2 – `Presume abort`。任何 MSDTC 有疑问的事务都假定为已停止。

有关更多信息，请参阅 Microsoft 文档中的[有疑问的 Xact 解析服务器配置选项](https://docs.microsoft.com)。

控制台

以下示例修改您为 SQL Server 标准版 2016 创建的参数组。

修改参数组

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择参数组。
3. 选择参数组，例如 msdtc-sqlserver-se-13。
4. 在参数下，从参数列表中筛选 **xact**。
5. 选择 in-doubt xact resolution (有疑问的 Xact 解析)。
6. 选择编辑参数。
7. 输入 **1** 或 **2**。
8. 选择 Save changes（保存更改）。

CLI

以下示例修改您为 SQL Server 标准版 2016 创建的参数组。

修改参数组

- 使用以下命令之一。

  Example

  对于 Linux、macOS 或 Unix：

```
aws rds modify-db-parameter-group \
    --db-parameter-group-name msdtc-sqlserver-se-13 \
    --parameters "ParameterName='in-doubt xact
 resolution',ParameterValue=1,ApplyMethod=immediate"
```

  对于 Windows：

```
aws rds modify-db-parameter-group ^
    --db-parameter-group-name msdtc-sqlserver-se-13 ^
    --parameters "ParameterName='in-doubt xact
 resolution',ParameterValue=1,ApplyMethod=immediate"
```

将选项组和参数组与数据库实例关联

您可以使用 Amazon Web Services Management Console 或 Amazon CLI 将 MSDTC 选项组和参数组与数据库实例相关联。

控制台

您可以将 MSDTC 选项组和参数组与新的或现有的数据库实例相关联。

- 对于新数据库实例，请在启动实例时将其关联。有关更多信息，请参阅"创建 Amazon RDS 数据库实例"。
- 对于现有数据库实例，请通过修改实例将其关联。有关更多信息，请参阅"修改 Amazon RDS 数据库实例"。

> **Note**
> 如果您使用加入域的现有数据库实例，则该实例必须已经关联了 Active Directory 域和 Amazon Identity and Access Management (IAM) 角色。如果您创建新的加入域实例，请指定现有的 Active Directory 域和 IAM 角色。有关更多信息，请参阅 将 Amazon Managed Active Directory 用于 RDS for SQL Server。

CLI

您可以将 MSDTC 选项组和参数组与新的或现有的数据库实例相关联。

> **Note**
> 如果您使用现有的加入域数据库实例，则该实例必须已经关联了 Active Directory 域和 IAM 角色。如果您创建新的加入域实例，请指定现有的 Active Directory 域和 IAM 角色。有关更多信息，请参阅"将 Amazon Managed Active Directory 用于 RDS for SQL Server"。

使用 MSDTC 选项组和参数组创建数据库实例

- 指定创建选项组时使用的相同数据库引擎类型和主要版本。

  Example

  对于 Linux、macOS 或 Unix：

```
aws rds create-db-instance \
    --db-instance-identifier mydbinstance \
    --db-instance-class db.m5.2xlarge \
    --engine sqlserver-se \
    --engine-version 13.00.5426.0.v1 \
    --allocated-storage 100 \
    --manage-master-user-password \
    --master-username admin \
    --storage-type gp2 \
    --license-model li \
    --domain-iam-role-name my-directory-iam-role \
    --domain my-domain-id \
    --option-group-name msdtc-se-2016 \
    --db-parameter-group-name msdtc-sqlserver-se-13
```

对于 Windows：

```
aws rds create-db-instance ^
    --db-instance-identifier mydbinstance ^
    --db-instance-class db.m5.2xlarge ^
    --engine sqlserver-se ^
    --engine-version 13.00.5426.0.v1 ^
    --allocated-storage 100 ^
    --manage-master-user-password ^
    --master-username admin ^
    --storage-type gp2 ^
    --license-model li ^
    --domain-iam-role-name my-directory-iam-role ^
    --domain my-domain-id ^
    --option-group-name msdtc-se-2016 ^
    --db-parameter-group-name msdtc-sqlserver-se-13
```

修改数据库实例并关联 MSDTC 选项组和参数组

- 使用以下命令之一。

  Example

  对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier mydbinstance \
    --option-group-name msdtc-se-2016 \
    --db-parameter-group-name msdtc-sqlserver-se-13 \
    --apply-immediately
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier mydbinstance ^
    --option-group-name msdtc-se-2016 ^
    --db-parameter-group-name msdtc-sqlserver-se-13 ^
    --apply-immediately
```

修改 MSDTC 选项

启用 MSDTC 选项后，您可以修改其设置。有关如何修改选项设置的信息，请参阅修改选项设置。

> **Note**
>
> 对 MSDTC 选项设置的一些更改需要重新启动 MSDTC 服务。此要求可能会影响正在运行的分布式事务。

使用事务

使用分布式事务

在 Amazon RDS for SQL Server 中，您可以按照与本地运行分布式事务相同的方式运行分布式事务：

- 使用 .NET 框架 `System.Transactions` 可提升事务，它通过将分布式事务的创建推迟到在需要时进行，以此优化分布式事务。

  在这种情况下，提升是自动的，不需要您进行任何干预。如果事务中只有一个资源管理器，则不执行任何提升。有关隐式事务范围的更多信息，请参阅 Microsoft 文档中的使用事务范围实施隐式事务。

  以下 .NET 实施支持可提升事务：

  - 从 ADO.NET 2.0 开始，`System.Data.SqlClient` 支持 SQL Server 的可提升事务。有关更多信息，请参阅 Microsoft 文档中的 System.Transactions 与 SQL Server 的集成。

  - ODP.NET 支持 System.Transactions。将为在 TransactionsScope 范围内与 Oracle Database 11g 发行版 1（版本 11.1）及更高版本建立的第一个连接创建一个本地事务。建立第二个连接时，此事务将自动提升为分布式事务。有关 ODP.NET 中的分布式事务支持的更多信息，请参阅 Microsoft 文档中的 Microsoft 分布式事务处理协调器集成。
- 使用 BEGIN DISTRIBUTED TRANSACTION 语句。有关更多信息，请参阅 Microsoft 文档中的 BEGIN DISTRIBUTED TRANSACTION (Transact-SQL)。

使用 XA 事务

从 RDS for SQL Server 2017 版本 14.00.3223.3 开始，您可以使用 JDBC 控制分布式事务。当您在 MSDTC 选项将 Enable XA 选项设置为 true 时，RDS 会自动启用 JDBC 事务并授予 SqlJDBCXAUser 角色给 guest 用户。这允许通过 JDBC 执行分布式事务。有关包括代码示例在内的更多信息，请参阅 Microsoft 文档中的了解 XA 事务。

使用事务跟踪

RDS 支持控制 MSDTC 事务跟踪并从 RDS 数据库实例下载它们以进行故障排除。您可以通过运行以下 RDS 存储过程来控制事务跟踪会话。

```
exec msdb.dbo.rds_msdtc_transaction_tracing 'trace_action',
[@traceall='0|1'],
[@traceaborted='0|1'],
[@tracelong='0|1'];
```

以下参数是必需参数：

- trace_action – 跟踪操作。它可以是 START、STOP 或 STATUS。

以下参数为可选参数：

- @traceall – 设置为 1 可跟踪所有分布式事务。默认值为 0。
- @traceaborted – 设置为 1 可跟踪已取消的分布式事务。默认值为 0。
- @tracelong – 设置为 1 可跟踪长时间运行的分布式事务。默认值为 0。

Example 启动跟踪操作示例

要启动新的事务跟踪会话，请运行以下示例语句。

```
exec msdb.dbo.rds_msdtc_transaction_tracing 'START',
@traceall='0',
@traceaborted='1',
@tracelong='1';
```

> **Note**
>
> 一次只能有一个活动的事务跟踪会话。如果在有跟踪会话处于活动状态时发出了新的跟踪会话 START 命令，则返回错误，并且活动的跟踪会话保持不变。

Example 停止跟踪操作示例

要停止事务跟踪会话，请运行以下语句。

```
exec msdb.dbo.rds_msdtc_transaction_tracing 'STOP'
```

此语句停止活动的事务跟踪会话，并将事务跟踪数据保存到 RDS 数据库实例上的日志目录中。输出的第一行包含整体结果，后面的行表示操作的详细信息。

以下是成功跟踪会话停止的示例。

```
OK: Trace session has been successfully stopped.
Setting log file to: D:\rdsdbdata\MSDTC\Trace\dtctrace.log
Examining D:\rdsdbdata\MSDTC\Trace\msdtctr.mof for message formats,  8 found.
Searching for TMF files on path: (null)
Logfile D:\rdsdbdata\MSDTC\Trace\dtctrace.log:
 OS version    10.0.14393  (Currently running on 6.2.9200)
 Start Time    <timestamp>
 End Time      <timestamp>
 Timezone is   @tzres.dll,-932 (Bias is 0mins)
 BufferSize            16384 B
 Maximum File Size     10 MB
 Buffers  Written      Not set (Logger may not have been stopped).
 Logger Mode Settings (11000002) ( circular paged
 ProcessorCount         1
Processing completed   Buffers: 1, Events: 3, EventsLost: 0 :: Format Errors: 0,
 Unknowns: 3
Event traces dumped to d:\rdsdbdata\Log\msdtc_<timestamp>.log
```

您可以使用详细信息查询生成的日志文件的名称。有关从 RDS 数据库实例下载日志文件的更多信息，请参阅 监控 Amazon RDS 日志文件。

跟踪会话日志在实例上保留 35 天。任何较旧的跟踪会话日志都会自动删除。

Example 状态跟踪操作示例

要跟踪事务跟踪会话的状态，请运行以下语句。

```
exec msdb.dbo.rds_msdtc_transaction_tracing 'STATUS'
```

此语句在规则集中将以下内容作为单独的行输出。

```
OK
SessionStatus: <Started|Stopped>
TraceAll: <True|False>
TraceAborted: <True|False>
TraceLongLived: <True|False>
```

第一行表示操作的整体结果：OK 或 ERROR，并带有详细信息（如果适用）。后面的行指示有关跟踪会话状态的详细信息：

- SessionStatus 的值可以是以下值之一：
  - Started，有跟踪会话正在运行时。
  - Stopped，没有跟踪会话正在运行时。
- 跟踪会话标志可以是 True 或 False，具体取决于它们在 START 命令中的设置方式。

禁用 MSDTC

要禁用 MSDTC，请从其选项组中删除 MSDTC 选项。

控制台

从其选项组中删除 MSDTC 选项

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择选项组。
3. 选择具有 MSDTC 选项的选项组（在前面的示例中为 msdtc-se-2016）。

4. 选择 Delete option (删除选项)。
5. 在删除选项下，为待删除的选项选择 MSDTC。
6. 在 Apply immediately (立即应用) 下，选择 Yes (是) 可立即删除选项，选择 No (否) 可在下次维护时段时删除。
7. 选择删除。

CLI

从其选项组中删除 MSDTC 选项

- 使用以下命令之一。

  Example

  对于 Linux、macOS 或 Unix：

  ```
  aws rds remove-option-from-option-group \
      --option-group-name msdtc-se-2016 \
      --options MSDTC \
      --apply-immediately
  ```

  对于 Windows：

  ```
  aws rds remove-option-from-option-group ^
      --option-group-name msdtc-se-2016 ^
      --options MSDTC ^
      --apply-immediately
  ```

RDS for SQL Server 的 MSDTC 故障排除

在某些情况下，您可能无法在客户端计算机上运行的 MSDTC，与在 RDS for SQL Server 数据库实例上运行的 MSDTC 服务之间建立连接。如果是这样，请确保满足以下条件：

- 与数据库实例关联的安全组的入站规则配置正确。有关更多信息，请参阅“无法连接到 Amazon RDS 数据库实例”。
- 您的客户端计算机配置正确。
- 客户端计算机上的 MSDTC 防火墙规则已启用。

配置客户端计算机

1. 打开组件服务。

   或者，在服务器管理器中，选择工具，然后选择组件服务。
2. 展开组件服务，展开计算机，展开 我的电脑，然后展开分布式事务处理协调器。
3. 打开 Local DTC (本地 DTC) 的上下文（右键单击）菜单，然后选择 Properties (属性)。
4. 选择安全性选项卡。
5. 选择以下所有选项：
   - 网络 DTC 访问
   - 允许入站
   - 允许出站
6. 确保选择了正确的身份验证模式：
   - 需要双向身份验证 – 客户端计算机与参与分布式事务的其他节点加入到同一个域中，或者在域之间配置了信任关系。
   - 无需身份验证 – 所有其他情况。
7. 选择确定以保存您的更改。
8. 如果系统提示您重新启动服务，请选择是。

启用 MSDTC 防火墙规则

1. 打开 Windows 防火墙，然后选择高级设置。

   打开服务器管理器，选择工具，然后选择高级安全 Windows 防火墙。

   > **Note**
   > 根据您的操作系统，Windows 防火墙可能称为 Windows Defender 防火墙。
2. 在左侧窗格中选择入站规则。
3. 如果尚未启用以下防火墙规则，请启用：
   - 分布式事务处理协调器 (RPC)
   - 分布式事务处理协调器 (RPC) - EPMAP

  - 分布式事务处理协调器 (TCP-In)

4. 关闭 Windows 防火墙。

Amazon RDS for Microsoft SQL Server 的常见 DBA 任务

本部分针对运行 Microsoft SQL Server 数据库引擎的数据库实例，介绍了一些常见 DBA 任务的 Amazon RDS 特定实施。为了让用户获得托管式服务体验，Amazon RDS 未提供对数据库实例的 Shell 访问权限，并且限制对需要高级权限的某些系统程序和表的访问权限。

> **Note**
>
> 当使用 SQL Server 数据库实例时，您可以运行脚本以修改新创建的数据库，但不能修改 [model] 数据库 (即用作新数据库模型的数据库)。

主题

访问 Amazon RDS 上 Microsoft SQL Server 数据库实例的临时数据库

您现在可以访问 Amazon RDS 上 Microsoft RDS Server 数据库实例的 `tempdb` 数据库。您可以通过 Microsoft SQL Server Management Studio (SSMS) 或任何其他标准 SQL 客户端应用程序，使用 Transact-SQL 在 `tempdb` 上运行代码。有关连接到数据库实例的更多信息，请参阅连接到 Microsoft SQL Server 数据库实例。

向数据库实例的主用户授予对 `CONTROL` 的 `tempdb` 访问权限，以便此用户能够修改 `tempdb` 数据库选项。主用户不是 `tempdb` 数据库的数据库拥有者。如果需要，主用户可以授予其他用户 `CONTROL` 访问权限，使得这些用户也可以修改 `tempdb` 数据库选项。

> **Note**
>
> 您可以在 `tempdb` 数据库上运行数据库控制台命令 (DBCC)。

修改临时数据库选项

您可以在 Amazon RDS 数据库实例上修改 `tempdb` 数据库的数据库选项。有关可以修改哪些选项的详细信息，请参阅 Microsoft 文档中的临时数据库。

最大文件大小等数据库选项在您重新启动数据库实例之后会保留。您可以修改数据库选项以优化导入数据时的性能，以及防止存储空间不足。

优化导入数据时的性能

在将大量数据导入数据库实例时，如果要优化性能，请将临时数据库的 `SIZE` 和 `FILEGROWTH` 属性设置为较大的数字。有关优化 `tempdb` 的更多信息，请参阅 Microsoft 文档中的优化临时数据库性能。

以下示例说明了将大小设置为 100 GB 并将文件增长设置为 10%。

```
alter database[tempdb] modify file (NAME = N'templog', SIZE=100GB, FILEGROWTH = 10%)
```

防止存储问题

要防止 `tempdb` 数据库使用所有可用的磁盘空间，请设置 `MAXSIZE` 属性。以下示例说明了将该属性设置为 2048 MB。

```
alter database [tempdb] modify file (NAME = N'templog', MAXSIZE = 2048MB)
```

缩减临时数据库

可通过两种方法缩减 Amazon RDS 数据库实例上的 tempdb 数据库。可以使用 rds_shrink_tempdbfile 过程，也可以设置 SIZE 属性。

使用 rds_shrink_tempdbfile 过程

您可以使用 Amazon RDS 过程 msdb.dbo.rds_shrink_tempdbfile 缩减 tempdb 数据库。您只能在具有对 rds_shrink_tempdbfile 数据库的 CONTROL 访问权限时调用 tempdb。在调用 rds_shrink_tempdbfile 时，数据库实例不会停机。

rds_shrink_tempdbfile 过程具有以下参数。

参数名称	数据类型	默认值	必需	描述
@temp_filename	SYSNAME	—	必需	要缩减的文件的逻辑名称。
@target_size	int	null	可选	文件的新大小 (以 MB 为单位)。

以下示例获取 tempdb 数据库的文件的名称。

```
use tempdb;
GO

select name, * from sys.sysfiles;
GO
```

以下示例缩减名为 tempdb 的 test_file 数据库文件，并请求新的 10 MB 大小：

```
exec msdb.dbo.rds_shrink_tempdbfile @temp_filename = N'test_file', @target_size = 10;
```

设置 SIZE 属性

您也可以通过设置 tempdb 属性，然后重新启动数据库实例来缩减 SIZE 数据库。有关重新启动数据库实例的更多信息，请参阅重启中的数据库实例。

以下示例说明了将 SIZE 属性设置为 1024 MB。

```
alter database [tempdb] modify file (NAME = N'templog', SIZE = 1024MB)
```

多可用区部署的 TempDB 配置

如果您的 RDS for SQL Server 数据库实例位于使用数据库镜像（DBM）或 Always On 可用性组（AG）的多可用区部署内，请注意以下有关使用 tempdb 数据库的注意事项。

您无法将 tempdb 数据从主数据库实例复制到辅助数据库实例。当您失效转移到辅助数据库实例时，辅助数据库实例上的 tempdb 将为空。

您可以将 tempdb 数据库选项的配置（包括其文件大小和自动增长设置）从主数据库实例同步到辅助数据库实例。所有 RDS for SQL Server 版本都支持同步 tempDB 配置。您可以使用以下存储过程开启 tempdb 配置的自动同步：

```
EXECUTE msdb.dbo.rds_set_system_database_sync_objects @object_types = 'TempDbFile';
```

Important

在使用 rds_set_system_database_sync_objects 存储过程之前，请确保已在主数据库实例（而不是辅助数据库实例）上设置了首选 tempdb 配置。如果您在辅助数据库实例上更改了配置，则在您开启自动同步时，您的首选 tempdb 配置可能会被删除。

您可以使用以下函数来确认是否开启了 tempdb 配置的自动同步：

```
SELECT * from msdb.dbo.rds_fn_get_system_database_sync_objects();
```

开启 tempdb 配置自动同步后，将显示 object_class 字段的返回值。当它关闭时，不会返回任何值。

您可以使用以下函数来查找上次同步对象的时间，以 UTC 时间表示：

```
SELECT * from msdb.dbo.rds_fn_server_object_last_sync_time();
```

例如，如果您在 01:00 修改 tempdb 配置，然后运行 rds_fn_server_object_last_sync_time 函数，则为 last_sync_time 返回的值应在 01:00 之后，表示发生了自动同步。

如果您也在使用 SQL Server Agent 任务复制，则可以通过在 @object_type 参数中提供 SQL 代理任务和 tempdb 配置来为它们启用复制：

```
EXECUTE msdb.dbo.rds_set_system_database_sync_objects @object_types =
 'SQLAgentJob,TempDbFile';
```

有关 SQL Server Agent 任务复制的更多信息，请参阅启用 SQL Server Agent 作业复制。

除了使用 `rds_set_system_database_sync_objects` 存储过程来确保自动同步 `tempdb` 配置更改之外，还可以使用以下手动方法之一：

> **Note**
>
> 我们建议使用 `rds_set_system_database_sync_objects` 存储过程开启 `tempdb` 配置的自动同步。使用自动同步可以防止每次更改 `tempdb` 配置时都需要执行这些手动任务。

- 首先修改数据库实例并关闭多可用区，然后修改临时数据库，最后重新启用多可用区。此方法不会有任何停机时间。

  有关更多信息，请参阅"修改 Amazon RDS 数据库实例"。
- 首先在原始主实例中修改 `tempdb`，然后手动执行故障转移，最后在新的主实例中修改 `tempdb`。此方法需要停机。

  有关更多信息，请参阅 重启中的数据库实例。

使用 Database Engine Tuning Advisor 分析 Amazon RDS for SQL Server 数据库实例上的数据库工作负载

Database Engine Tuning Advisor 是 Microsoft 提供的客户端应用程序，用于分析数据库工作负载，并根据您运行的查询种类对 Microsoft SQL Server 数据库提供最佳索引集建议。像 SQL Server Management Studio 一样，在连接到运行 SQL Server 的 Amazon RDS 数据库实例的客户端计算机上运行 Tuning Advisor。客户端计算机可以是在您自有网络内部运行的本地计算机，也可以是在您的 Amazon RDS 数据库实例所在区域中运行的 Amazon EC2 Windows 实例。

本部分介绍如何捕获工作负载供 Tuning Advisor 分析。对于捕获工作负载，这是一个首选过程，因为 Amazon RDS 限制了主机对 SQL Server 实例的访问。有关更多信息，请参阅 Microsoft 文档中的 Database Engine Tuning Advisor。

要使用 Tuning Advisor，必须为其提供所谓的工作负载。工作负载是一系列 Transact - SQL 语句，它们是针对您要优化的一个或多个数据库上运行的。在优化数据库时，Database Engine Tuning Advisor

会以工作负载输入的形式使用跟踪文件、跟踪表、Transact - SQL 脚本或 XML 文件。在使用 Amazon RDS 时，工作负载可能是客户端计算机上的文件，也可能是客户端计算机可访问的 Amazon RDS for SQL Server 数据库上的数据库表。该文件或表必须包含对您要优化的数据库进行查询的命令，并且这些命令必须是适合重新执行的格式。

为了让 Tuning Advisor 实现最大的效率，工作负载应尽可能真实。您可以通过对数据库实例执行跟踪来生成工作负载文件或表。运行跟踪时，可以在数据库实例上模拟负载，也可以用正常负载运行应用程序。

跟踪方式有两种：客户端跟踪和服务器端跟踪。客户端跟踪的设置比较容易，并且可以在 SQL Server Profiler 中实时观察捕获到的跟踪事件。服务器端跟踪的设置比较复杂，并需要编写一些 Transact - SQL 脚本。此外，因为跟踪会写入 Amazon RDS 数据库实例上的文件中，所以跟踪会占用存储空间。请务必跟踪正在运行的服务器端跟踪所占用的存储空间大小，这点非常重要，因为数据库实例可能会进入存储已满状态，从而会在存储空间不足时不再可用。

对于客户端跟踪，在 SQL Server Profiler 捕获到足量的跟踪数据后，您可以通过以下方式生成工作负载文件：将跟踪保存到本地计算机上的文件中，或保存到客户端计算机可用的数据库实例上的数据库表中。使用客户端跟踪的主要缺点是，在高负载情况下，跟踪可能无法捕获到所有查询。这可能会降低 Database Engine Tuning Advisor 执行的分析的有效性。如果您需要在高负载下运行跟踪并要确保可捕获到跟踪会话期间的所有查询，则应使用服务器端跟踪。

对于服务器端跟踪，您必须将数据库实例上的跟踪文件存储到合适的工作负载文件，您也可以在完成跟踪后将跟踪保存到数据库实例上的表中。您可以使用 SQL Server Profiler 将跟踪保存到本地计算机上的文件，或从数据库实例的跟踪表中读取 Tuning Advisor。

在 SQL Server 数据库实例上运行客户端跟踪

在 SQL Server 数据库实例上运行客户端跟踪

1. 启动 SQL Server Profiler。此工具安装在 SQL Server 实例文件夹的“性能工具”文件夹内。要启动客户端跟踪，您必须加载或定义一个跟踪定义模板。
2. 在 SQL Server Profiler 的“File (文件)”菜单中，选择 New Trace (新跟踪)。在 Connect to Server 对话框中，输入要运行跟踪的数据库的数据库实例终端节点、端口、主用户名和密码。
3. 在 Trace Properties 对话框中，输入跟踪名称并选择跟踪定义模板。应用程序附带有默认模板 TSQL_Replay。您可以通过编辑此模板来定义跟踪。在 Trace Properties 对话框的 Events Selection 选项卡下编辑事件和事件信息。

   有关跟踪定义模板及使用 SQL Server Profiler 指定客户端跟踪的详细信息，请参阅 Microsoft 文档中的 Database Engine Tuning Advisor。

4. 启动客户端跟踪并实时查看在数据库实例上运行的 SQL 查询。
5. 完成跟踪后，在 File (文件) 菜单中选择 Stop Trace (停止跟踪)。在数据库实例上将结果另存为文件或跟踪表。

在 SQL Server 数据库实例上运行服务器端跟踪

通过编写脚本创建服务器端跟踪这一过程可能较为复杂，且已超出本文档的讨论范围。本部分包含一些您可以用作示例的示例脚本。与客户端跟踪一样，服务器端跟踪的目的是，创建可通过 Database Engine Tuning Advisor 打开的工作负载文件或跟踪表。

下面显示的是一个简化的示例脚本，用于启动服务器端跟踪并将详细信息捕获到工作负载文件。跟踪最初会将信息保存到 D:\RDSDBDATA\Log 目录中的 RDSTrace.trc 文件，且会每 100 MB 转换一个跟踪文件，这样，后续的跟踪文件命会依次命名为 RDSTrace_1.trc、RDSTrace_2.trc 等。

```
DECLARE @file_name NVARCHAR(245) = 'D:\RDSDBDATA\Log\RDSTrace';
DECLARE @max_file_size BIGINT = 100;
DECLARE @on BIT = 1
DECLARE @rc INT
DECLARE @traceid INT

EXEC @rc = sp_trace_create @traceid OUTPUT, 2, @file_name, @max_file_size
IF (@rc = 0) BEGIN
   EXEC sp_trace_setevent @traceid, 10, 1, @on
   EXEC sp_trace_setevent @traceid, 10, 2, @on
   EXEC sp_trace_setevent @traceid, 10, 3, @on
 . . .
   EXEC sp_trace_setfilter @traceid, 10, 0, 7, N'SQL Profiler'
   EXEC sp_trace_setstatus @traceid, 1
   END
```

以下示例是停止跟踪的脚本。请注意，以前的脚本所创建的跟踪会继续运行，直到您显式地停止跟踪或磁盘空间不足以运行此进程为止。

```
DECLARE @traceid INT
SELECT @traceid = traceid FROM ::fn_trace_getinfo(default)
WHERE property = 5 AND value = 1 AND traceid <> 1

IF @traceid IS NOT NULL BEGIN
   EXEC sp_trace_setstatus @traceid, 0
   EXEC sp_trace_setstatus @traceid, 2
```

```
END
```

您可以将服务器端跟踪结果保存到数据库表，然后借助 fn_trace_gettable 函数，将数据库表用作 Tuning Advisor 的工作负载。以下命令会将 D:\rdsdbdata\Log 目录（包括 RDSTrace_1.trc 等所有滚动更新文件）中名为 RDSTrace.trc 的所有文件的结果加载到当前数据库中名为 RDSTrace 的表中。

```
SELECT * INTO RDSTrace
FROM fn_trace_gettable('D:\rdsdbdata\Log\RDSTrace.trc', default);
```

要将特定滚动更新文件保存到表中，例如 RDSTrace_1.trc 文件，请为 fn_trace_gettable 指定滚动更新文件的名称和替换值 1，以代替默认值这一最后的参数。

```
SELECT * INTO RDSTrace_1
FROM fn_trace_gettable('D:\rdsdbdata\Log\RDSTrace_1.trc', 1);
```

使用跟踪运行 Tuning Advisor

创建跟踪后，无论是作为本地文件还是数据库表，都可以针对数据库实例运行 Tuning Advisor。借助 Amazon RDS 使用 Tuning Advisor 的进程与使用远程独立 SQL Server 实例的进程相同。您可以在客户端计算机上使用 Tuning Advisor UI，或者使用命令行 Dta.exe 实用程序。在这两种情况下，您都必须使用数据库实例的终端节点连接 Amazon RDS 数据库实例，并在使用 Tuning Advisor 时提供主用户名和主用户密码。

以下代码示例演示了针对 Amazon RDS 数据库实例（使用终端节点 **dta.cnazcmklsdei.us-east-1.rds.amazonaws.com**）使用 dta.exe 命令行实用程序的方法。此示例包含主用户名 **admin** 和主用户密码 **test**，要优化的示例数据库名为 **C:\RDSTrace.trc**。此外，示例命令行代码会指定名为 **RDSTrace1** 的跟踪会话，并指定输出文件到本地计算机，其中名为 **RDSTrace.sql** 的本地计算机适用于 SQL 输出脚本，名为 **RDSTrace.txt** 的本地计算机适用于结果文件，名为 **RDSTrace.xml** 的本地计算机适用于分析的 XML 文件。此外，还有一个在名为 **RDSTraceErrors** 的 RDSDTA 数据库上指定的错误表。

```
dta -S dta.cnazcmklsdei.us-east-1.rds.amazonaws.com -U admin -P test -D RDSDTA -
if C:\RDSTrace.trc -s RDSTrace1 -of C:\ RDSTrace.sql -or C:\ RDSTrace.txt -ox C:\
 RDSTrace.xml -e RDSDTA.dbo.RDSTraceErrors
```

下面是相同的示例命令行代码，只有一点不同：即输入工作负载是 **RDSTrace** 数据库上名为 **RDSDTA** 的远程 Amazon RDS 实例上的一个表。

```
dta -S dta.cnazcmklsdei.us-east-1.rds.amazonaws.com -U admin -P test -D RDSDTA -it
 RDSDTA.dbo.RDSTrace -s RDSTrace1 -of C:\ RDSTrace.sql -or C:\ RDSTrace.txt -ox C:\
 RDSTrace.xml -e RDSDTA.dbo.RDSTraceErrors
```

有关 dta 实用工具命令行参数的完整列表，请参阅 Microsoft 文档中的 dta 实用工具。

将 Amazon RDS for SQL Server 数据库的 **db_owner** 更改为 **rdsa** 账户

当您在 RDS for SQL Server 数据库实例中创建或还原数据库时，Amazon RDS 会将该数据库的所有者设置为 `rdsa`。如果您使用 SQL Server 数据库镜像（DBM）或 Always On 可用性组（AG）进行多可用区部署，Amazon RDS 会将辅助数据库实例上的数据库所有者设置为 `NT AUTHORITY\SYSTEM`。在辅助数据库实例提升为主角色之前，无法更改辅助数据库的所有者。在大多数情况下，在执行查询时将数据库的所有者设置为 `NT AUTHORITY\SYSTEM` 不会出现问题，但在执行系统存储过程（例如，需要提升权限才能执行的 `sys.sp_updatestats`）时，可能会引发错误。

您可以使用以下查询来识别 `NT AUTHORITY\SYSTEM` 所拥有的数据库的所有者：

```
SELECT name FROM sys.databases WHERE SUSER_SNAME(owner_sid) = 'NT AUTHORITY\SYSTEM';
```

您可以使用 Amazon RDS 存储过程 `rds_changedbowner_to_rdsa` 将数据库的所有者更改为 `rdsa`。不允许将以下数据库与 `rds_changedbowner_to_rdsa` 一起使用：`master, model, msdb, rdsadmin, rdsadmin_ReportServer, rdsadmin_ReportServerTempDB, SSISDB`。

要将数据库的所有者更改为 `rdsa`，请调用 `rds_changedbowner_to_rdsa` 存储过程并提供数据库的名称。

Example 用法：

```
exec msdb.dbo.rds_changedbowner_to_rdsa 'TestDB1';
```

以下参数是必需参数：

- `@db_name` – 要将数据库所有者更改为 `rdsa` 的数据库的名称。

> **Important**
>
> 您不能使用 `rds_changedbowner_to_rdsa` 将数据库的所有权更改为 `rdsa` 之外的登录名。例如，您不能将所有权更改为您创建数据库时所用的登录名。当没有其他数据库用户可

用于授予成员资格时，要在 db_owner 角色中还原主用户丢失的成员资格，请重置主用户密码来在 db_owner 角色中获取成员资格。有关更多信息，请参阅 重置 Amazon RDS for SQL Server 主用户的 db_owner 角色成员资格。

管理 Amazon RDS for Microsoft SQL Server 的排序规则和字符集

本主题提供有关如何在 Amazon RDS 中管理 Microsoft SQL Server 的排序规则和字符集的指导。它说明了如何在数据库创建期间配置排序规则并在以后对其进行修改，从而确保根据语言和区域设置要求正确处理文本数据。此外，它还介绍了在 Amazon RDS 中的 SQL Server 环境内保持兼容性和性能的最佳实践。

SQL Server 支持多个级别的排序规则。您可以在创建数据库实例时设置默认服务器排序规则。您可以在数据库、表或列级别覆盖排序规则。

主题

Microsoft SQL Server 的服务器级别排序规则

在创建 Microsoft SQL Server 数据库实例时，您可以设置所要使用的服务器排序规则。如果您未选择其他排序规则，则服务器级别排序规则默认为 SQL_Latin1_General_CP1_CI_AS。服务器排序规则默认应用到所有数据库和数据库对象。

> **Note**
>
> 从数据库快照还原时，无法更改排序规则。

Amazon RDS 当前支持以下服务器排序规则：

排序规则	描述
Arabic_CI_AS	Arabic，不区分大小写，区分重音，不区分假名，不区分全半角
Chinese_PRC_BIN2	Chinese-PRC，二进制代码点排序顺序

排序规则	描述
Chinese_PRC_CI_AS	Chinese-PRC，不区分大小写，区分重音，不区分假名，不区分全半角
Danish_Norwegian_CI_AS	Danish-Norwegian，不区分大小写，区分重音，不区分假名，不区分全半角
Danish_Norwegian_CI_AS_KS	Danish-Norwegian，不区分大小写，区分重音，区分假名，不区分全半角
Danish_Norwegian_CI_AS_KS_WS	Danish-Norwegian，不区分大小写，区分重音，区分假名，区分全半角
Danish_Norwegian_CI_AS_WS	Danish-Norwegian，不区分大小写，区分重音，不区分假名，区分全半角
Danish_Norwegian_CS_AI	Danish-Norwegian，区分大小写，不区分重音，不区分假名，不区分全半角
Danish_Norwegian_CS_AI_KS	Danish-Norwegian，区分大小写，不区分重音，区分假名，不区分全半角
Finnish_Swedish_100_BIN	Finnish-Swedish-100，二进制排序
Finnish_Swedish_100_BIN2	Finnish-Swedish-100，二进制代码点比较排序
Finnish_Swedish_100_CI_AI	Finnish-Swedish-100，不区分大小写，不区分重音，不区分假名，不区分全半角
Finnish_Swedish_100_CI_AS	Finnish-Swedish-100，不区分大小写，区分重音，不区分假名，不区分全半角
Finnish_Swedish_CI_AS	芬兰语、瑞典语和瑞典语（芬兰）不区分大小写，区分重音，不区分假名，不区分全半角
French_CI_AS	French，不区分大小写，区分重音，不区分假名，不区分全半角
Hebrew_BIN	Hebrew，二进制排序

排序规则	描述
Hebrew_CI_AS	Hebrew，不区分大小写，区分重音，不区分假名，不区分全半角
Japanese_BIN	Japanese，二进制排序
Japanese_CI_AS	Japanese，不区分大小写，区分重音，不区分假名，不区分全半角
Japanese_CS_AS	Japanese，区分大小写，区分重音，不区分假名，不区分全半角
Japanese_XJIS_140_CI_AS	Japanese，不区分大小写，区分重音，不区分假名，不区分全半角，补充字符，不区分变体选择器
Japanese_XJIS_140_CI_AS_KS_VSS	Japanese，不区分大小写，区分重音，区分假名，不区分全半角，补充字符，区分变体选择器
Japanese_XJIS_140_CI_AS_VSS	Japanese，不区分大小写，区分重音，不区分假名，不区分全半角，补充字符，区分变体选择器
Japanese_XJIS_140_CS_AS_KS_WS	Japanese，区分大小写，区分重音，区分假名，区分全半角，补充字符，不区分变体选择器
Korean_Wansung_CI_AS	Korean_Wansung，不区分大小写，区分重音，不区分假名，不区分全半角
Latin1_General_100_BIN	Latin1-General-100，二进制排序
Latin1_General_100_BIN2	Latin1-General-100，二进制代码点排序顺序
Latin1_General_100_BIN2_UTF8	Latin1-General-100，二进制代码点排序顺序，UTF-8 编码
Latin1_General_100_CI_AS	Latin1-General-100，不区分大小写，区分重音，不区分假名，不区分全半角

排序规则	描述
Latin1_General_100_CI_AS_SC_UTF8	Latin1-General-100，不区分大小写，区分重音，补充字符，UTF-8 编码
Latin1_General_BIN	Latin1-General，二进制排序
Latin1_General_BIN2	Latin1-General，二进制代码点排序顺序
Latin1_General_CI_AI	Latin1-General，不区分大小写，不区分重音，不区分假名，不区分全半角
Latin1_General_CI_AS	Latin1-General，不区分大小写，区分重音，不区分假名，不区分全半角
Latin1_General_CI_AS_KS	Latin1-General，不区分大小写，区分重音，区分假名，不区分全半角
Latin1_General_CS_AS	Latin1-General，区分大小写，区分重音，不区分假名，不区分全半角
Modern_Spanish_CI_AS	Modern_Spanish，不区分大小写，区分重音，不区分假名，不区分全半角
Polish_CI_AS	Polish，不区分大小写，区分重音，不区分假名，不区分全半角
SQL_1xCompat_CP850_CI_AS	对于 Unicode 数据为 Latin1-General，不区分大小写，区分重音，不区分假名，不区分全半角，对于非 Unicode 数据，为代码页 850 上的 SQL Server 排序顺序 49
SQL_Latin1_General_CP1_CI_AI	对于 Unicode 数据为 Latin1-General，不区分大小写，不区分重音，不区分假名，不区分全半角；对于非 Unicode 数据，为代码页 1252 上的 SQL Server 排序顺序 54

排序规则	描述
SQL_Latin1_General_CP1_CI_AS（默认值）	对于 Unicode 数据为 Latin1-General，不区分大小写，区分重音，不区分假名，不区分全半角；对于非 Unicode 数据，为代码页 1252 上的 SQL Server 排序顺序 52
SQL_Latin1_General_CP1_CS_AS	对于 Unicode 数据为 Latin1-General，区分大小写，区分重音，不区分假名，不区分全半角；对于非 Unicode 数据，为代码页 1252 上的 SQL Server 排序顺序 51
SQL_Latin1_General_CP437_CI_AI	对于 Unicode 数据为 Latin1-General，不区分大小写，不区分重音，不区分假名，不区分全半角；对于非 Unicode 数据，为代码页 437 上的 SQL Server 排序顺序 34
SQL_Latin1_General_CP850_BIN	Latin1-General，对于 Unicode 数据为二进制排序顺序，对于非 Unicode 数据为代码页 850 上的 SQL Server 排序顺序 40
SQL_Latin1_General_CP850_BIN2	Latin1-General，对于 Unicode 数据为二进制代码点排序顺序，对于非 Unicode 数据为代码页 850 上的 SQL Server 排序顺序 40
SQL_Latin1_General_CP850_CI_AI	对于 Unicode 数据为 Latin1-General，不区分大小写，不区分重音，不区分假名，不区分全半角；对于非 Unicode 数据，为代码页 850 上的 SQL Server 排序顺序 44
SQL_Latin1_General_CP850_CI_AS	对于 Unicode 数据为 Latin1-General，不区分大小写，区分重音，不区分假名，不区分全半角，对于非 Unicode 数据，为代码页 850 上的 SQL Server 排序顺序 42

排序规则	描述
SQL_Latin1_General_CP1256_CI_AS	对于 Unicode 数据为 Latin1-General，不区分大小写，区分重音，不区分假名，不区分全半角；对于非 Unicode 数据，为代码页 1256 上的 SQL Server 排序顺序 146
SQL_Latin1_General_CP1255_CS_AS	对于 Unicode 数据为 Latin1-General，区分大小写，区分重音，不区分假名，不区分全半角；对于非 Unicode 数据，为代码页 1255 上的 SQL Server 排序顺序 137
Thai_CI_AS	Thai，不区分大小写，区分重音，不区分假名，不区分全半角
Turkish_CI_AS	Turkish，不区分大小写，区分重音，不区分假名，不区分全半角

选择排序规则：

- 如果您使用 Amazon RDS 控制台，则在创建新的数据库实例时，请选择 Additional configuration（其他配置），然后在 Collation（排序规则）字段中输入排序规则。有关更多信息，请参阅 创建 Amazon RDS 数据库实例。
- 如果使用 Amazon CLI，请在 `--character-set-name` 命令中使用 `create-db-instance` 选项。有关更多信息，请参阅 create-db-instance。
- 如果使用 Amazon RDS API，请在 `CharacterSetName` 操作中使用 `CreateDBInstance` 参数。有关更多信息，请参阅 CreateDBInstance。

Microsoft SQL Server 的数据库级别排序规则

当创建新数据库或数据库对象时，您可以通过覆盖排序规则在数据库、表或列级别更改默认排序规则。例如，如果您的默认服务器排序规则为 SQL_Latin1_General_CP1_CI_AS，您可将其更改为 Mohawk_100_CI_AS 以支持 Mohawk 排序规则。如果需要，甚至可以对查询中的参数进行类型转换以便使用不同的排序规则。

例如，以下查询将 AccountName 列的默认排序规则更改为 Mohawk_100_CI_AS

```
CREATE TABLE [dbo].[Account]
 (
     [AccountID] [nvarchar](10) NOT NULL,
     [AccountName] [nvarchar](100) COLLATE Mohawk_100_CI_AS NOT NULL
 ) ON [PRIMARY];
```

Microsoft SQL Server 数据库引擎通过内置 NCHAR、NVARCHAR 和 NTEXT 数据类型支持 Unicode。例如，如果您需要 CJK 支持，请对字符存储使用这些 Unicode 数据类型，并在创建数据库和表时覆盖默认的服务器排序规则。下面列出了一些由 Microsoft 提供的链接，其中包括 SQL Server 的排序规则和 Unicode 支持：

- 使用排序规则
- 排序规则和国际术语
- 使用 SQL Server 排序规则
- 数据库和数据库引擎应用程序的国际注意事项

为 Amazon RDS for SQL Server 创建数据库用户

通过运行类似以下示例的 T-SQL 脚本，可以为 Amazon RDS for Microsoft SQL Server 数据库实例创建数据库用户。使用 SQL Server 管理套件（SSMS）之类的应用程序。您以创建数据库实例时创建的主用户身份登录数据库实例。

```
--Initially set context to master database
USE [master];
GO
--Create a server-level login named theirname with password theirpassword
CREATE LOGIN [theirname] WITH PASSWORD = 'theirpassword';
GO
--Set context to msdb database
USE [msdb];
GO
--Create a database user named theirname and link it to server-level login theirname
CREATE USER [theirname] FOR LOGIN [theirname];
GO
```

有关将数据库用户添加到角色的示例，请参阅将用户添加到 SQLagentUser 角色。

Note

如果您在添加用户时出现权限错误，可以通过修改数据库实例主用户密码来恢复。有关更多信息，请参阅 重置 Amazon RDS for SQL Server 主用户的 db_owner 角色成员资格。
在应用程序中克隆主用户权限不是最佳实践。有关更多信息，请参阅 How to clone master user permissions in Amazon RDS for SQL Server。

确定 Amazon RDS for SQL Server 数据库的恢复模式

在 Amazon RDS 中，恢复模式、保留期和数据库状态是相关联的。

在对这些设置之一进行更改之前，请务必了解结果。每个设置都可能会影响其他设置。例如：

- 如果您在启用备份保留的情况下将数据库的恢复模式更改为 SIMPLE 或 BULK_LOGGED，Amazon RDS 会在五分钟内将恢复模式重置为 FULL。这还会导致 RDS 制作数据库实例的快照。
- 如果您将备份保留设置为 `0` 天，则 RDS 会将恢复模式设置为 SIMPLE。
- 如果您在将备份保留设置为 `0` 天的同时，将数据库的恢复模式从 SIMPLE 更改为任何其他选项，则 RDS 会将恢复模式重置为 SIMPLE。

Important

绝不要更改多可用区实例上的恢复模式，即使您似乎可以执行此操作 – 例如，通过使用 ALTER DATABASE。Backup retention，因此，多可用区需要 FULL 恢复模式。如果您更改恢复模式，RDS 会立即将其更改回 FULL。
此自动重置强制 RDS 完全重建镜像。在此重建期间，数据库的可用性将降低，时间约为 30-90 分钟，直到镜像已准备好故障转移。数据库实例还会遇到性能降低，与从单可用区转换为多可用区期间相同。性能降低的时间长度取决于数据库存储大小 — 存储的数据库越大，降低的时间越长。

有关 SQL Server 恢复模型的更多信息，请参阅 Microsoft 文档中的恢复模型 (SQL Server)。

确定 Amazon RDS for SQL Server 的上次失效转移时间

要确定上次故障转移时间，请使用以下存储过程：

```
execute msdb.dbo.rds_failover_time;
```

此过程返回以下信息。

输出参数	描述
errorlog_available_from	显示错误日志在日志目录中开始可用的时间。
recent_failover_time	显示上次故障转移时间（如果在错误日志中可用）。否则显示为 `null`。

> **Note**
> 存储过程搜索日志目录中的所有可用 SQL Server 错误日志，以检索最近的故障转移时间。如果故障转移消息已被 SQL Server 覆盖，则该过程不会检索故障转移时间。

Example 最近无故障转移示例

此示例显示错误日志中最近没有故障转移的输出。自 2020-04-29 23:59:00.01 以来，没有发生故障转移。

errorlog_available_from	recent_failover_time
2020-04-29 23:59:00.0100000	null

Example 最近故障转移示例

此示例显示错误日志中最近有故障转移的输出。最近一次故障转移的时间是在 2020-05-05 18:57:51.89。

errorlog_available_from	recent_failover_time
2020-04-29 23:59:00.0100000	2020-05-05 18:57:51.8900000

排查由于日志序列号差异而导致的时间点故障恢复故障

在 RDS for SQL Server 中尝试时间点故障恢复（PITR）时，由于日志序列号（LSN）存在差异，您可能会遇到故障。这些差异会阻止 RDS 将数据库还原到所请求的时间，并且 RDS 会将正在还原的实例置于 `incompatible-restore` 状态。

此问题的常见原因为：

- 手动更改数据库恢复模式。
- 由于用于完成事务日志备份的资源不足，RDS 更改了自动恢复模式。

要确定数据库中的 LSN 差异，请运行此查询：

```
SELECT * FROM msdb.dbo.rds_fn_list_tlog_backup_metadata(database_name)
ORDER BY backup_file_time_utc desc;
```

如果您发现 LSN 差异，您可以：

- 选择 LSN 差异之前的一个还原点。
- 等待并还原到下一次实例备份完成后的某个点。

为防止出现此问题，我们建议您不要手动更改 RDS for SQL Server 数据库的恢复模式，因为这会中断实例的持久性。我们还建议您选择具有足够资源来处理工作负载的实例类型，以确保定期备份事务日志。

有关事务日志管理的更多信息，请参阅 Microsoft SQL Server 文档中的 [SQL Server transaction log architecture and management guide]。

拒绝或支持查看 Amazon RDS for SQL Server 的数据库名称

主用户无法设置 `DENY VIEW ANY DATABASE TO LOGIN` 来向用户隐藏数据库。要更改此权限，请改用以下存储过程：

- 拒绝 *LOGIN* 的数据库视图访问权限：

```
EXEC msdb.dbo.rds_manage_view_db_permission @permission='DENY',
 @server_principal='LOGIN'
```

```
go
```

- 支持 *LOGIN* 的数据库视图访问权限：

```
EXEC msdb.dbo.rds_manage_view_db_permission @permission='GRANT',
 @server_principal='LOGIN'

go
```

使用此存储过程时，请注意以下事项：

- 数据库名称对于 SSMS 和内部 DMV（动态管理视图）处于隐藏状态。但是，仍然可以从审计、日志和元数据表中看到数据库名称。这些是受保护的 `VIEW ANY DATABASE` 服务器权限。有关更多信息，请参阅 [DENY Server Permissions]。
- 一旦权限恢复为 `GRANT`（支持），*LOGIN* 就可以查看所有数据库。
- 如果您删除并重新创建 *LOGIN*，则与 LOGIN 相关的查看权限将重置为 `ALLOW`。
- 对于多可用区实例，仅在主要主机上为 *LOGIN* 设置 `DENY` 或 `GRANT` 权限。更改会自动传播到辅助主机。
- 此权限仅更改登录名是否可以查看数据库名称。但是，对数据库和其中对象的访问权限是单独管理的。

在 Amazon RDS for SQL Server 批量加载期间禁用快速插入

从 SQL Server 2016 开始，便默认启用了快速插入。快速插入利用数据库处于简单或批量日志记录恢复模型时发生的最小日志记录来优化插入性能。使用快速插入，每个批量加载批次都会获得新范围，从而绕过对具有可用空闲空间的现有范围的分配查找，从而优化插入性能。

然而，使用快速插入时，小批量大小的批量加载可能会增加对象消耗的未使用空间。如果不能提高批量大小，则启用跟踪标志 692 可以帮助减少未使用的预留空间，但以牺牲性能为代价。启用此跟踪标志会在将数据批量加载到堆或集群化索引时禁用快速插入。

使用数据库参数组启用跟踪标志 692 作为启动参数。有关更多信息，请参阅“Amazon RDS 的参数组”。

在 SQL Server 2016 及更高版本上针对 Amazon RDS 支持跟踪标志 692。有关跟踪标志的更多信息，请参阅 Microsoft 文档中的 DBCC TRACEON - 跟踪标志。

删除 Amazon RDS for Microsoft SQL Server 数据库实例中的数据库

您可以删除在单可用区或多可用区部署中运行 Microsoft SQL Server 的 Amazon RDS 数据库实例上的数据库。要删除数据库，请使用以下命令：

```
--replace your-database-name with the name of the database you want to drop
EXECUTE msdb.dbo.rds_drop_database  N'your-database-name'
```

> **Note**
>
> 在命令中使用直单引号。智能引号会引发错误。

在您使用此过程删除数据库之后，Amazon RDS 将删除与数据库的现有连接并删除数据库的备份历史记录。

要向其他用户授予备份和还原限额，请按照以下步骤操作：

```
USE master
GO
CREATE LOGIN user1 WITH PASSWORD=N'changeThis', DEFAULT_DATABASE=master,
 CHECK_EXPIRATION=OFF, CHECK_POLICY=OFF
GO
USE msdb
GO
CREATE USER user1 FOR LOGIN user1
GO
use msdb
GO
GRANT EXECUTE ON msdb.dbo.rds_backup_database TO user1
GO
GRANT EXECUTE ON msdb.dbo.rds_restore_database TO user1
GO
```

在多可用区部署中重命名 Amazon RDS for Microsoft SQL Server 数据库

要对使用多可用区的 Microsoft SQL Server 数据库实例重命名，请使用以下过程：

1. 首先，为此数据库实例关闭多可用区。
2. 运行 `rdsadmin.dbo.rds_modify_db_name`，对数据库重命名。

3. 然后，为此数据库实例启用“多可用区镜像”或“永不停机”可用性组，将它恢复为原始状态。

有关更多信息，请参阅“向 Microsoft SQL Server 数据库实例添加多可用区”。

> Note
> 如果实例未使用多可用区，则无需在运行 `rdsadmin.dbo.rds_modify_db_name` 之前或之后更改任何设置。

示例：在以下示例中，存储的 `rdsadmin.dbo.rds_modify_db_name` 过程会将数据库由 **MOO** 重命名为 **ZAR**。这与运行 DDL `ALTER DATABASE [MOO] MODIFY NAME = [ZAR]` 语句类似。

```
EXEC rdsadmin.dbo.rds_modify_db_name N'MOO', N'ZAR'
GO
```

重置 Amazon RDS for SQL Server 主用户的 db_owner 角色成员资格

如果您将主用户锁定在 RDS for SQL Server 数据库上的 `db_owner` 角色成员资格之外，并且没有其他数据库用户可以授予该成员资格，则可以通过修改数据库实例主用户密码来还原丢失的成员资格。

通过更改数据库实例的主用户密码，RDS 可以向 `db_owner` 授予针对数据库实例中数据库的可能已被意外撤销的成员资格。您可以使用 Amazon RDS 控制台、Amazon CLI 命令 modify-db-instance，或者使用 ModifyDBInstance API 操作更改数据库实例密码。有关修改数据库实例的更多信息，请参阅 修改 Amazon RDS 数据库实例。

还原 Amazon RDS for SQL Server 的许可证终止的数据库实例

Microsoft 要求一些未报告其 Microsoft 许可证移动性信息的 Amazon RDS 客户终止其数据库实例。Amazon RDS 会获取这些数据库实例的快照，您可以从快照还原到包含许可证模型的新数据库实例。

您可以从标准版快照还原到标准版或企业版。

您可以从企业版快照还原到标准版或企业版。

要在 Amazon RDS 创建您的实例的最终快照后从 SQL Server 快照还原，请执行以下操作：

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择快照。
3. 选择 SQL Server 数据库实例的快照。Amazon RDS 会创建数据库实例的最终快照。已终止的实例快照的名称采用格式 `instance_name-final-snapshot`。例如，如果数据库实例名称是 **`mytest.cdxgahslksma.us-east-1.rds.com`**，那么最终快照的名称将为 **`mytest-final-snapshot`**，并位于与原始数据库实例相同的 Amazon 区域。
4. 对于 Actions (操作)，选择 Restore Snapshot (还原快照)。

   将显示 Restore DB Instance 窗口。
5. 对于 License Model (许可模式)，选择 license-included (附带许可)。
6. 选择要使用的 SQL Server 数据库引擎。
7. 对于 DB Instance Identifier (数据库实例标识符)，请输入还原后的数据库实例的名称。
8. 选择 Restore DB Instance。

有关从快照还原的更多信息，请参阅[还原到数据库实例]。

将 Amazon RDS for SQL Server 数据库从脱机状态转换为联机状态

您可以将 Amazon RDS 数据库实例上的 Microsoft SQL Server 数据库从 `OFFLINE` 转换为 `ONLINE` 状态。

SQL Server 方法	Amazon RDS 方法
ALTER DATABASE *db_name* SET ONLINE;	EXEC rdsadmin.dbo.rds_set_database_online *db_name*

在 Amazon RDS for SQL Server 上使用变更数据捕获

Amazon RDS 支持运行 Microsoft SQL Server 的数据库实例的更改数据捕获 (CDC)。CDC 可捕获对表中的数据进行的更改。它存储有关每个更改的元数据，您稍后可以访问这些元数据。有关 CDC 的工作原理的更多信息，请参阅 Microsoft 文档中的[更改数据捕获]。在将 CDC 用于您的 Amazon RDS 数据库实例之前，请通过运行 `msdb.dbo.rds_cdc_enable_db` 在数据库中启用它。启用 CDC 后，具备该数据库的 `db_owner` 身份的任何用户都可以在该数据库中的表上启用或禁用 CDC。

> **Important**
>
> 在还原期间，CDC 将被禁用。所有相关元数据都将自动从数据库中删除。这适用于快照还原、时间点还原和从 S3 还原 SQL Server Native。执行这些类型的还原之一后，您可以重新启用 CDC 并重新指定要跟踪的表。

要为数据库实例启用 CDC，请运行 `msdb.dbo.rds_cdc_enable_db` 存储过程。

```
exec msdb.dbo.rds_cdc_enable_db 'database_name'
```

要为数据库实例禁用 CDC，请运行 `msdb.dbo.rds_cdc_disable_db` 存储过程。

```
exec msdb.dbo.rds_cdc_disable_db 'database_name'
```

要向用户授予 CDC 权限，请使用以下过程：

```
go
  GRANT EXECUTE ON msdb.dbo.rds_cdc_enable_db TO User1
  GRANT EXECUTE ON msdb.dbo.rds_cdc_disable_db TO User1
```

主题

- 使用更改数据捕获跟踪表
- 更改数据捕获作业
- 多可用区实例的更改数据捕获

使用更改数据捕获跟踪表

在数据库上启用 CDC 后，您可以开始跟踪特定表。通过运行 sys.sp_cdc_enable_table，您可以选择要跟踪的表。

```
--Begin tracking a table
exec sys.sp_cdc_enable_table
   @source_schema           = N'source_schema'
,  @source_name             = N'source_name'
,  @role_name               = N'role_name'

--The following parameters are optional:
```

```
--, @capture_instance        = 'capture_instance'
--, @supports_net_changes    = supports_net_changes
--, @index_name              = 'index_name'
--, @captured_column_list    = 'captured_column_list'
--, @filegroup_name          = 'filegroup_name'
--, @allow_partition_switch = 'allow_partition_switch'
;
```

要查看您的表的 CDC 配置，请运行 sys.sp_cdc_help_change_data_capture。

```
--View CDC configuration
exec sys.sp_cdc_help_change_data_capture

--The following parameters are optional and must be used together.
--  'schema_name', 'table_name'
;
```

有关 CDC 表、函数和 SQL Server 文档中的存储过程的更多信息，请参阅以下内容：

- 更改数据捕获存储过程 (Transact-SQL)
- 更改数据捕获函数 (Transact-SQL)
- 更改数据捕获表 (Transact-SQL)

更改数据捕获作业

当您启用 CDC 时，SQL Server 会创建 CDC 作业。数据库所有者 (db_owner) 可以查看、创建、修改和删除 CDC 作业。但是，RDS 系统账户拥有这些作业。因此，这些作业从本机视图、过程或 SQL Server Management Studio 中不可见。

要控制 CDC 在数据库中的行为，请使用本机 SQL Server 过程，例如 sp_cdc_enable_table 和 sp_cdc_start_job。要更改 CDC 作业参数（例如 `maxtrans` 和 `maxscans`），您可以使用 sp_cdc_change_jobs。

要获取有关 CDC 作业的更多信息，您可以查询以下动态管理视图：

- sys.dm_cdc_errors
- sys.dm_cdc_log_scan_sessions
- sysjobs

- sysjobhistory

多可用区实例的更改数据捕获

如果您在多可用区实例上使用 CDC，请确保镜像的 CDC 作业配置与主体上的配置匹配。CDC 作业映射到 database_id。如果辅助副本上的数据库 ID 与主体不同，则作业不会与正确的数据库关联。要尝试阻止故障转移后发生错误，RDS 将删除并在新的主体上重新创建作业。重新创建的作业使用故障转移之前主体记录的参数。

尽快此过程快速运行，CDC 作业仍然可能在 RDS 更正它们之前运行。以下是在主副本和辅助副本之间使参数保持一致的三种方式：

- 对启用了 CDC 的所有数据库使用相同作业参数。
- 在更改 CDC 作业配置之前，将多可用区实例转换为单可用区。
- 只要您在主体上更改参数，就手动传输参数。

查看并定义故障转移后用于重新创建 CDC 作业的 CDC 参数，使用 rds_show_configuration 和 rds_set_configuration。

以下示例将返回 cdc_capture_maxtrans 的值。对于设置为 RDS_DEFAULT 的任何参数，RDS 会自动配置该值。

```
-- Show configuration for each parameter on either primary and secondary replicas.
exec rdsadmin.dbo.rds_show_configuration 'cdc_capture_maxtrans';
```

要在辅助副本上设置配置，请运行 rdsadmin.dbo.rds_set_configuration。此过程将设置辅助服务器上所有数据库的参数值。这些设置仅在故障转移后使用。以下示例将所有 CDC 捕获作业的 maxtrans 设置为 *1000*：

```
--To set values on secondary. These are used after failover.
exec rdsadmin.dbo.rds_set_configuration 'cdc_capture_maxtrans', 1000;
```

要在委托人上设置 CDC 作业参数，请改用 sys.sp_cdc_change_job。

使用 SQL Server Agent for Amazon RDS

借助 Amazon RDS，您可以在运行 Microsoft SQL Server 企业版、标准版或 Web 版的数据库实例上使用 SQL Server Agent。SQL Server Agent 是一项 Microsoft Windows 服务，可运行计划的管理任

务，即所谓的作业。您可以使用 SQL Server Agent 运行 T - SQL 作业，以在 SQL Server 数据库实例中重新生成索引、运行损坏检查和聚合数据。

创建 SQL Server 数据库实例时，主用户以 `SQLAgentUserRole` 角色登记。

SQL Server Agent 可以按照时间表运行作业，对特定事件或需求做出响应。有关详细信息，请参阅 Microsoft 文档中的 [SQL Server Agent]。

> **Note**
>
> 避免在数据库实例维护和备份 Windows 期间计划运行作业。由 Amazon 启动的维护和备份过程可能会中断作业或导致作业取消。
> 在多可用区部署中，启用作业复制功能时，SQL Server Agent 作业将从主要主机复制到辅助主机。有关更多信息，请参阅 启用 SQL Server Agent 作业复制。
> 多可用区部署具有 10000 个 SQL Server 代理任务的限制。如果您需要更高的限制，则可联系 Amazon Web Services 支持 请求增加限制。打开 Amazon Web Services 支持 Center (Amazon Web Services Support 中心) 页面，登录（如有必要），然后选择 Create case (创建案例)。选择 Service Limit increase (提高服务限制)。填写并提交表格。

要查看 SQL Server Management Studio (SSMS) 中单个 SQL Server Agent 作业的历史记录，需打开 Object Explorer，右键单击此作业，然后选择 View History。

SQL Server Agent 正在数据库实例中的托管主机上运行，因此不支持某些操作：

- 不支持使用 ActiveX、Windows Command Shell或 Windows PowerShell 运行复制作业和命令行脚本。
- 无法手动启动、停止或重新启动 SQL Server Agent。
- 无法从数据库实例获取通过 SQL Server Agent 发出的电子邮件通知。
- 不支持 SQL Server Agent 提示和运算符。
- 不支持使用 SQL Server Agent 创建备份。用 Amazon RDS 备份数据库实例。
- 目前，RDS for SQL Server 不支持使用 SQL Server Agent 令牌。

启用 SQL Server Agent 作业复制

您可以使用以下存储的过程启用 SQL Server Agent 作业复制：

```
EXECUTE msdb.dbo.rds_set_system_database_sync_objects @object_types = 'SQLAgentJob';
```

您可以在 Amazon RDS for SQL Server 支持的所有 SQL Server 版本上运行存储的过程。将复制以下类别中的作业：

- [未分类（本地）]
- [未分类（多服务器）]
- [未分类]
- 数据收集器
- 数据库引擎优化顾问
- 数据库维护
- 全文

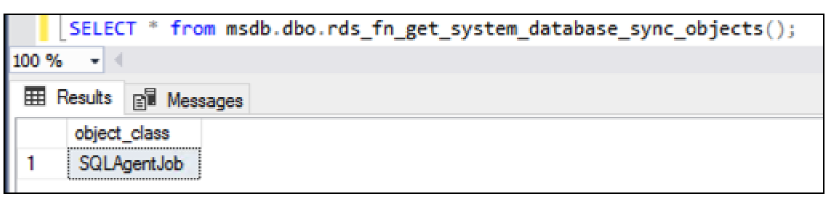
仅复制使用 T-SQL 作业步骤的作业。不复制步骤类型为 SQL Server Integration Services（SSIS）、SQL Server Reporting Services（SSRS）、复制和 PowerShell 等的作业。不复制使用数据库邮件和服务器级别对象的作业。

> **Important**
>
> 主要主机是复制的信任源。在开启作业复制之前，请确保您的 SQL Server Agent 任务位于主要主机上。否则，如果在辅助主机上使用较新的任务时开启该功能，则可能会导致删除 SQL Server Agent 任务。

您可以使用以下函数来确认复制是否已开启。

```
SELECT * from msdb.dbo.rds_fn_get_system_database_sync_objects();
```

如果正在复制 SQL Server Agent 作业，T-SQL 查询将返回以下内容。如果未复制这些作业，将不会返回任何 `object_class` 内容。

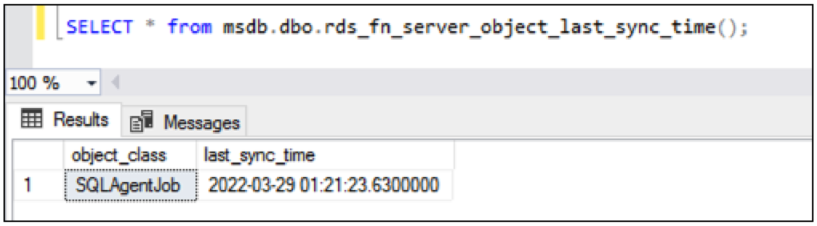


您可以使用以下函数来查找上次同步对象的时间，以 UTC 时间表示。

```
SELECT * from msdb.dbo.rds_fn_server_object_last_sync_time();
```

例如，假设您在 01:00 修改 SQL Server Agent 作业。您预计最近的同步时间将在 01:00 之后，这表示同步已经进行。

同步后，为辅助节点上的 date_created 和 date_modified 返回的值预计将匹配。



如果您也在使用 tempdb 复制，则可以通过在 @object_type 参数中提供 SQL 代理任务和 tempdb 配置来为它们启用复制：

```
EXECUTE msdb.dbo.rds_set_system_database_sync_objects @object_types =
 'SQLAgentJob,TempDbFile';
```

有关 tempdb 复制的更多信息，请参阅多可用区部署的 TempDB 配置。

SQL Server Agent 角色

RDS for SQL Server 支持以下 SQL Server Agent 角色，这些角色具有不同的权限级别来管理作业：

- SQLAgentUserRole

  权限

  - 创建和管理自己的作业、计划和操作员
  - 查看自己的作业和计划的属性
  - 无法查看或管理由其他用户创建的作业

  此角色适用于需要创建和管理自己的作业，但不需要访问由其他用户创建的作业的用户。
- SQLAgentReaderRole

  权限

  - SQLAgentUserRole 的所有权限
  - 查看所有作业和计划（包括其他用户创建的作业和计划）的列表
  - 查看所有作业的属性

  - 查看作业历史记录

  此角色适用于需要监控所有作业的状态但不需要管理这些作业的用户。

- SQLAgentOperatorRole

  权限

  - SQLAgentUserRole 和 SQLAgentReaderRole 的所有权限
  - 执行、停止或启动作业
  - 管理作业历史记录
  - 启用/禁用作业和计划
  - 查看操作员和代理

  此角色提供最全面的权限，适合需要完全控制所有作业的用户。

使用以下命令将角色分配给您的 SQL Server 登录名：

```
USE msdb;
EXEC sp_addrolemember 'SQLAgentOperatorRole', 'username';
```

在 RDS for SQL Server 中管理 SQLAgentOperatorRole

要查看当前作业，必须将 SQLAgentOperatorRole 添加到 SQL Server 登录名，并在断开与数据库的连接之前将其移除。

要在 SQL Server Management Studio 中可视化 SQL Server Agent 树，请按照以下说明操作：

在 SQL Server Management Studio（SSMS）上查看 SQL Server Agent

1. 使用 RDS 主凭证，登录 RDS SQL Server 实例并向所需用户授予 SQLAgentUserRole。

   ```
   USE msdb
   GO
   IF NOT EXISTS(SELECT name FROM sys.database_principals WHERE name = 'UserName')
   BEGIN
   CREATE USER UserName FROM LOGIN UserName
   END
   GO
   ALTER ROLE SQLAgentUserRole ADD MEMBER UserName
   GO
   ```

```
GRANT ALTER ON ROLE::[SQLAgentOperatorRole] to UserName
GO
```

如果该用户不存在，则这些命令会在 msdb 数据库上创建该用户。它还会在 SQLAgentUserRole 上添加用户，因此可以看到 SSMS 上的 SQL Server Agent 树。最后，它向用户授予对 SQLAgentOperatorRole 的更改权限。这可让用户在该角色中添加/移除自身。

2. 要将自己添加到上述角色中，请使用需要查看作业的用户连接到 RDS SQL Server 实例，然后运行以下脚本。

```
use msdb
go
ALTER ROLE SQLAgentOperatorRole ADD MEMBER UserName
GO
```

之后，右键单击作业文件夹，然后选择刷新。

3. 执行此操作时，作业选项卡会显示一个 +（加号）按钮。单击可展开 SQL Server Agent 作业列表。

4.

> **Important**
>
> 在断开与 RDS SQL Server 实例的连接之前，您需要将自己从 SQLAgentOperatorRole 中移除。

要从 SQLAgentOperatorRole 中移除登录名，请在断开或关闭 Management Studio 之前运行以下查询：

```
USE msdb
GO
ALTER ROLE SQLAgentOperatorRole DROP MEMBER UserName
GO
```

有关更多信息，请参阅 [Leveraging SQLAgentOperatorRole in RDS SQL Server](#)。

将用户添加到 SQLagentUser 角色

要允许其他登录名或用户使用 SQL Server Agent，请以主用户身份登录并执行以下操作：

1. 使用 `CREATE LOGIN` 命令创建另一个服务器级别的登录名。

2. 使用 msdb 命令在 CREATE USER 内创建一个用户，然后将此用户链接到上一步骤中创建的登录名。
3. 使用 SQLAgentUserRole 系统存储过程将用户添加到 sp_addrolemember。

例如，假设您的主用户名为 **admin**，并且您要授予一个用户对 SQL Server Agent 的访问权限，此用户的用户名为 **theirname**，密码为 **theirpassword**。在这种情况下，您可以使用以下过程。

要将用户添加到 SQLagentUser 角色

1. 以主用户身份登录。
2. 运行以下命令：

```
--Initially set context to master database
USE [master];
GO
--Create a server-level login named theirname with password theirpassword
CREATE LOGIN [theirname] WITH PASSWORD = 'theirpassword';
GO
--Set context to msdb database
USE [msdb];
GO
--Create a database user named theirname and link it to server-level login
 theirname
CREATE USER [theirname] FOR LOGIN [theirname];
GO
--Added database user theirname in msdb to SQLAgentUserRole in msdb
EXEC sp_addrolemember [SQLAgentUserRole], [theirname];
```

删除 SQL Server Agent 作业

您可以使用 sp_delete_job 存储过程删除 Amazon RDS for Microsoft SQL Server 上的 SQL Server Agent 作业。

不能使用 SSMS 删除 SQL Server Agent 作业。如果您尝试这样做，就会收到类似于以下内容的错误消息：

```
The EXECUTE permission was denied on the object 'xp_regread', database
 'mssqlsystemresource', schema 'sys'.
```

作为托管服务，RDS 在运行访问 Windows 注册表的过程时受到了限制。使用 SSMS 时，其会尝试运行没有 RDS 授权的进程 (`xp_regread`)。

> **Note**
>
> 在 RDS for SQL Server 上，只允许 sysadmin 角色的成员更新或删除由其他登录账户拥有的任务。有关更多信息，请参阅 Leveraging SQLAgentOperatorRole in RDS SQL Server。

要删除 SQL Server Agent 作业

- 运行以下 T-SQL 语句：

```
EXEC msdb..sp_delete_job @job_name = 'job_name';
```

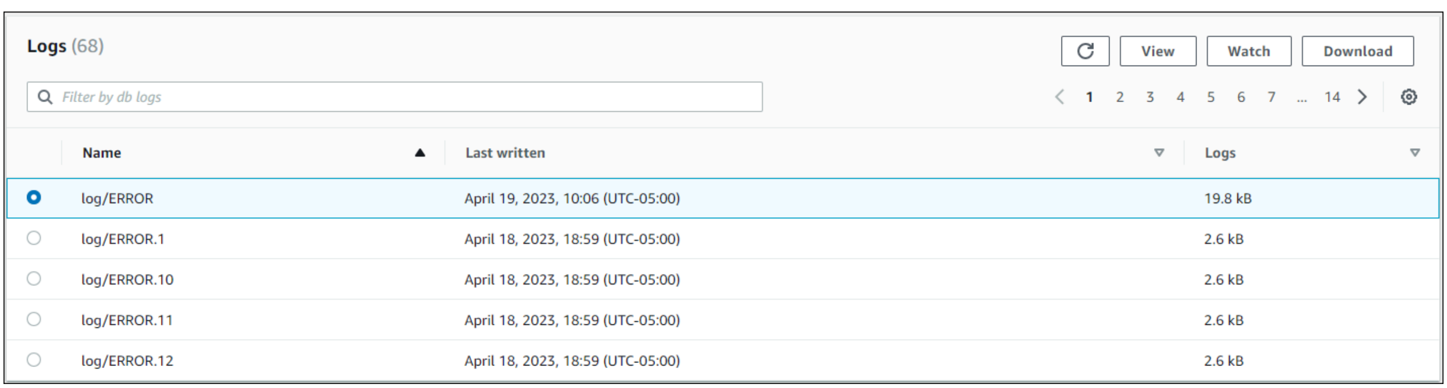
使用 Amazon RDS for Microsoft SQL Server 日志

您可使用 Amazon RDS 控制台来查看、监视和下载 SQL Server Agent 日志、Microsoft SQL Server 错误日志以及 SQL Server Reporting Services（SSRS）日志。

监视日志文件

如果在 Amazon RDS 控制台中查看日志，则可查看该日志中的实时内容。通过在控制台中监视日志，可以打开日志并使其处于动态状态，以便能实时查看该日志的更新。

仅最新日志处于活动状态以供监视。例如，假设您拥有以下所示日志：



仅主动更新 log/ERROR (最新日志)。您可选择监视其他日志，但它们是静态的且不会更新。

存档日志文件

Amazon RDS 控制台显示过去一周直至当日的日志。您可下载日志并将其存档以便在该时间后进行参考。一种对日志进行存档的方法是，将日志加载到 Amazon S3 存储桶中。有关如何设置 Amazon S3 存储桶和上传文件的说明，请参阅 Amazon Simple Storage Service 入门指南 中的 Amazon S3 基础知识，并单击开始使用。

查看错误和代理日志

要查看 Microsoft SQL Server 错误和代理日志，请将 Amazon RDS 存储过程 `rds_read_error_log` 与以下参数配合使用：

- **`@index`** – 要检索的日志的版本。默认值为 0，检索当前错误日志。指定 1 可检索上一个日志，指定 2 可检索该日志之前的日志，依此类推。
- **`@type`** – 要检索的日志的类型。指定 1 可检索错误日志。指定 2 可检索代理日志。

Example

以下示例请求当前错误日志。

```
EXEC rdsadmin.dbo.rds_read_error_log @index = 0, @type = 1;
```

有关 SQL Server 错误的更多信息，请参阅 Microsoft 文档中的 数据库引擎错误。

使用 Amazon RDS for SQL Server 的跟踪和转储文件

本部分介绍对运行 Microsoft SQL Server 的 Amazon RDS 数据库实例使用跟踪文件和转储文件。

生成跟踪 SQL 查询

```
declare @rc int
declare @TraceID int
declare @maxfilesize bigint

set @maxfilesize = 5

exec @rc = sp_trace_create @TraceID output,  0, N'D:\rdsdbdata\log\rdstest',
 @maxfilesize, NULL
```

查看打开的跟踪

```
select * from ::fn_trace_getinfo(default)
```

查看跟踪内容

```
select * from ::fn_trace_gettable('D:\rdsdbdata\log\rdstest.trc', default)
```

设置跟踪文件和转储文件的保留期

跟踪文件和转储文件会累积并占用磁盘空间。默认情况下，Amazon RDS 可清除超过 7 天的跟踪文件和转储文件。

要查看当前跟踪和转储文件保留期，请使用 `rds_show_configuration` 过程，如以下示例中所示。

```
exec rdsadmin..rds_show_configuration;
```

要修改跟踪文件的保留期，请使用 `rds_set_configuration` 过程并设置 `tracefile retention` (以分钟为单位)。以下示例将跟踪文件保留期设置为 24 小时。

```
exec rdsadmin..rds_set_configuration 'tracefile retention', 1440;
```

要修改转储文件的保留期，请使用 `rds_set_configuration` 过程并设置 `dumpfile retention` (以分钟为单位)。以下示例将转储文件保留期设置为 3 天。

```
exec rdsadmin..rds_set_configuration 'dumpfile retention', 4320;
```

出于安全考虑，您无法删除 SQL Server 数据库实例上的特定跟踪文件或转储文件。要删除所有未使用的跟踪文件或转储文件，请将这些文件的保留期设置为 0。

Amazon RDS for MySQL

Amazon RDS 支持多个 MySQL 版本的数据库实例。有关支持的版本的完整信息，请参阅 Amazon RDS 上的 MySQL 版本。

要创建 Amazon RDS for MySQL 数据库实例，请使用 Amazon RDS 管理工具或界面。然后，您可执行以下操作：

- 调整数据库实例大小
- 授权与您的数据库实例的连接
- 从备份或快照创建和还原
- 创建多可用区辅助实例
- 创建只读副本
- 监控数据库实例的性能

要存储和访问数据库实例中的数据，请使用标准 MySQL 实用程序和应用程序。

Amazon RDS for MySQL 符合很多行业标准。例如，您可以使用 RDS for MySQL 数据库来构建符合 HIPAA 标准的应用程序。您可以使用 RDS for MySQL 数据库依据与 Amazon 签订的业务合作协议 (BAA) 存储与医疗保健相关的信息，包括受保护医疗信息 (PHI)。Amazon RDS for MySQL 还满足联邦风险与授权管理项目 (FedRAMP) 安全要求。此外，在 Amazon GovCloud (US) 区域中，Amazon RDS for MySQL 获得了 FedRAMP 联合授权委员会 (JAB) 的 FedRAMP 高基准临时操作授权 (P-ATO)。有关支持的合规性标准的更多信息，请参阅Amazon云合规性。

有关每个 MySQL 版本中的功能的信息，请参阅 MySQL 文档中的 MySQL 的主要功能。

在创建数据库实例之前，完成设置 Amazon RDS 环境 中的步骤。创建数据库实例时，RDS 主用户将获得 DBA 权限，但有一些限制。此账户用于管理任务，如创建其他数据库账户。

您可执行以下操作：

- 数据库实例
- 数据库快照
- 时间点恢复
- 自动备份
- 手动备份

您可以在基于 Amazon VPC 的虚拟私有云 (VPC) 中使用运行 MySQL 的数据库实例。您还可以通过开启各种选项向 MySQL 数据库实例添加功能。Amazon RDS 支持对 MySQL 进行多可用区部署，作为一种高可用性失效转移解决方案。

> **Important**
>
> 为了提供托管服务体验，Amazon RDS 未提供数据库实例的 Shell 访问权限。并且还限制了对需要高级权限的某些系统程序和表的访问。您可以使用标准 SQL 客户端（例如 mysql 客户端）访问您的数据库。但是，您无法使用远程登录或安全外壳 (SSH) 直接访问主机。

主题

Amazon RDS 上的 MySQL 功能支持

RDS for MySQL 支持 MySQL 的大部分特性和功能。有些功能可能仅提供有限支持或受限制的权限。

您可以在数据库的新增功能？页面上筛选新的 Amazon RDS 特征。对于 Products (产品)，请选择 Amazon RDS。然后使用 **MySQL 2022** 之类的关键词进行搜索。

> **Note**
>
> 以下列表并不详尽。

主题

- Amazon RDS for MySQL 主要版本上的 MySQL 功能支持
- 支持的 RDS for MySQL 存储引擎
- 在 Amazon RDS 上通过 MySQL 使用 memcached 和其他选项
- Amazon RDS 上 MySQL 的 InnoDB 缓存预热
- 有关 RDS for MySQL 8.4 的包容性语言更改
- Amazon RDS 不支持的 MySQL 功能

Amazon RDS for MySQL 主要版本上的 MySQL 功能支持

在以下几节中，可查找有关 Amazon RDS for MySQL 主要版本上的 MySQL 功能支持的信息：

主题

- Amazon RDS 支持 MySQL 8.4

有关受支持的 Amazon RDS for MySQL 次要版本的信息，请参阅Amazon RDS 上支持的 MySQL 次要版本。

Amazon RDS 支持 MySQL 8.4

对于运行 MySQL 8.4 或更高版本的数据库实例，Amazon RDS 支持以下新功能。

- 密码库 – RDS for MySQL 将 OpenSSL 替换为已获得 FIPS 140-3 认证的 Amazon Libcrypto（Amazon-LC）。有关更多信息，请参阅 Amazon-LC GitHub 存储库（网址为 https://github.com/aws/aws-lc）。

- TLS 更改 – RDS for MySQL 仅支持 TLS 1.2 和 TLS 1.3。有关更多信息，请参阅 the section called "MySQL 的 SSL/TLS 支持"。
- Memcached 支持 – Memcached 接口在 MySQL 8.4 上不再可用。有关更多信息，请参阅 MySQL memcached 支持。
- 默认身份验证插件 - 默认身份验证插件是 `caching_sha2_password`。有关更多信息，请参阅 the section called "MySQL 默认身份验证插件"。
- **`mysqlpump`** 客户端实用工具 – MySQL 8.4 中不再提供 `mysqlpump` 客户端实用工具。有关更多信息，请参阅《Amazon 规范性指南》中的 RDS for MySQL 的基于角色的权限模型以及 mysqldump and mysqlpump。
- 托管复制存储过程 - 在使用这些存储过程管理复制时，如果其复制用户配置了 `caching_sha2_password`，则必须通过指定 `SOURCE_SSL=1` 来配置 TLS。`caching_sha2_password` 是 RDS for MySQL 8.4 的默认身份验证插件。
- 参数行为更改 - MySQL 8.4 的以下参数已更改。
  - `innodb_dedicated_server` – 此参数默认处于启用状态。有关更多信息，请参阅 在 MySQL 8.4 中配置缓冲池大小和重做日志容量。
  - `innodb_buffer_pool` – 数据库引擎现在计算此参数，但您可以覆盖此设置。有关更多信息，请参阅 在 MySQL 8.4 中配置缓冲池大小和重做日志容量。
  - `innodb_redo_log_capacity` – 此参数现在控制重做日志文件的大小。数据库引擎现在计算此参数，但您可以覆盖此设置。有关更多信息，请参阅 在 MySQL 8.4 中配置缓冲池大小和重做日志容量。
- 已弃用或删除的参数 – RDS for MySQL 从 MySQL 8.4 数据库实例的参数组中删除了以下参数。`innodb_redo_log_capacity` 参数现在控制重做日志文件的大小。
  - `innodb_log_file_size`
  - `innodb_log_files_in_group`
- 参数的新默认值 – 对于 MySQL 8.4 数据库实例，以下参数具有新的默认值：
  - 与性能相关的各种 MySQL 社区参数发生了更改。有关更多信息，请参阅 What is New in MySQL 8.4 since MySQL 8.0。

    建议您在迁移生产实例之前，在 RDS for MySQL 8.4 上测试应用程序的性能。
  - `innodb_purge_threads` – 将默认值设置为公式 `LEAST({DBInstanceVCPU/2},4)`，以防止 InnoDB 历史记录列表长度变得过大。
  - `group_replication_exit_state_action` – 默认值为 `OFFLINE_MODE`，这与 MySQL 社区中的默认值一致。有关更多信息，请参阅 RDS for MySQL 主动-主动集群的注意事项和最佳实践。

  - `binlog_format` – 默认值为 `ROW`，这与 MySQL 社区中的默认值一致。您可以修改单可用区数据库实例或多可用区数据库实例的参数，但多可用区数据库集群的参数不可修改。多可用区数据库集群使用半同步复制，当 `binlog_format` 设置为 `MIXED` 或 `STATEMENT` 时，复制将失败。
- 包容性语言更改 – RDS for MySQL 8.4 包含自 RDS for MySQL 8.0 起，与包容性语言的关键字和系统架构相关的更改。有关更多信息，请参阅 有关 RDS for MySQL 8.4 的包容性语言更改。

要查看所有 MySQL 8.4 功能和更改的列表，请参阅 MySQL 文档中的 What is New in MySQL 8.4 since MySQL 8.0。

有关不支持特征的列表，请参阅 the section called “不支持的功能”。

支持的 RDS for MySQL 存储引擎

虽然 MySQL 支持功能不同的多种存储引擎，但并非所有引擎都为恢复和数据持久性进行了优化。Amazon RDS 完全支持 MySQL 数据库实例的 InnoDB 存储引擎。Amazon RDS 的时间点恢复和快照恢复功能要求使用恢复存储引擎，且仅支持 InnoDB 存储引擎。有关更多信息，请参阅 MySQL memcached 支持。

Amazon RDS for MySQL 当前不支持联合存储引擎。

对于用户创建的架构，MyISAM 存储引擎不支持可靠的恢复，并且在恢复后重新启动 MySQL 时可能会导致数据丢失或损坏，使时间点还原或快照还原无法发挥预期的作用。不过，如果您仍选择配合使用 MyISAM 和 Amazon RDS，在某些情况下，快照可能也会有所帮助。

> **Note**
>
> `mysql` 架构中的系统表可以位于 MyISAM 存储中。

如果要将现有的 MyISAM 表转换为 InnoDB 表，您可以使用 `ALTER TABLE` 命令（例如，`alter table TABLE_NAME engine=innodb;`）。请注意，MyISAM 和 InnoDB 各有优点和缺点，所以在执行前请充分评估转换可能对您的应用程序造成的影响。

Amazon RDS 不再支持 MySQL 5.1、5.5 和 5.6。但您可以还原现有的 MySQL 5.1、5.5 和 5.6 快照。在还原 MySQL 5.1、5.5 和 5.6 快照时，数据库实例将自动升级到 MySQL 5.7。

在 Amazon RDS 上通过 MySQL 使用 memcached 和其他选项

大多数 Amazon RDS 数据库引擎均支持一些选项组，通过这些选项组，可选择其他功能用于数据库实例。RDS for MySQL 数据库实例支持 `memcached` 选项，它是一个简单的基于键的缓存。有关 `memcached` 和其他选项的更多信息，请参阅适用于 MySQL 数据库实例的选项。有关使用选项组的详细信息，请参阅使用选项组。

Amazon RDS 上 MySQL 的 InnoDB 缓存预热

InnoDB 缓存预热可为您的 MySQL DB 实例提供性能收益，方式是在数据库实例关闭时保存缓冲池的当前状态，然后在数据库实例启动时从保存的信息重新加载缓冲池。这将不再需要缓冲池使用常规数据库进行“预热”，而是预加载具有已知常见查询的页的缓冲池。存储已保存缓冲池信息的文件仅存储缓冲池内的页的元数据，而不是存储页本身。因此，此文件需要的存储空间不多。此文件的大小约为缓存大小的 0.2%。例如，对于 64 GiB 的缓存，缓存预热文件大小为 128 MiB。有关 InnoDB 缓存预热的更多信息，请参阅 MySQL 文档中的保存和还原缓冲池状态。

RDS for MySQL 数据库实例支持 InnoDB 缓存预热。要启用 InnoDB 缓存预热，请将数据库实例的参数组中的 `innodb_buffer_pool_dump_at_shutdown` 和 `innodb_buffer_pool_load_at_startup` 参数设置为 1。更改参数组中的这些参数值将影响使用该参数组的所有 MySQL 数据库实例。要为特定 MySQL 数据库实例启用 InnoDB 缓存预热，您可能需要为这些实例创建新的参数组。有关参数组的信息，请参阅Amazon RDS 的参数组。

InnoDB 缓存预热主要为使用标准存储的数据库实例提供性能收益。如果您使用 PIOPS 存储，则通常看不到显著的性能收益。

> **Important**
>
> 如果您的 MySQL 数据库实例未正常关闭（例如，在失效转移期间），则缓冲池状态将不会保存到磁盘。在此情况下，MySQL 将在重新启动数据库实例时加载任何可用的缓冲池文件。这不会造成任何损害，但还原后的缓冲池可能不会反映缓冲池在重新启动前的最新状态。要确保您拥有启动时可用于预热 InnoDB 缓存的缓冲池的最新状态，建议您定期“按需”转储缓冲池。
>
> 您可创建事件来定期自动转储缓冲池。例如，以下语句创建一个名为 `periodic_buffer_pool_dump` 的事件，该事件每小时转储一次缓冲池。
>
> ```
> CREATE EVENT periodic_buffer_pool_dump
> ON SCHEDULE EVERY 1 HOUR
> DO CALL mysql.rds_innodb_buffer_pool_dump_now();
> ```

有关 MySQL 事件的更多信息，请参阅 MySQL 文档中的事件语法。

按需转储和加载缓冲池

您可“按需”保存和加载 InnoDB 缓存。

- 要将缓冲池的当前状态转储到磁盘，请调用 mysql.rds_innodb_buffer_pool_dump_now 存储过程。
- 要从磁盘加载缓冲池的已保存状态，请调用 mysql.rds_innodb_buffer_pool_load_now 存储过程。
- 要取消正在进行的加载操作，请调用 mysql.rds_innodb_buffer_pool_load_abort 存储过程。

有关 RDS for MySQL 8.4 的包容性语言更改

RDS for MySQL 8.4 包含 MySQL 8.4 社区版本中与包容性语言的关键字和系统架构相关的更改。例如，`SHOW REPLICA STATUS` 命令替换了 `SHOW SLAVE STATUS`。

主题

- 配置参数名称更改
- 存储过程名称更改

配置参数名称更改

以下配置参数在 RDS for MySQL 8.4 中具有新名称。

为实现兼容，您可以在 `mysql` 客户端使用任何一个名称检查参数名称。当修改自定义 MySQL 8.4 参数组中的值时，只能使用新名称。有关更多信息，请参阅 原定设置和自定义参数组。

要删除的名称	新名称或首选名称
`init_slave`	`init_replica`
`log_slave_updates`	`log_replica_updates`
`log_slow_slave_statements`	`log_slow_replica_statements`

要删除的名称	新名称或首选名称	
rpl_stop_slave_timeout	rpl_stop_replica_timeout	
skip_slave_start	skip_replica_start	
slave_checkpoint_group	replica_checkpoint_group	
slave_checkpoint_period	replica_checkpoint_period	
slave_compressed_protocol	replica_compressed_protocol	
slave_exec_mode	replica_exec_mode	
slave_load_tmpdir	replica_load_tmpdir	
slave_max_allowed_packet	replica_max_allowed_packet	
slave_net_timeout	replica_net_timeout	
slave_parallel_type	replica_parallel_type	
slave_parallel_workers	replica_parallel_workers	
slave_pending_jobs_size_max	replica_pending_jobs_size_max	
slave_preserve_commit_order	replica_preserve_commit_order	
slave_skip_errors	replica_skip_errors	

要删除的名称	新名称或首选名称	
slave_sql_verify_checksum	replica_sql_verify_checksum	
slave_transaction_retries	replica_transaction_retries	
slave_type_conversions	replica_type_conversions	
sql_slave_skip_counter	sql_replica_skip_counter	

> **Note**
> 参数 `replica_allow_batching` 不可用，因为 Amazon RDS 不支持 NDB 集群。

存储过程名称更改

以下存储过程在 RDS for MySQL 8.4 中具有新名称。

为实现兼容，您可以在 RDS for MySQL 8.4 初始版本中使用任一名称。未来版本将删除旧的过程名称。有关更多信息，请参阅 配置、启动和停止二进制日志（binlog）复制。

要删除的名称	新名称或首选名称	
mysql.rds_next_master_log	mysql.rds_next_source_log	
mysql.rds_reset_external_master	mysql.rds_reset_external_source	
mysql.rds_set_external_master	mysql.rds_set_external_source	

要删除的名称	新名称或首选名称	
mysql.rds_set_external_master_with_auto_position	mysql.rds_set_external_source_with_auto_position	
mysql.rds_set_external_master_with_delay	mysql.rds_set_external_source_with_delay	
mysql.rds_set_master_auto_position	mysql.rds_set_source_auto_position	

Amazon RDS 不支持的 MySQL 功能

Amazon RDS 当前不支持以下 MySQL 功能：

- 身份验证插件
- 将错误记录到系统日志
- InnoDB 表空间加密
- NDB 集群
- 密码强度插件
- 永久保存的系统变量
- Rewriter 查询重写插件
- 半同步复制，多可用区数据库集群除外
- 可传输的表空间
- X 插件

为了提供托管服务体验，Amazon RDS 未提供数据库实例的 Shell 访问权限。并且它还限制了对需要高级权限的某些系统程序和表的访问。Amazon RDS 支持使用任何标准 SQL 客户端应用程序访问数据库实例上的数据库。Amazon RDS 不允许使用 Telnet、安全外壳（SSH）或 Windows 远程桌面连接直接从主机访问数据库实例。创建数据库实例时，系统会将您分配为该实例上所有数据库的 db_owner，并且您将具有除用于备份的权限以外的所有数据库级别权限。Amazon RDS 会为您管理备份。

Amazon RDS 上的 MySQL 版本

对于 MySQL，版本号的组织方式为版本 = X.Y.Z。在 Amazon RDS 术语中，X.Y 表示主要版本号，Z 是次要版本号。对于 Amazon RDS 实现，如果主要版本号更改，则将版本更改视为重大更改—例如，从版本 5.7 更改为 8.0。如果只有次要版本号发生更改（例如，从版本 8.0.32 更改为 8.0.34），则将版本更改视为次要更改。

主题

Amazon RDS 上支持的 MySQL 次要版本

Amazon RDS 目前支持 MySQL 的以下次要版本。

> **Note**
>
> 只有月份和年份的发布日期均为大概日期，这些日期会在确切日期明确后进行更新。
> Amazon RDS 扩展支持不适用于次要版本。

下表显示了 Amazon RDS 当前支持的 MySQL 8.4 的次要版本。

MySQL 引擎版本	社区发布日期	RDS 发布日期	RDS 标准支持终止日期
8.4.5	2025 年 4 月 15 日	2025 年 4 月 29 日	2026 年 9 月
8.4.4	2025 年 1 月 21 日	2025 年 2 月 19 日	2026 年 3 月

MySQL 引擎版本	社区发布日期	RDS 发布日期	RDS 标准支持终止日期
8.4.3	2024 年 10 月 15 日	2024 年 11 月 21 日	2026 年 3 月

下表显示了 Amazon RDS 当前支持的 MySQL 8.0 的次要版本。

> **Note**
> 次要版本可以先于主要版本终止标准支持。例如，次要版本 8.0.28 的标准支持终止日期为 2024 年 3 月 28 日，而主要版本 8.0 的标准支持终止日期为 2026 年 7 月 31 日。RDS 将支持 MySQL 社区在这两个日期之间发布的其他 8.0.* 次要版本。建议您尽快将所有主要版本升级到可用的最新次要版本。

MySQL 引擎版本	社区发布日期	RDS 发布日期	RDS 标准支持终止日期
8.0.42	2025 年 4 月 15 日	2025 年 4 月 29 日	2026 年 9 月
8.0.41	2025 年 1 月 21 日	2025 年 2 月 19 日	2026 年 3 月
8.0.40	2024 年 10 月 15 日	2024 年 11 月 13 日	2026 年 3 月
8.0.39	2024 年 7 月 23 日	2024 年 8 月 12 日	2025 年 9 月
8.0.37	2024 年 4 月 30 日	2024 年 6 月 18 日	2025 年 9 月

下表显示了 Amazon RDS 扩展支持下可用的 MySQL 5.7 的次要版本。

> **Note**
> 次要版本可能先于主要版本到达扩展支持终止日期。例如，次要版本 5.7.44-RDS.20240529 将于 2025 年 9 月到达扩展支持终止日期，而主要版本 5.7 将于 2027 年 2 月 28 日到达该日期。RDS 将在这些日期之间生成并发布其他 5.7.44-rds.xxyyzz 次要版本。建议您尽快将所有主要版本升级到可用的最新次要版本。

MySQL 引擎版本	社区发布日期	RDS 发布日期	RDS 扩展支持终止日期
5.7.44-RDS.20250508*	不适用	2025 年 5 月 20 日	2026 年 9 月
5.7.44-RDS.20250213*	不适用	2025 年 3 月 12 日	2026 年 9 月
5.7.44-RDS.20250103*	不适用	2025 年 2 月 13 日	2026 年 3 月
5.7.44-RDS.20240808*	不适用	2024 年 8 月 28 日	2025 年 9 月
5.7.44-RDS.20240529*	不适用	2024 年 6 月 25 日	2025 年 9 月
5.7.44-RDS.20240408*	不适用	2024 年 5 月 17 日	2025 年 9 月

* MySQL 社群已停用主要版本 5.7，并且不会发布新的次要版本。这是 Amazon RDS 发布的次要版本，包含适用于 MySQL 5.7 数据库的关键安全补丁和错误修复，这些补丁和错误修复在 RDS 扩展支持涵盖范围内。有关这些次要版本的更多信息，请参阅the section called “RDS 扩展支持版本”。有关 RDS 扩展支持的更多信息，请参阅Amazon RDS 的 Amazon RDS 扩展支持。

在创建新数据库实例时，您可以指定目前支持的任何 MySQL 版本。您可以指定主要版本（例如 MySQL 5.7），以及所指定主要版本的任何受支持的次要版本。如果没有指定版本，Amazon RDS 会默认使用支持的版本，通常是最新的版本。如果指定了主要版本但未指定次要版本，Amazon RDS 将默认使用您指定的主要版本的最新版本。要查看受支持版本的列表以及新建数据库实例的默认值，请运行 `describe-db-engine-versions` Amazon CLI 命令。

例如，要列出 RDS For MySQL 支持的引擎版本，请运行以下 CLI 命令：

```
aws rds describe-db-engine-versions --engine mysql --query "*[].
{Engine:Engine,EngineVersion:EngineVersion}" --output text
```

默认 MySQL 版本可能因 Amazon Web Services 区域而异。要创建具有特定次要版本的数据库实例，请在数据库实例创建过程中指定次要版本。可以通过运行以下 Amazon CLI 命令来确定 Amazon Web Services 区域的默认次要版本：

```
aws rds describe-db-engine-versions --default-only --engine mysql
 --engine-version major_engine_version --region region --query "*[].
{Engine:Engine,EngineVersion:EngineVersion}" --output text
```

将 *major_engine_version* 替换为主要引擎版本，并将 *region* 替换为 Amazon Web Services 区域。例如，以下 Amazon CLI 命令返回 5.7 主要版本和美国西部（俄勒冈州）Amazon Web Services 区域 (us-west-2) 的默认 MySQL 次要引擎版本：

```
aws rds describe-db-engine-versions --default-only --engine mysql --engine-version 5.7
 --region us-west-2 --query "*[].{Engine:Engine,EngineVersion:EngineVersion}" --output
 text
```

借助于 Amazon RDS，您就可以控制何时将 MySQL 实例升级到 Amazon RDS 支持的新主要版本。可与 MySQL 的特定版本保持兼容、对应用程序测试新版本，然后再部署到生产环境中以及在最适合计划的时间执行主要版本升级。

启用自动次要版本升级后，您的数据库实例将自动升级到 Amazon RDS 支持的新 MySQL 次要版本。此修补在计划的维护时段内进行。您可以修改数据库实例以启用或禁用自动次要版本升级。

如果选择退出自动计划的升级，您可以按照与主要版本更新相同的步骤手动升级到支持的次要版本。有关信息，请参阅升级数据库实例引擎版本。

Amazon RDS 目前支持 MySQL 数据库引擎的以下主要版本升级：

- MySQL 5.7 到 MySQL 8.0
- MySQL 8.0 到 MySQL 8.4

由于主要版本升级涉及某些兼容性风险，所以不会自行进行；您必须发出修改数据库实例的请求。您应当在升级生产实例前全面测试任何升级。有关升级 MySQL 数据库实例的信息，请参阅升级 RDS for MySQL 数据库引擎。

在版本升级前，您可以针对新的版本测试数据库实例，方法是：创建现有数据库实例的数据库快照，从该数据库快照进行还原以创建新的数据库实例，然后对新的数据库实例启动版本升级。然后，则可以在升级的数据库实例克隆上安全测试，再决定是否升级原始的数据库实例。

Amazon RDS 上的 MySQL 次要版本

有关 MySQL 社区对次要版本所做的更改，请参阅 Oracle 网站上的 Critical Patch Updates, Security Alerts and Bulletins。在关键补丁更新下，选择 Oracle 发布次要版本的月份。然后，在受影响的产品和版本下选择 MySQL 次要版本。

次要版本

MySQL 版本 8.4.5

MySQL 版本 8.4.5 现在可在 Amazon RDS 上使用。此版本包含由 MySQL 社区和 Amazon RDS 添加的修复和改进。

新功能和增强功能

- 更新了时区信息，使其基于 `tzdata2025b`。

MySQL 版本 8.4.4

MySQL 版本 8.4.4 现在可在 Amazon RDS 上使用。此版本包含由 MySQL 社区和 Amazon RDS 添加的修复和改进。

新功能和增强功能

- 更新了时区信息，使其基于 `tzdata2025a`。
- 修复了在执行 Amazon RDS 存储过程 `mysql.rds_set_configuration` 和 `mysql.rds_kill` 时导致排序规则错误的错误。

MySQL 版本 8.0.42

MySQL 版本 8.0.42 现在可在 Amazon RDS 上使用。此版本包含由 MySQL 社区和 Amazon RDS 添加的修复和改进。

新功能和增强功能

- 更新了时区信息，使其基于 `tzdata2025b`。

MySQL 版本 8.0.41

MySQL 版本 8.0.41 现在可在 Amazon RDS 上使用。此版本包含由 MySQL 社区和 Amazon RDS 添加的修复和改进。

新功能和增强功能

- 更新了时区信息，使其基于 `tzdata2025a`。
- 修复了在执行 Amazon RDS 存储过程 `mysql.rds_set_configuration` 和 `mysql.rds_kill` 时导致排序规则错误的错误。

MySQL 版本 8.0.40

MySQL 版本 8.0.40 现在可在 Amazon RDS 上使用。此版本包含由 MySQL 社区和 Amazon RDS 添加的修复和改进。

新功能和增强功能

- 修复了导致数据库升级期间字符集匹配失败的错误。

MySQL 版本 8.0.39

MySQL 版本 8.0.39 现在可在 Amazon RDS 上使用。此版本包含由 MySQL 社区和 Amazon RDS 添加的修复和改进。

新功能和增强功能

- 修复了阻止 `sql_log_off` 正确使用 `SESSION_VARIABLES_ADMIN` 权限的错误。
- 修复了阻止主用户能够向其他数据库用户授予 `SESSION_VARIABLE_ADMIN` 权限的错误。
- 修复了在执行 RDS 提供的存储过程时导致非法混合排序规则的错误。

MySQL 版本 8.0.37

MySQL 版本 8.0.37 现在可在 Amazon RDS 上使用。此版本包含由 MySQL 社区和 Amazon RDS 添加的修复和改进。

新功能和增强功能

- 修复了一个错误，该错误在执行即时 DDL 语句后执行 UPDATE 时会导致断言失败。

Amazon RDS 上支持的 MySQL 主要版本

RDS for MySQL 主要版本至少会在相应社区版本的社区生命周期终止前，在标准支持下保持可用状态。在 RDS 标准支持终止日期之后，您可以继续运行主要版本，但需付费。有关更多信息，请参阅Amazon RDS 的 Amazon RDS 扩展支持和 Amazon RDS for MySQL 定价。

您可以参照下列日期规划您的测试和升级周期。

> **Note**
> 只有月份和年份的发布日期均为大概日期，这些日期会在确切日期明确后进行更新。
> 您还可以使用 Amazon CLI 或 RDS API 查看有关主要引擎版本支持日期的信息。有关更多信息，请参阅在 Amazon RDS 扩展支持中查看引擎版本的支持日期。

MySQL 主要版本	社区发布日期	RDS 发布日期	社区生命周期终止日期	RDS 标准支持终止日期	RDS 扩展支持第 1 年定价开始日期	RDS 扩展支持第 3 年定价开始日期	RDS 扩展支持终止日期
MySQL 8.4	2024 年 4 月 30 日	2024 年 11 月 21 日	2029 年 4 月	2029 年 7 月 31 日	2029 年 8 月 1 日	2031 年 8 月 1 日	2032 年 7 月 31 日
MySQL 8.0	2018 年 4 月 19 日	2018 年 10 月 23 日	2026 年 4 月	2026 年 7 月 31 日	2026 年 8 月 1 日	2028 年 8 月 1 日	2029 年 7 月 31 日

MySQL 主要版本	社区发布日期	RDS 发布日期	社区生命周期终止日期	RDS 标准支持终止日期	RDS 扩展支持第 1 年定价开始日期	RDS 扩展支持第 3 年定价开始日期	RDS 扩展支持终止日期
MySQL 5.7*	2015 年 10 月 21 日	2016 年 2 月 22 日	2023 年 10 月	2024 年 2 月 29 日	2024 年 3 月 1 日	2026 年 3 月 1 日	2027 年 2 月 28 日

* MySQL 5.7 现已提供 RDS 扩展支持。有关更多信息，请参阅Amazon RDS 的 Amazon RDS 扩展支持。

适用于 RDS for MySQL 的 Amazon RDS 扩展支持版本

以下内容列出了适用于 RDS for MySQL 版本的所有 RDS 扩展支持发行版。

发行版

- 适用于 RDS for MySQL 版本 5.7.44-RDS.20250508 的 RDS 扩展支持
- 适用于 RDS for MySQL 版本 5.7.44-RDS.20250213 的 RDS 扩展支持
- 适用于 RDS for MySQL 版本 5.7.44-RDS.20250103 的 RDS 扩展支持
- 适用于 RDS for MySQL 版本 5.7.44-RDS.20240808 的 RDS 扩展支持
- 适用于 RDS for MySQL 版本 5.7.44-RDS.20240529 的 RDS 扩展支持
- RDS for MySQL 版本 5.7.44-RDS.20240408 的 RDS 扩展支持

适用于 RDS for MySQL 版本 5.7.44-RDS.20250508 的 RDS 扩展支持

适用于 RDS for MySQL 版本 5.7.44-RDS.20250508 的 RDS 扩展支持已发布。

已修复错误：

- 修复了当 `index_id` 大于最大 `uint32` 时回滚后虚拟索引不稳定的问题。
- 修复了测试因内存问题而失败的问题。
- 修复了 <COMMAND_CLASS> 对于 <NAME>Execute</NAME> 为空的问题。
- 修复了使用 GCC 14 [noclose 5.7] 编译 MySQL 的问题。

已修复 CVE：

- CVE-2025-30682
- CVE-2025-30687
- CVE-2025-30688
- CVE-2025-21581
- CVE-2025-21585
- CVE-2025-30689
- CVE-2025-30722
- CVE-2025-21577
- CVE-2025-30693
- CVE-2025-30695
- CVE-2025-30703

适用于 RDS for MySQL 版本 5.7.44-RDS.20250213 的 RDS 扩展支持

适用于 RDS for MySQL 版本 5.7.44-RDS.20250213 的 RDS 扩展支持已发布。

已修复错误：

- 修复了 InnoDB 失败断言 `result != FTS_INVALID`。
- 修复了在 `ALTER TABLE` 操作使用 `INPLACE` 算法重建 InnoDB 表后，空间索引崩溃和广泛损坏的问题。
- 修复了在 `ON DELETE CASCADE` 时，在 `innobase_get_computed_value` 中生成列崩溃的问题。
- 修复了 `row_MySQL_pad_col` 中的断言失败。
- 修复了在线 DDL 导致以下错误的问题：`ERROR 1712 (HY000): Index PRIMARY is corrupted`。
- 修复了 `Item_rollup_sum_switcher::current_arg` 的崩溃问题。
- 修复了在 `DROP USER` 时未刷新数据库缓存的问题。
- 修复了 `my_print_help` 中的缓冲区溢出问题。
- 修复了一个 InnoDB 问题，即 `FULLTEXT` 索引将 `FTS_DOC_ID` 限制为最大无符号 32 位值。

已修复 CVE：

- CVE-2025-21497
- CVE-2025-21555
- CVE-2025-21559
- CVE-2025-21490
- CVE-2025-21491
- CVE-2025-21500
- CVE-2025-21501
- CVE-2025-21540
- CVE-2025-21543
- CVE-2025-21520

适用于 RDS for MySQL 版本 5.7.44-RDS.20250103 的 RDS 扩展支持

适用于 RDS for MySQL 版本 5.7.44-RDS.20250103 的 RDS 扩展支持已发布。

已修复错误：

- 修复了在同一事务中删除和添加 `FULLTEXT` 索引时的 FTS 清理问题。
- 优化了 MySQL 客户端中的内存分配时间安排，以防止任何潜在的泄漏。
- 修复了使用 `UNION` 运算符时结果截断为 34 字节的问题。
- 修复了由于授权码中的 `ulong bitmask` 而可能导致的越界访问问题。

已修复 CVE：

- CVE-2024-21230
- CVE-2024-21201
- CVE-2024-21241
- CVE-2024-21203

适用于 RDS for MySQL 版本 5.7.44-RDS.20240808 的 RDS 扩展支持

适用于 RDS for MySQL 版本 5.7.44-RDS.20240808 的 RDS 扩展支持已发布。

已修复错误：

- 修复了与字典列索引相关的断言失败。
- 修复了 `is_binlog_cache_empty()` 函数的问题。
- 修复了 `sql/item.cc` 文件中的 `heap-use-after-free` 错误。
- 通过禁用空间索引进行 `index-only` 读取，修复了多个空间索引问题。
- 修复了 `LOCK_ORDER: CONNECTION_CONTROL` 插件的检测问题。
- 修复了线程被 `CONNECTION_CONTROL` 插件卡住的问题。
- 修复了未对 `PREPARED STATEMENTS` 更新 `PSI_THREAD_INFO` 的问题。
- 修复了使用 `innodb_optimize_fulltext_only` 对 FTS 索引字进行双重处理的问题。

已修复 CVE：

- [CVE-2024-21177](CVE-2024-21177)

适用于 RDS for MySQL 版本 5.7.44-RDS.20240529 的 RDS 扩展支持

适用于 RDS for MySQL 版本 5.7.44-RDS.20240529 的 RDS 扩展支持已发布。

已修复错误：

- 修复了以下错误：实施 `fix_after_pullout` 时出现 `field.cc` 断言失败。
- 修复了以下错误：向客户端返回某些 SQL 查询的元数据时出现空指针失败。这些查询在 `SELECT` 子句中包含动态参数和子查询。
- 修复了以下错误：使用 `GROUP BY` 进行松散索引扫描或扫描非连续索引范围时出现错误结果。
- 修复了以下错误：持久化期间 MySQL 发生崩溃时丢失 GTID 信息。
- 修复了可能导致 InnoDB 事务无限期挂起的争用条件。
- 修复了组复制的认证信息清理中出现的争用条件。
- 修复了并发页面操作的向后索引扫描问题。
- 修复了并发场景中的全文搜索（FTS）状态不一致的问题。
- 修复了删除表时更改缓冲区的断言问题。
- 跨所有插件类型调用 `deinit` 函数的统一行为。

已修复 CVE：

- [CVE-2024-20963](CVE-2024-20963)

- CVE-2024-20993
- CVE-2024-20998
- CVE-2024-21009
- CVE-2024-21054
- CVE-2024-21055
- CVE-2024-21057
- CVE-2024-21062
- CVE-2024-21008
- CVE-2024-21013
- CVE-2024-21047
- CVE-2024-21087
- CVE-2024-21096

RDS for MySQL 版本 5.7.44-RDS.20240408 的 RDS 扩展支持

RDS for MySQL 版本 5.7.44-RDS.20240408 的 RDS 扩展支持已可用。

此版本包含以下 CVE 的修补程序：

- CVE-2024-20963

使用数据库预览环境

2023 年 7 月，Oracle 宣布推出一款适用于 MySQL 的新版本模型。该模型包括两种类型的版本：创新版本和 LTS 版本。Amazon RDS 在 RDS 预览环境中提供 MySQL 创新版本。要了解有关 MySQL 创新版本的更多信息，请参阅 MySQL 创新和长期支持（LTS）版本简介。

数据库预览环境中的 RDS for MySQL 数据库实例在功能上类似于其他 RDS for MySQL 数据库实例。但是，您不能将数据库预览环境用于生产工作负载。

预览环境具有以下限制：

- Amazon RDS 会在创建所有数据库实例 60 天之后将其删除，包括其所有备份和快照。
- 您只能使用通用 SSD 和预置 IOPS SSD 存储。
- 您无法从 Amazon Web Services 支持 获得数据库实例的帮助。

- 您无法将数据库实例的快照复制到生产环境。

预览版支持以下选项。

- 您可以使用 db.m6i、db.r6i、db.m6g、db.m5、db.t3、db.r6g 和 db.r5 数据库实例类创建数据库实例。有关 RDS 实例类的更多信息，请参阅数据库实例类。
- 您可以同时使用单可用区和多可用区部署。
- 您可以使用标准 MySQL 转储和加载函数从数据库预览环境中导出数据库或将数据库导入数据库预览环境。

数据库预览环境中不支持的功能

以下功能在预览环境中不可用：

- 跨区域快照复制
- 跨区域只读副本
- RDS 代理

数据库预览环境中的 MySQL 版本 9.3

MySQL 版本 9.3 现可在 Amazon RDS 数据库预览环境中使用。MySQL 版本 9.3 包含一些改进，如 Changes in MySQL 9.3.0 中所述。

有关数据库预览环境的信息，请参阅the section called “数据库预览环境”。要从控制台中访问预览环境，请选择 https://console.aws.amazon.com/rds-preview/。

数据库预览环境中的 MySQL 版本 9.2

MySQL 版本 9.2 现可在 Amazon RDS 数据库预览环境中使用。MySQL 版本 9.2 包含一些改进，如 Changes in MySQL 9.2.0 中所述。

有关数据库预览环境的信息，请参阅the section called “数据库预览环境”。要从控制台中访问预览环境，请选择 https://console.aws.amazon.com/rds-preview/。

数据库预览环境中的 MySQL 版本 9.1

MySQL 版本 9.1 现可在 Amazon RDS 数据库预览环境中使用。MySQL 版本 9.1 包含一些改进，如 Changes in MySQL 9.1.0 中所述。

有关数据库预览环境的信息，请参阅the section called “数据库预览环境”。要从控制台中访问预览环境，请选择 https://console.aws.amazon.com/rds-preview/。

Amazon RDS for MySQL 已弃用的版本

Amazon RDS for MySQL 版本 5.1、5.5 和 5.6 已弃用。

有关 Amazon RDS for MySQL 的弃用策略的信息，请参阅 Amazon RDS 常见问题。

连接到 MySQL 数据库实例

在与运行 MySQL 数据库引擎的数据库实例进行连接之前，您必须首先创建数据库实例。有关信息，请参阅 创建 Amazon RDS 数据库实例。Amazon RDS 预配置数据库实例后，您就可以使用标准 MySQL 客户端应用程序或实用程序与该实例连接。在连接字符串中，指定数据库实例端点中的 DNS 地址作为主机参数，并指定数据库实例端点中的端口号作为端口参数。

要对您的 RDS 数据库实例进行身份验证，可以使用一种适用于 MySQL 的身份验证方法和 Amazon Identity and Access Management（IAM）数据库身份验证：

- 要了解如何使用一种适用于 MySQL 的身份验证方法对 MySQL 进行身份验证，请参阅 MySQL 文档中的身份验证方法。
- 要了解如何使用 IAM 数据库身份验证方法对 MySQL 进行身份验证，请参阅 适用于 MariaDB、MySQL 和 PostgreSQL 的 IAM 数据库身份验证。

可使用 MySQL 命令行客户端等工具连接 MySQL 数据库实例。有关使用 MySQL 命令行客户端的更多信息，请参阅 MySQL 文档中的 mysql - MySQL 命令行客户端。您可以使用 MySQL Workbench 这款基于 GUI 的应用程序进行连接。有关更多信息，请参阅下载 MySQL Workbench 页。有关安装 MySQL（包括 MySQL 命令行客户端）的信息，请参阅安装和升级 MySQL。

要从 Amazon VPC 外部连接到数据库实例，数据库实例必须可公开访问，必须使用数据库实例安全组的入站规则授予访问权限，并且必须满足其他要求。有关更多信息，请参阅 无法连接到 Amazon RDS 数据库实例。

您可以针对与 MySQL 数据库实例的连接使用安全套接字层 (SSL) 或传输层安全性协议 (TLS) 加密。有关信息，请参阅Amazon RDS 上 MySQL 数据库实例的 SSL/TLS 支持。如果您要使用 Amazon Identity and Access Management (IAM) 数据库身份验证，请确保使用 SSL/TLS 连接。有关信息，请参阅 适用于 MariaDB、MySQL 和 PostgreSQL 的 IAM 数据库身份验证。

您也可以从 Web 服务器连接到数据库实例。有关更多信息，请参阅“教程：创建 Web 服务器和 Amazon RDS 数据库实例”。

> **Note**
>
> 有关连接到 MariaDB 数据库实例的信息，请参阅连接到 MariaDB 数据库实例。

要查找并连接到 RDS for MySQL 数据库实例，请参阅以下主题。

主题

- 查找 RDS for MySQL 数据库实例的连接信息
- 安装 MySQL 命令行客户端
- 从 MySQL 命令行客户端连接（未加密）
- 从 MySQL Workbench 连接
- 使用适用于 MySQL 的 Amazon JDBC 驱动程序、Amazon Python 驱动程序和 Amazon ODBC 驱动程序连接到 RDS for MySQL
- 排除与 MySQL 数据库实例的连接故障

查找 RDS for MySQL 数据库实例的连接信息

数据库实例的连接信息包括其端点、端口和有效数据库用户，如主用户。例如，假设端点值为 `mydb.123456789012.us-east-1.rds.amazonaws.com`。在这种情况下，端口值是 `3306`，数据库用户是 `admin`。鉴于此信息，您可以在连接字符串中指定以下值：

- 对于主机或主机名或 DNS 名称，请指定 `mydb.123456789012.us-east-1.rds.amazonaws.com`。
- 对于端口，请指定 `3306`。
- 对于用户，请指定 `admin`。

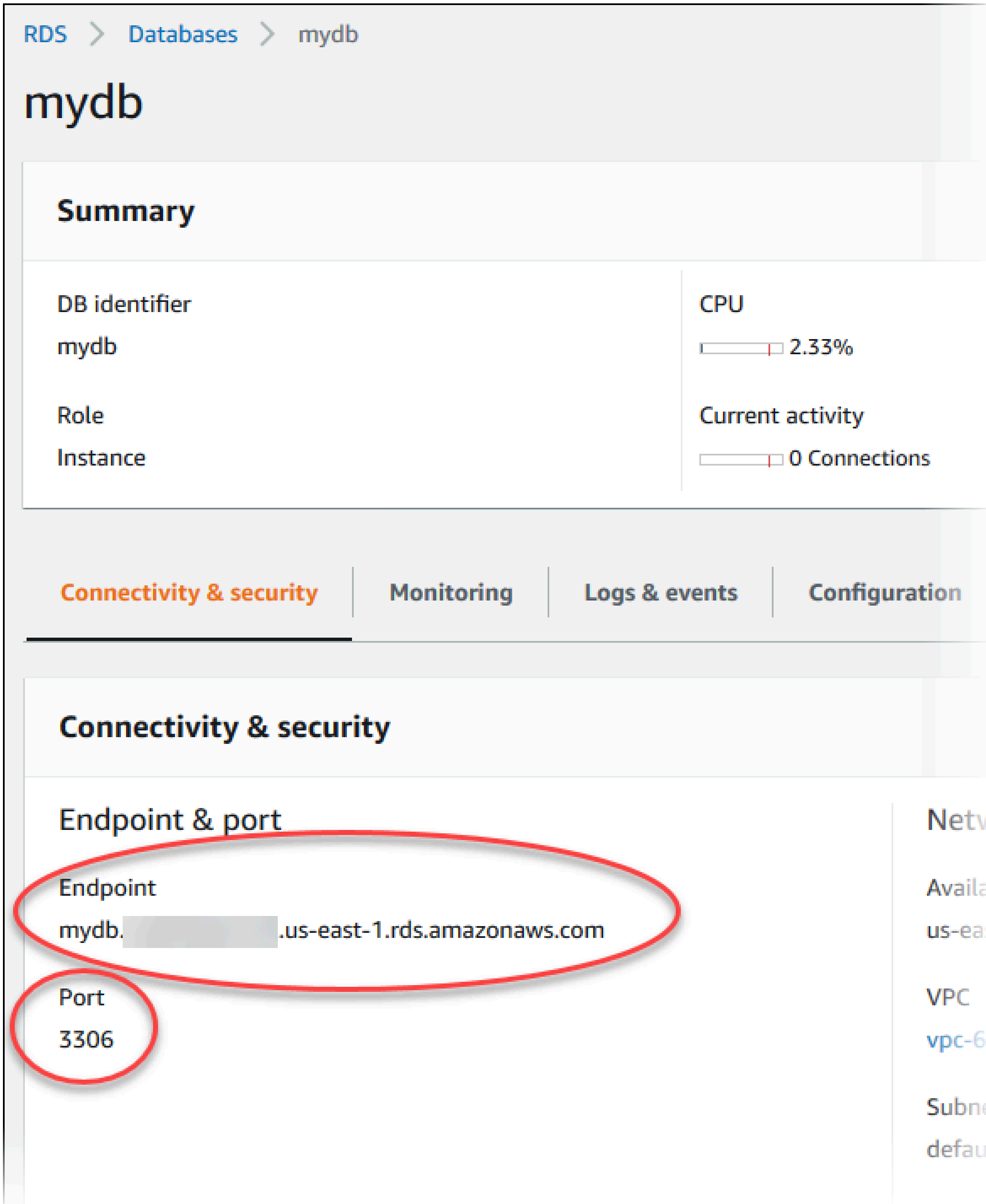
要连接到数据库实例，请使用 MySQL 数据库引擎的任何客户端。例如，您可以使用 MySQL 命令行客户端或 MySQL Workbench。

要查找数据库实例的连接信息，您可以使用Amazon Web Services Management Console、Amazon CLI describe-db-instances 命令，或 Amazon RDS API DescribeDBInstances 操作列出其详细信息。

控制台

在 Amazon Web Services Management Console 中查找数据库实例的连接信息

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 数据库 以显示数据库实例的列表。
3. 选择 MySQL 数据库实例名称以显示其详细信息。
4. 在连接和安全性选项卡上，复制端点。另请注意端口号。您需要端点和端口号才能连接到数据库实例。



5. 如果需要查找主用户名，请选择 Configuration (配置) 选项卡并查看 Master username (主用户名) 值。

Amazon CLI

要使用 Amazon CLI 查找 MySQL 数据库实例的相关连接信息，请运行 describe-db-instances 命令。在调用中，查询数据库实例 ID、端点、端口和主用户名。

对于 Linux、macOS 或 Unix：

```
aws rds describe-db-instances \
  --filters "Name=engine,Values=mysql" \
  --query "*[].[DBInstanceIdentifier,Endpoint.Address,Endpoint.Port,MasterUsername]"
```

对于 Windows：

```
aws rds describe-db-instances ^
  --filters "Name=engine,Values=mysql" ^
  --query "*[].[DBInstanceIdentifier,Endpoint.Address,Endpoint.Port,MasterUsername]"
```

您的输出应类似于以下内容。

```
[
    [
        "mydb1",
        "mydb1.123456789012.us-east-1.rds.amazonaws.com",
        3306,
        "admin"
    ],
    [
        "mydb2",
        "mydb2.123456789012.us-east-1.rds.amazonaws.com",
        3306,
        "admin"
    ]
]
```

RDS API

要使用 Amazon RDS API 查找数据库实例的连接信息，请调用 DescribeDBInstances 操作。在输出中，找到端点地址、端点端口和主用户名的值。

安装 MySQL 命令行客户端

大多数 Linux 发行版包括 MariaDB 客户端，而不是 Oracle MySQL 客户端。要在 Amazon Linux 2023 上安装 MySQL 命令行客户端，请运行以下命令：

```
sudo dnf install mariadb105
```

要在 Amazon Linux 2 上安装 MySQL 命令行客户端，请运行以下命令：

```
sudo yum install mariadb
```

要在大多数基于 DEB 的 Linux 发行版上安装 MySQL 命令行客户端，请运行以下命令：

```
apt-get install mariadb-client
```

要检查 MySQL 命令行客户端版本，请运行以下命令：

```
mysql --version
```

要阅读当前客户端版本的 MySQL 文档，请运行以下命令：

```
man mysql
```

从 MySQL 命令行客户端连接（未加密）

> **Important**
>
> 仅当客户端和服务器位于同一 VPC 中，并且网络受信任时，才会使用未加密的 MySQL 连接。有关使用加密连接的信息，请参阅 [从 MySQL 命令行客户端使用 SSL/TLS 连接到 Amazon RDS 上的 MySQL 数据库实例（已加密）]。

要使用 MySQL 命令行客户端连接到数据库实例，请在命令提示符处输入以下命令。对于参数 -h，替换为数据库实例的 DNS 名称（端点）。将参数 -P 替换为数据库实例的端口。对于 -u 参数，替换有效数据库用户（如主用户）的用户名。根据系统提示输入主用户密码。

```
mysql -h mysql-instance1.123456789012.us-east-1.rds.amazonaws.com -P 3306 -
u mymasteruser -p
```

在输入用户的密码后，您应该会看到类似于以下内容的输出。

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 9738
Server version: 8.0.28 Source distribution

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql>
```

从 MySQL Workbench 连接

若要从 MySQL Workbench 连接

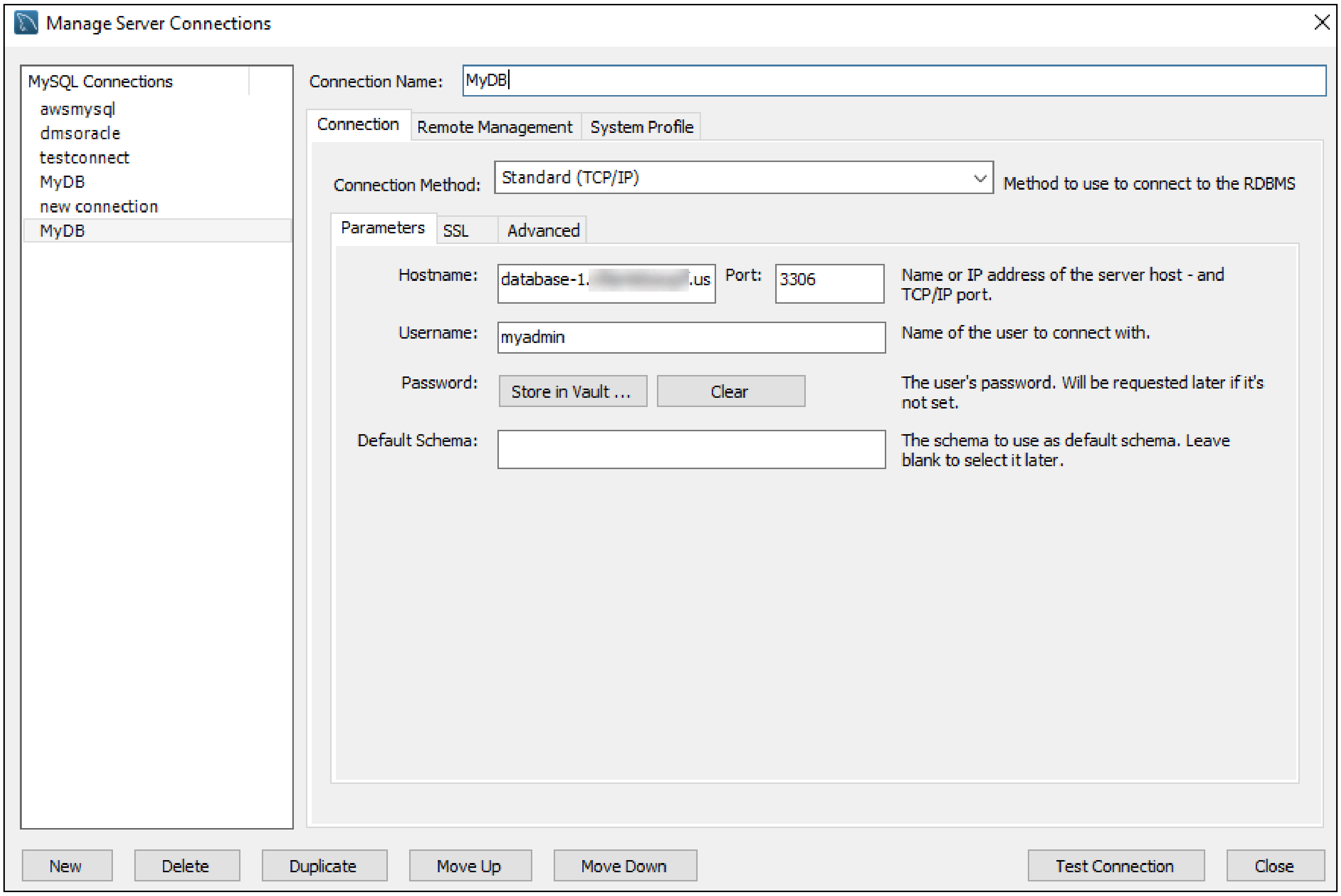
1. 在下载 MySQL Workbench 页面下载并安装 MySQL Workbench。
2. 打开 MySQL Workbench。

3. 从 Database (数据库) 中选择 Manage Connections (管理连接)。
4. 在 Manage Server Connections (管理服务器连接) 窗口中，选择 New (新建)。
5. 在 Connect to Database (连接到数据库) 窗口中，输入以下信息：

   - Stored Connection (存储连接) – 输入连接的名称，例如 **MyDB**。
   - Hostname (主机名) – 输入数据库实例的端点。
   - Port (端口) – 输入数据库实例使用的端口。
   - Username (用户名) – 输入有效数据库用户（如主用户）的用户名。

- Password (密码) –（可选）选择 Store in Vault (存储在文件库中)，然后输入并保存用户的密码。

窗口看上去类似下面这样：



您可以使用 MySQL Workbench 的功能来自定义连接。例如，您可以使用 SSL 选项卡配置 SSL/TLS 连接。有关使用 MySQL Workbench 的信息，请参阅 MySQL Workbench 文档。使用 SSL/TLS 加密客户端与 MySQL 数据库实例的连接，请参阅使用 SSL/TLS 加密客户端与 Amazon RDS 上 MySQL 数据库实例的连接。

6. （可选）选择 Test Connection (测试连接) 以确认数据库实例的连接成功。
7. 选择关闭。
8. 从 Database (数据库) 中选择 Connect to Database (连接到数据库)。
9. 从 Stored Connection (存储连接) 中选择您的连接。
10. 选择 OK。

使用适用于 MySQL 的 Amazon JDBC 驱动程序、Amazon Python 驱动程序和 Amazon ODBC 驱动程序连接到 RDS for MySQL

使用适用于 MySQL 的 Amazon JDBC 驱动程序、Amazon Python 驱动程序和 Amazon ODBC 驱动程序连接到 RDS for MySQL 数据库实例。有关更多信息，请参阅以下主题。

主题

使用 Amazon Web Services（Amazon）JDBC 驱动程序连接到 RDS for MySQL

Amazon Web Services（Amazon）JDBC 驱动程序已重新设计为高级 JDBC 包装器。此包装器是对现有 JDBC 驱动程序各项功能的补充和扩展。该驱动程序与社区 MySQL Connector/J 驱动程序和社区 MariaDB Connector/J 驱动程序兼容。

要安装 Amazon JDBC 驱动程序，请附加 Amazon JDBC 驱动程序.jar 文件（位于应用程序 `CLASSPATH` 中），并保留对相应社区驱动程序的引用。按如下方式更新相应的连接 URL 前缀：

- `jdbc:mysql://` 到 `jdbc:aws-wrapper:mysql://`
- `jdbc:mariadb://` 到 `jdbc:aws-wrapper:mariadb://`

有关 Amazon JDBC 驱动程序的更多信息及其完整使用说明，请参阅 Amazon Web Services (Amazon) JDBC Driver GitHub repository。

使用 Amazon Web Services（Amazon）Python 驱动程序连接到 RDS for MySQL

Amazon Web Services（Amazon）驱动程序设计为高级 Python 包装器。这款包装器是对开源 Psycopg 驱动程序的各项功能的补充和扩展。Amazon Python 驱动程序支持 Python 3.8 及更高版本。您可以使用 `pip` 命令和 `psycopg` 开源软件包安装 `aws-advanced-python-wrapper` 程序包。

有关 Amazon JDBC 驱动程序的更多信息及其完整使用说明，请参阅 Amazon Web Services（Amazon）Python 驱动程序 GitHub 存储库。

使用适用于 MySQL 的 Amazon Web Services（Amazon）ODBC 驱动程序连接到 RDS for MySQL

适用于 MySQL 的 Amazon ODBC 驱动程序是专为实现 RDS for MySQL 的高可用性而设计的客户端驱动程序。该驱动程序可以与 MySQL 连接器/ODBC 驱动程序并存，并且与相同的工作流程兼容。

有关适用于 MySQL 的 Amazon ODBC 驱动程序的更多信息及其完整安装和使用说明，请参阅 Amazon Web Services (Amazon) ODBC Driver for MySQL GitHub 存储库。

排除与 MySQL 数据库实例的连接故障

导致新数据库实例连接故障的两个常见原因是：

- 创建数据库实例时所用的安全组无法授权连接运行 MySQL 应用程序或实用程序的设备或 Amazon EC2 实例。数据库实例必须具有授权进行连接的 VPC 安全组。有关更多信息，请参阅 Amazon VPC 和 Amazon RDS。

  您可以在安全组中添加或编辑入站规则。对于 Source，选择 My IP。这允许从浏览器中检测到的 IP 地址访问数据库实例。
- 数据库实例是使用默认端口 3306 创建的，而您公司的防火墙规则不允许公司网络中的设备连接到该端口。要修复此故障，请重新创建使用不同端口的实例。

有关连接问题的更多信息，请参阅无法连接到 Amazon RDS 数据库实例。

保护 MySQL 数据库实例连接安全

您可以实施强有力的安全措施来保护 MySQL 数据库实例免受未经授权的访问和潜在威胁。安全组、SSL/TLS 加密和 IAM 数据库身份验证协同工作，来为 MySQL 数据库实例创建多层连接安全性。这些安全控制措施有助于您满足合规性要求，防止数据泄露，并维护应用程序和数据库之间的安全通信通道。可以通过加密传输中数据、限制对特定 IP 范围的访问以及通过 IAM 角色而不是数据库密码管理用户身份验证，来保护 MySQL 数据库实例。

MySQL 数据库实例的安全性在三个级别上进行管理：

- Amazon Identity and Access Management 控制可以对数据库实例执行 Amazon RDS 管理操作的人员。使用 IAM 证书连接到Amazon时，您的 IAM 账户必须具有授予执行 Amazon RDS 管理操作所需的权限的 IAM 策略。有关更多信息，请参阅 [Amazon RDS 的 Identity and Access Management](#)。
- 创建数据库实例时，可以使用 VPC 安全组控制哪些设备和 Amazon EC2 实例能够建立到数据库实例的端点和端口的连接。可以使用安全套接字层 (SSL)/传输层安全性协议 (TLS) 建立这些连接。此外，公司的防火墙规则也可以控制公司中运行的哪些设备可以建立到数据库实例的连接。
- 要对 MySQL 数据库实例的登录信息和权限进行身份验证，可单独或组合采用以下各种方式：
  - 您可采用与独立 MySQL 实例相同的方式。`CREATE USER`、`RENAME USER`、`GRANT`、`REVOKE` 和 `SET PASSWORD` 等命令的作用与它们在本地数据库中的作用相同，就像直接修改数据库架构表。但是，直接修改数据库架构表并不是最佳做法，而且从 RDS for MySQL 版本 8.0.36 开始，不支持直接修改数据库架构表。有关信息，请参阅 MySQL 文档中的[访问控制和账户管理](#)。
  - 您还可以使用 IAM 数据库身份验证。如果采用 IAM 数据库身份验证方式，可使用 IAM 用户或 IAM 角色以及身份验证令牌对数据库实例进行身份验证。身份验证令牌是使用签名版本 4 签名流程生成的唯一值。通过使用 IAM 数据库身份验证，您可以使用相同的凭证来控制对 Amazon 资源和数据库的访问。有关更多信息，请参阅 [适用于 MariaDB、MySQL 和 PostgreSQL 的 IAM 数据库身份验证](#)。
  - 另一个选项是 RDS for MySQL 的 Kerberos 身份验证。数据库实例与 Amazon Directory Service for Microsoft Active Directory (Amazon Managed Microsoft AD) 配合使用以启用 Kerberos 身份验证。当用户在加入信任域的 MySQL 数据库实例中进行身份验证时，将转发身份验证请求。转发的请求将转到您使用 Amazon Directory Service 创建的域目录。有关更多信息，请参阅 [对 Amazon RDS for MySQL 进行 Kerberos 身份验证](#)。

当您创建 Amazon RDS 数据库实例时，主用户有以下默认权限：

引擎版本	系统权限	数据库角色
RDS for MySQL 版本 8.4.3 及更高版本	GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, RELOAD, PROCESS, REFERENCES ,INDEX, ALTER, SHOW DATABASES , CREATE TEMPORARY TABLES, LOCK TABLES, EXECUTE, REPLICATION SLAVE , REPLICATION CLIENT , CREATE VIEW, SHOW VIEW, CREATE ROUTINE, ALTER ROUTINE, CREATE USER, EVENT, TRIGGER, CREATE ROLE, DROP ROLE, APPLICATION_PASSWORD_ADMIN , FLUSH_OPTIMIZER_COSTS , FLUSH_PRIVILEGES , FLUSH_STATUS , FLUSH_TABLES , FLUSH_USER_RESOURCES , ROLE_ADMIN , SENSITIVE_VARIABLES_OBSERVER , SESSION_VARIABLES_ADMIN , SET_ANY_DEFINER , SHOW_ROUTINE , XA_RECOVER_ADMIN	`rds_superuser_role` 有关 `rds_superuser_role` 的更多信息，请参阅RDS for MySQL 的基于角色的权限模型。
RDS for MySQL 版本 8.0.36 及更高版本	SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, RELOAD, PROCESS, REFERENCES , INDEX, ALTER, SHOW DATABASES , CREATE TEMPORARY TABLES, LOCK TABLES, EXECUTE, REPLICATION SLAVE, REPLICATION CLIENT , CREATE VIEW, SHOW VIEW, CREATE ROUTINE, ALTER ROUTINE, CREATE USER, EVENT, TRIGGER, CREATE ROLE, DROP ROLE, APPLICATION_PASSWORD_ADMIN , ROLE_ADMIN , SET_USER_ID , XA_RECOVER_ADMIN	`rds_superuser_role` 有关 `rds_superuser_role` 的更多信息，请参阅RDS for MySQL 的基于角色的权限模型。
低于 8.0.36 的 RDS for MySQL 版本	SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, RELOAD, PROCESS, REFERENCES , INDEX, ALTER, SHOW DATABASES , CREATE TEMPORARY TABLES, LOCK TABLES, EXECUTE, REPLICATION CLIENT , CREATE VIEW, SHOW VIEW, CREATE ROUTINE,	无

引擎版本	系统权限	数据库角色
	ALTER ROUTINE, CREATE USER, EVENT, TRIGGER, REPLICATION SLAVE	

> **Note**
>
> 尽管能够在数据库实例上删除主用户，但不推荐这样做。要重新创建主用户，请执行 ModifyDBInstance RDS API 操作或运行 modify-db-instance Amazon CLI 命令，并使用相应的参数指定新的主用户密码。如果实例中不存在主用户，将使用指定的密码创建主用户。

要为每个数据库实例提供管理服务，需在创建数据库实例时创建 `rdsadmin` 用户。如果试图删掉、重命名、修改密码，或者修改 `rdsadmin` 账户的权限，会导致出错。

若要允许管理数据库实例，需限制标准的 `kill` 和 `kill_query` 命令。通过所提供的 Amazon RDS 命令 `rds_kill` 和 `rds_kill_query`，您可以结束数据库实例上的用户会话或查询。

RDS for MySQL 的密码验证

MySQL 提供 `validate_password` 插件以提高安全性。该插件使用数据库参数组中的参数为您的 MySQL 数据库实例实施密码策略。运行 MySQL 5.7、8.0 和 8.4 版的数据库实例支持该插件。有关 `validate_password` 插件的更多信息，请参阅 MySQL 文档中的密码验证插件。

为 MySQL 数据库实例启用 **`validate_password`** 插件

1. 连接到 MySQL 数据库实例并运行以下命令。

   ```
   INSTALL PLUGIN validate_password SONAME 'validate_password.so';
   ```

2. 在数据库实例使用的数据库参数组中为插件配置参数。

   有关参数插件的更多信息，请参阅 MySQL 文档中的密码验证插件选项和变量。

   有关修改数据库实例参数的更多信息，请参阅在 Amazon RDS 中修改数据库参数组中的参数。
3. 重新启动数据库实例。

启用 `validate_password` 插件之后，请重置现有密码以遵循新的验证策略。

Amazon RDS 不验证密码。MySQL 数据库实例执行密码验证。如果您使用 Amazon Web Services Management Console、`modify-db-instance` Amazon CLI 命令或 `ModifyDBInstance` RDS API 操作设置用户密码，则即使新密码不符合您的密码策略，更改也会成功。但是，仅当新密码符合密码策略时，才会在 MySQL 数据库实例中设置它。在这种情况下，Amazon RDS 记录以下事件。

```
"RDS-EVENT-0067" - An attempt to reset the master password for the DB instance has failed.
```

有关 Amazon RDS 事件的更多信息，请参阅 [使用 Amazon RDS 事件通知]。

使用 SSL/TLS 加密客户端与 Amazon RDS 上 MySQL 数据库实例的连接

安全套接字层 (SSL) 是用于保护客户端和服务器之间的网络连接的行业标准协议。在 SSL 版本 3.0 之后，此名称已更改为传输层安全性协议 (TLS)。Amazon RDS 对于 MySQL 数据库实例支持 SSL/TLS 加密。使用 SSL/TLS，您可加密应用程序客户端和 MySQL 数据库实例之间的连接。SSL/TLS 支持在 MySQL 的所有 Amazon Web Services 区域中都可用。

借助 Amazon RDS，可以通过使用 SSL/TLS 加密客户端与 MySQL 数据库实例的连接，要求所有与 MySQL 数据库实例的连接都使用 SSL/TLS，以及使用 SSL/TLS（加密）从 MySQL 命令行客户端进行连接，从而保护传输中数据。以下各节提供了有关为 Amazon RDS 上的 MySQL 数据库实例配置和使用 SSL/TLS 加密的指导。

主题

Amazon RDS 上 MySQL 数据库实例的 SSL/TLS 支持

在 Amazon RDS 预置数据库实例时，Amazon RDS 创建 SSL/TLS 证书，并将该证书安装在数据库实例上。这些证书由证书颁发机构签署。SSL/TLS 证书会将数据库实例端点作为 SSL/TLS 证书的公用名 (CN) 包含在内以防止欺诈攻击。

有关下载证书的信息，请参阅 使用 SSL/TLS 加密与数据库实例或集群的连接。有关将 SSL/TLS 与 MySQL 一起使用的更多信息，请参阅更新应用程序以使用新的 SSL/TLS 证书连接到 MySQL 数据库实例。

对于 MySQL 8.0 及更低版本，Amazon RDS for MySQL 使用 OpenSSL 进行安全连接。对于 MySQL 8.4 及更高版本，Amazon RDS for MySQL 使用 Amazon-LC。TLS 支持取决于 MySQL 版本。下表显示了支持 TLS 的 MySQL 版本。

MySQL 版本	TLS 1.0	TLS 1.1	TLS 1.2	TLS 1.3
MySQL 8.4	不支持	不支持	支持	支持
MySQL 8.0	不支持	不支持	支持	支持
MySQL 5.7	支持	支持	支持	不支持

Amazon RDS 上 MySQL 数据库实例的特定用户账户需要 SSL/TLS

对于与 Amazon RDS 上的 MySQL 数据库实例的指定用户账户连接，可以要求进行 SSL/TLS 加密。保护敏感信息免遭未经授权的访问或拦截，对于在考虑数据机密性的情况下强制执行安全策略至关重要。

若要求特定用户的账户建立 SSL/TLS 连接，请使用以下语句之一（具体取决于 MySQL 版本），来要求用户账户 `encrypted_user` 建立 SSL/TLS 连接。

为此，请使用以下语句。

```
ALTER USER 'encrypted_user'@'%' REQUIRE SSL;
```

有关使用 MySQL 进行 SSL/TLS 连接的更多信息，请参阅 MySQL 文档中的使用加密连接。

所有与 Amazon RDS 上的 MySQL 数据库实例的连接都需要 SSL/TLS

使用 `require_secure_transport` 参数要求与 MySQL 数据库实例的所有用户连接都使用 SSL/TLS。默认情况下，`require_secure_transport` 参数设置为 `OFF`。您可将 `require_secure_transport` 参数设置为 `ON` 以要求使用 SSL/TLS 连接到数据库实例。

您可通过更新数据库实例的数据库参数组来设置 `require_secure_transport` 参数值。您无需重启数据库实例即可使更改生效。

当数据库实例的 require_secure_transport 参数设置为 ON 时，如果数据库客户端能够建立加密连接，则可以连接到该数据库实例。否则，将向客户端返回类似于以下内容的错误消息：

```
MySQL Error 3159 (HY000): Connections using insecure transport are prohibited while --
require_secure_transport=ON.
```

有关设置参数的信息，请参阅 在 Amazon RDS 中修改数据库参数组中的参数。

有关 require_secure_transport 参数的更多信息，请参阅 MySQL 文档。

从 MySQL 命令行客户端使用 SSL/TLS 连接到 Amazon RDS 上的 MySQL 数据库实例（已加密）

根据您使用的 MySQL 或 MariaDB 版本，mysql 客户端程序参数会略有不同。

要查看使用的版本，请使用 --version 选项运行 mysql 命令。在以下示例中，结果显示客户端程序来自 MariaDB。

```
$ mysql --version
mysql  Ver 15.1 Distrib 10.5.15-MariaDB, for osx10.15 (x86_64) using readline 5.1
```

大多数 Linux 发行版（例如 Amazon Linux、CentOS、SUSE 和 Debian）都用 MariaDB 取代了 MySQL，mysql 版本来自 MariaDB。

要使用 SSL/TLS 连接到数据库实例，请执行以下步骤：

使用 MySQL 命令行客户端连接到使用 SSL/TLS 加密的数据库实例

1. 下载适用于所有 Amazon Web Services 区域的根证书。

   有关下载证书的信息，请参阅 使用 SSL/TLS 加密与数据库实例或集群的连接。
2. 使用 MySQL 命令行客户端，通过 SSL/TLS 加密连接到数据库实例。对于 -h 参数，替换为数据库实例的 DNS 名称（端点）。对于 --ssl-ca 参数，替换为 SSL/TLS 证书文件名。对于 -P 参数，替换为数据库实例的端口。对于 -u 参数，替换为有效数据库用户（如主用户）的用户名。根据系统提示输入主用户密码。

   以下示例说明了如何在 MySQL 5.7 和更高版本中使用 --ssl-ca 参数启动客户端。

   ```
   mysql -h myinstance.123456789012.rds-cn-north-1.amazonaws.com --ssl-ca=global-
   bundle.pem --ssl-mode=REQUIRED -P 3306 -u myadmin -p
   ```

如果要求 SSL/TLS 连接对照 SSL/TLS 证书中的端点验证数据库实例端点，请输入以下命令：

```
mysql -h myinstance.123456789012.rds-cn-north-1.amazonaws.com --ssl-ca=global-bundle.pem --ssl-mode=VERIFY_IDENTITY -P 3306 -u myadmin -p
```

以下示例说明了如何在 MariaDB 客户端和更高版本中使用 `--ssl-ca` 参数启动客户端。

```
mysql -h myinstance.123456789012.rds-cn-north-1.amazonaws.com --ssl-ca=global-bundle.pem --ssl -P 3306 -u myadmin -p
```

3. 根据系统提示输入主用户密码。

您会看到类似以下内容的输出。

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 9738
Server version: 8.0.28 Source distribution

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql>
```

更新应用程序以使用新的 SSL/TLS 证书连接到 MySQL 数据库实例

自 2023 年 1 月 13 日起，Amazon RDS 发布了新的证书颁发机构（CA）证书，以便使用安全套接字层或传输层安全性协议（SSL/TLS）连接到 RDS 数据库实例。接下来，您可以找到有关更新应用程序以使用新证书的信息。

本主题可帮助您确定是否有任何客户端应用程序使用 SSL/TLS 连接到您的数据库实例。如果是这样，您可以进一步检查这些应用程序是否需要证书验证才能连接。

> **Note**
>
> 某些应用程序配置为仅在它们可以成功验证服务器上的证书时才连接到 MySQL 数据库实例。对于此类应用程序，您必须更新客户端应用程序信任存储，以包括新的 CA 证书。
>
> 您可以指定以下 SSL 模式：`disabled`、`preferred` 和 `required`。当您使用 `preferred` SSL 模式且 CA 证书不存在或不是最新时，连接会恢复为不使用 SSL，在不加密的情况下连接。

不建议使用 preferred 模式。在 preferred 模式下，如果连接遇到无效证书，则会停止使用加密并在未加密的状态下继续。

更新客户端应用程序信任存储中的 CA 证书后，可以在数据库实例上轮换这些证书。强烈建议在生产环境中实现这些过程之前，先在开发或测试环境中测试它们。

有关证书轮换的更多信息，请参阅轮换 SSL/TLS 证书。有关下载证书的更多信息，请参阅使用 SSL/TLS 加密与数据库实例或集群的连接。有关对 MySQL 数据库实例使用 SSL/TLS 的信息，请参阅Amazon RDS 上 MySQL 数据库实例的 SSL/TLS 支持。

主题

确定是否有任何应用程序使用 SSL 连接到 MySQL 数据库实例

如果您使用的是 Amazon RDS for MySQL 版本 5.7、8.0 或 8.4，并且启用了 Performance Schema，请运行以下查询来检查连接是否使用了 SSL/TLS。有关启用 Performance Schema 的信息，请参阅 MySQL 文档中的 Performance Schema 快速入门。

```
mysql> SELECT id, user, host, connection_type
       FROM performance_schema.threads pst
       INNER JOIN information_schema.processlist isp
       ON pst.processlist_id = isp.id;
```

在该示例输出中，您可以看到自己的会话 (admin) 以及作为 webapp1 登录的应用程序均在使用 SSL。

```
+----+-----------------+------------------+-----------------+
| id | user            | host             | connection_type |
+----+-----------------+------------------+-----------------+
```

```
|  8 | admin           | 10.0.4.249:42590 | SSL/TLS         |
|  4 | event_scheduler | localhost        | NULL            |
| 10 | webapp1         | 159.28.1.1:42189 | SSL/TLS         |
+----+-----------------+------------------+-----------------+
3 rows in set (0.00 sec)
```

确定客户端是否需要证书验证才能连接

您可以检查 JDBC 客户端和 MySQL 客户端是否需要证书验证才能连接。

JDBC

以下使用 MySQL Connector/J 8.0 的示例显示了一种方法，用于检查应用程序的 JDBC 连接属性以确定成功的连接是否需要有效证书。有关 MySQL 的所有 JDBC 连接选项的更多信息，请参阅 MySQL 文档中的配置属性。

当使用 MySQL Connector/J 8.0 时，如果连接属性将 `sslMode` 设置为 `VERIFY_CA` 或 `VERIFY_IDENTITY`，则 SSL 连接需要对数据库服务器证书进行验证，如以下示例所示。

```
Properties properties = new Properties();
properties.setProperty("sslMode", "VERIFY_IDENTITY");
properties.put("user", DB_USER);
properties.put("password", DB_PASSWORD);
```

> **Note**
>
> 如果您使用 MySQL Java Connector v5.1.38 或更高版本或者 MySQL Java Connector v8.0.9 或更高版本连接到数据库，即使您没有明确配置应用程序在连接到数据库时使用 SSL/TLS，这些客户端驱动程序仍默认为使用 SSL/TLS。此外，在使用 SSL/TLS 时，它们会执行部分证书验证，如果数据库服务器证书过期，则无法连接。

MySQL

以下使用 MySQL 客户端的示例显示了两种方法，用于检查脚本的 MySQL 连接以确定成功的连接是否需要有效证书。有关 MySQL 客户端的所有连接选项的更多信息，请参阅 MySQL 文档中的加密连接的客户端配置。

当使用 MySQL 客户端 5.7 及更高版本时，如果对于 `--ssl-mode` 选项，您指定 `VERIFY_CA` 或 `VERIFY_IDENTITY`，则 SSL 连接需要对服务器 CA 证书进行验证，如以下示例所示。

```
mysql -h mysql-database.rds.amazonaws.com -uadmin -ppassword --ssl-ca=/tmp/ssl-cert.pem
 --ssl-mode=VERIFY_CA
```

更新应用程序信任存储

有关更新 MySQL 应用程序的信任存储的信息，请参阅 MySQL 文档中的安装 SSL 证书。

有关下载根证书的信息，请参阅 使用 SSL/TLS 加密与数据库实例或集群的连接。

有关导入证书的示例脚本，请参阅 将证书导入信任存储的示例脚本。

> **Note**
>
> 更新信任存储时，除了添加新证书外，还可以保留较旧证书。

如果在应用程序中使用 mysql JDBC 驱动程序，请在该应用程序中设置以下属性。

```
System.setProperty("javax.net.ssl.trustStore", certs);
System.setProperty("javax.net.ssl.trustStorePassword", "password");
```

启动应用程序时，请设置以下属性。

```
java -Djavax.net.ssl.trustStore=/path_to_trust_store/MyTruststore.jks -
Djavax.net.ssl.trustStorePassword=my_trust_store_password com.companyName.MyApplication
```

> **Note**
>
> 作为安全最佳实践，请指定除此处所示提示以外的密码。

用于建立 SSL 连接的示例 Java 代码

以下代码示例展示了如何设置使用 JDBC 验证服务器证书的 SSL 连接。

```
public class MySQLSSLTest {

        private static final String DB_USER = "username";
        private static final String DB_PASSWORD = "password";
        // This trust store has only the prod root ca.
        private static final String TRUST_STORE_FILE_PATH = "file-path-to-trust-store";
        private static final String TRUST_STORE_PASS = "trust-store-password";

        public static void test(String[] args) throws Exception {
            Class.forName("com.mysql.jdbc.Driver");

            System.setProperty("javax.net.ssl.trustStore", TRUST_STORE_FILE_PATH);
            System.setProperty("javax.net.ssl.trustStorePassword", TRUST_STORE_PASS);

            Properties properties = new Properties();
            properties.setProperty("sslMode", "VERIFY_IDENTITY");
            properties.put("user", DB_USER);
            properties.put("password", DB_PASSWORD);


            Connection connection = null;
            Statement stmt = null;
            ResultSet rs = null;
            try {
                connection =
 DriverManager.getConnection("jdbc:mysql://mydatabase.123456789012.us-
east-1.rds.amazonaws.com:3306",properties);
                stmt = connection.createStatement();
                rs=stmt.executeQuery("SELECT 1 from dual");
            } finally {
                if (rs != null) {
                    try {
                        rs.close();
                    } catch (SQLException e) {
                    }
                }
                if (stmt != null) {
                   try {
                        stmt.close();
```

```
                    } catch (SQLException e) {
                   }
                }
                if (connection != null) {
                    try {
                        connection.close();
                    } catch (SQLException e) {
                        e.printStackTrace();
                    }
                }
            }
            return;
        }
    }
```

> **Important**
>
> 在确定了数据库连接使用 SSL/TLS 并更新了应用程序信任存储之后，可以更新数据库以使用 rds-ca-rsa2048-g1 证书。有关说明，请参阅 通过修改数据库实例或集群来更新 CA 证书 中的步骤 3。
>
> 作为安全最佳实践，请指定除此处所示提示以外的密码。

对 Amazon RDS for MySQL 进行 Kerberos 身份验证

当用户连接到 MySQL 数据库实例时，您可以使用 Kerberos 身份验证来验证用户的身份。数据库实例与 Amazon Directory Service for Microsoft Active Directory (Amazon Managed Microsoft AD) 配合使用以启用 Kerberos 身份验证。当用户在加入信任域的 MySQL 数据库实例中进行身份验证时，将转发身份验证请求。转发的请求将转到您使用 Amazon Directory Service 创建的域目录。

将所有凭证保存在同一目录中可以节省您的时间和精力。使用这种方法，您具有一个集中位置用于存储和管理多个数据库实例的凭证。使用目录还可以改善您的整体安全概要。

区域和版本可用性

功能可用性和支持因每个数据库引擎的特定版本以及 Amazon Web Services 区域而异。有关使用 Kerberos 身份验证的 Amazon RDS 的版本和区域可用性的更多信息，请参阅支持 Amazon RDS 中 Kerberos 身份验证功能的区域和数据库引擎。

为 MySQL 数据库实例设置 Kerberos 身份验证概述

要为 MySQL 数据库实例设置 Kerberos 身份验证，请完成以下一般步骤，稍后将详细介绍这些步骤：

1. 使用 Amazon Managed Microsoft AD 创建 Amazon Managed Microsoft AD 目录。您可以使用 Amazon Web Services Management Console、Amazon CLI 或 Amazon Directory Service 创建目录。有关执行此操作的详细信息，请参阅 Amazon Directory Service 管理指南中的创建 Amazon Managed Microsoft AD 目录。
2. 创建使用托管式 IAM 策略 `AmazonRDSDirectoryServiceAccess` 的 Amazon Identity and Access Management（IAM）角色。此角色允许 Amazon RDS 调用您的目录。

   为了让角色允许访问，Amazon Security Token Service（Amazon STS）端点必须在您的 Amazon 账户的 Amazon Web Services 区域中激活。Amazon STS 端点默认情况下在所有 Amazon Web Services 区域中保持活跃，且您无需任何进一步动作即可使用它们。有关更多信息，请参阅《IAM 用户指南》中的在 Amazon Web Services 区域中激活和停用 Amazon STS。
3. 使用 Microsoft Active Directory 工具在 Amazon Managed Microsoft AD 目录中创建和配置用户。有关在 Active Directory 中创建用户的更多信息，请参阅 Amazon 管理指南中的在 Amazon Directory Service 托管式 Microsoft AD 中管理用户和组。
4. 创建或修改 MySQL 数据库实例。如果您在创建请求中使用 CLI 或 RDS API，请使用 `Domain` 参数指定域标识符。使用在创建目录时生成的 `d-*` 标识符和您创建的角色的名称。

   如果您将现有 MySQL 数据库实例修改为使用 Kerberos 身份验证，请为数据库实例设置域和 IAM 角色参数。在与域目录相同的 VPC 中查找数据库实例。
5. 使用 Amazon RDS 主用户凭证连接到 MySQL 数据库实例。使用 `CREATE USER` 子句 `IDENTIFIED WITH 'auth_pam'` 在 MySQL 中创建用户。您以此方式创建的用户可以使用 Kerberos 身份验证登录到 MySQL 数据库实例。

为 MySQL 数据库实例设置 Kerberos 身份验证

您可以使用 Amazon Managed Microsoft AD 为 MySQL 数据库实例设置 Kerberos 身份验证。要设置 Kerberos 身份验证，请执行以下步骤。

步骤 1：使用 Amazon Managed Microsoft AD 创建目录

Amazon Directory Service 将在 Amazon 云中创建一个完全托管的 Active Directory。创建 Amazon Managed Microsoft AD 目录时，Amazon Directory Service 将代表您创建两个域控制器和域名系统 (DNS) 服务器。目录服务器在 VPC 中的不同子网中创建。这种冗余有助于确保始终可以访问目录，即使发生了故障。

创建 Amazon Managed Microsoft AD 目录时，Amazon Directory Service 代表您执行以下任务：

- 在 VPC 中设置 Active Directory。
- 使用用户名 Admin 和指定密码创建目录管理员账户。您可以使用此账户管理您的目录。

> **Note**
> 请务必保存此密码。Amazon Directory Service 不会存储它。您可以重置它，但无法检索它。

- 为目录控制器创建安全组。

在启动 Amazon Managed Microsoft AD 时，Amazon 创建一个组织单位 (OU)，其中包含目录的所有对象。此 OU 具有您在创建目录时键入的 NetBIOS 名称且位于域根目录中。此域根目录由 Amazon 拥有和管理。

使用您的 Amazon Managed Microsoft AD 目录创建的 Admin 账户对您的 OU 的最常见管理活动具有以下权限：

- 创建、更新或删除用户
- 将资源添加到域（如文件或打印服务器），然后为 OU 中的用户分配这些资源的权限
- 创建额外的 OU 和容器
- 委托授权
- 从 Active Directory 回收站还原删除的对象
- 在 Active Directory Web 服务上运行 AD 和 DNS Windows PowerShell 模块

Admin 账户还具有执行下列域范围活动的权限：

- 管理 DNS 配置（添加、删除或更新记录、区域和转发器）
- 查看 DNS 事件日志
- 查看安全事件日志

使用 Amazon Managed Microsoft AD 创建目录

1. 登录 Amazon Web Services Management Console，然后打开 Amazon Directory Service 控制台，网址为：https://console.aws.amazon.com/directoryservicev2/。

2. 在导航窗格中，选择 Directories (目录)，然后选择 Set up Directory (设置目录)。
3. 选择 Amazon Managed Microsoft AD。Amazon Managed Microsoft AD 是当前唯一可以与 Amazon RDS 一起使用的选项。
4. 输入以下信息：

   目录 DNS 名称

   　目录的完全限定名称，例如 **`corp.example.com`**。

   目录 NetBIOS 名称

   　目录的短名称，如 **`CORP`**。

   目录描述

   　（可选）目录的描述。

   管理员密码

   　目录管理员的密码。目录创建过程将使用用户名 Admin 和此密码创建一个管理员账户。

   　目录管理员密码不能包含单词“admin”。此密码区分大小写，且长度必须介于 8 – 64 个字符之间。至少，它还必须包含下列四种类别中三种类别的一个字符：

   - 小写字母 (a–z)
   - 大写字母 (A–Z)
   - 数字 (0–9)
   - 非字母数字字符 (~!@#$%^&*_-+=`|\(){}[]:;"'<>,.?/)

   确认密码

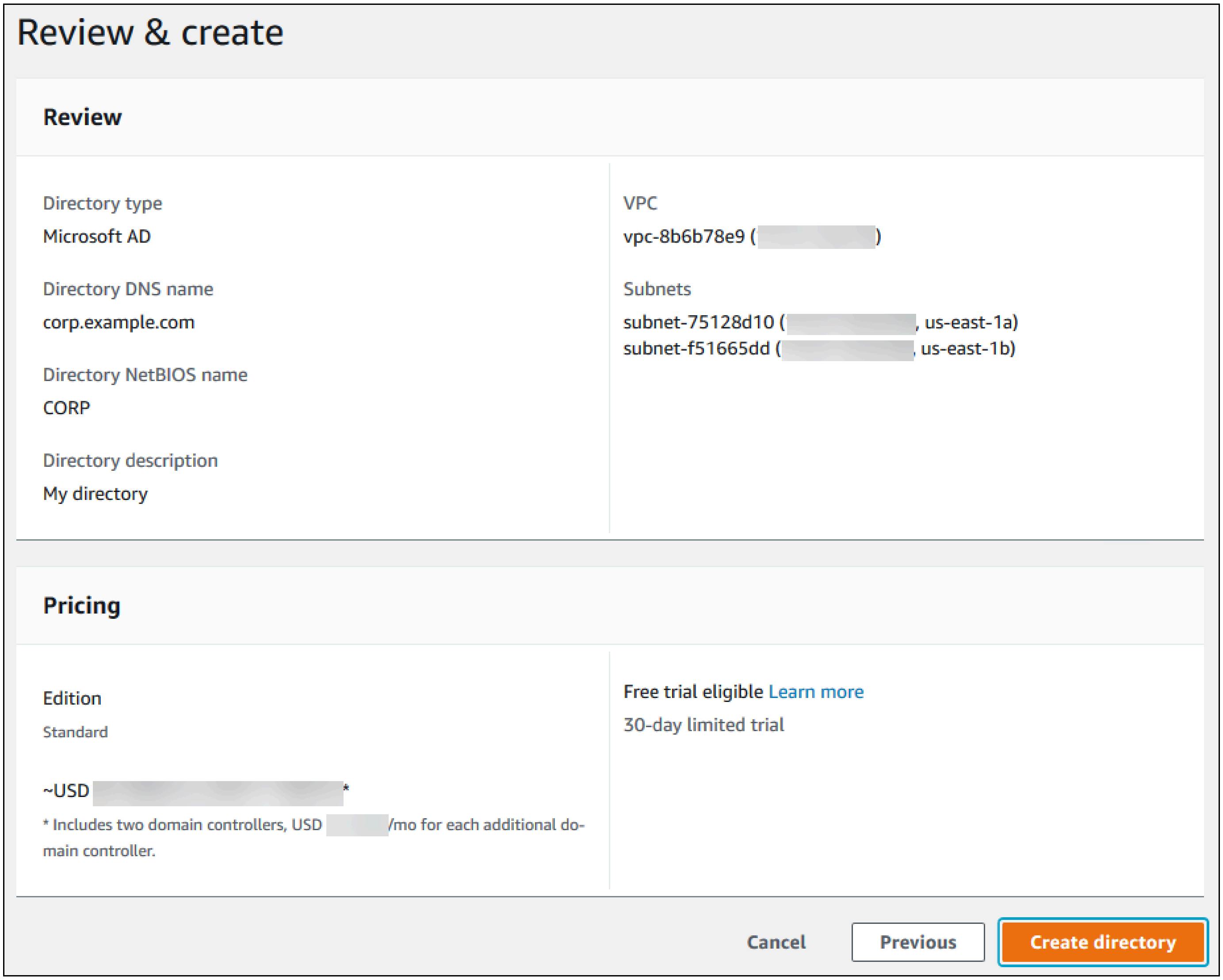
   　已重新键入管理员密码。
5. 选择下一步。
6. 在 Networking (网络) 部分中输入以下信息，然后选择 Next (下一步)：

   VPC

   　目录的 VPC。在同一 VPC 中创建 MySQL 数据库实例。

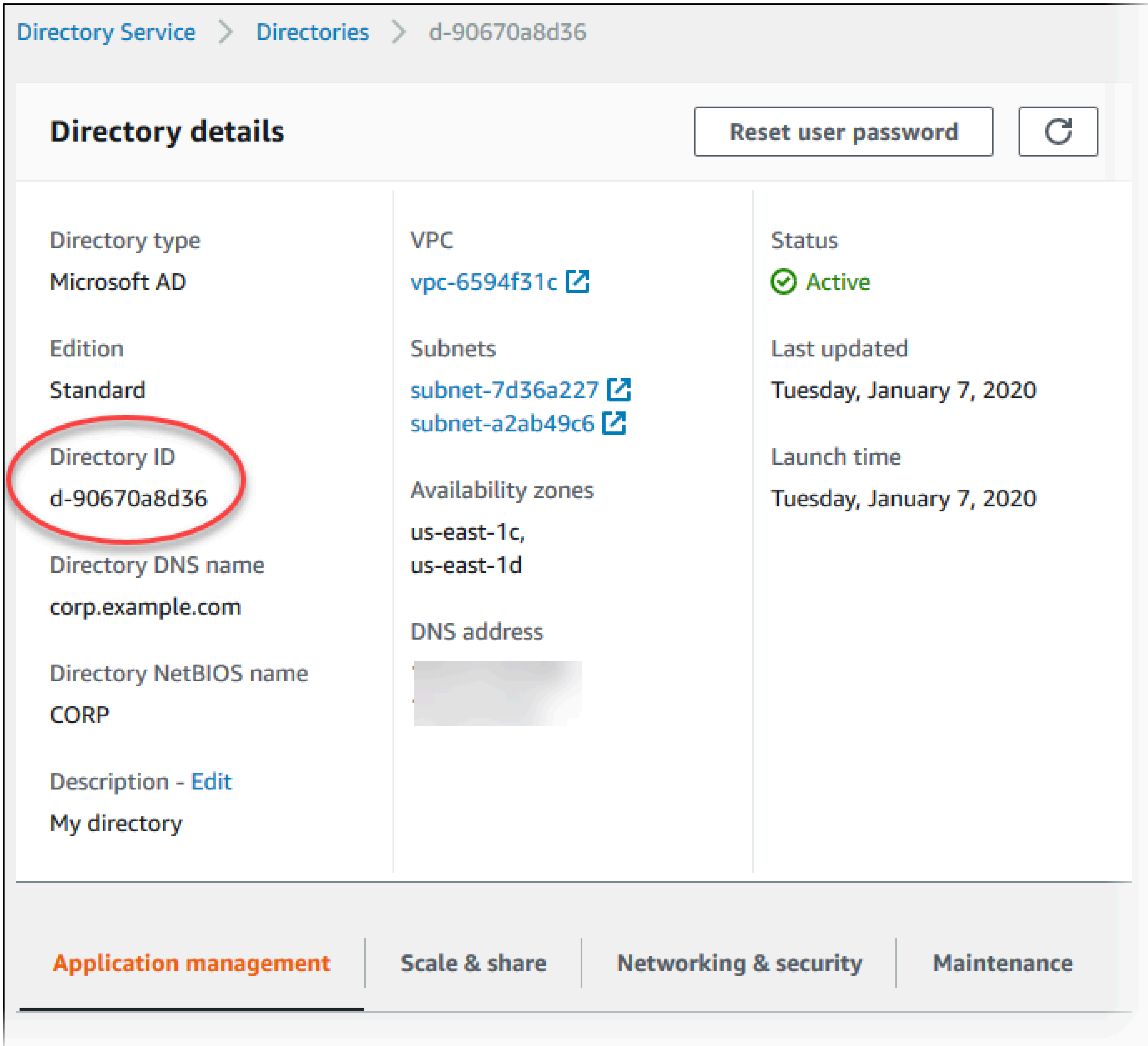
   子网

   　目录服务器的子网。两个子网必须位于不同的可用区。
7. 查看目录信息并进行必要的更改。如果信息正确，请选择 Create directory (创建目录)。



目录创建需要几分钟时间。创建成功后，Status (状态) 值将更改为 Active (活动)。

要查看有关您的目录的信息，请在目录列表中选择目录名称。请记下 Directory ID (目录 ID) 值，因为您在创建或修改 MySQL 数据库实例时需要此值。



步骤 2：创建 Amazon RDS 使用的 IAM 角色

要使 Amazon RDS 为您调用 Amazon Directory Service，需要使用托管 IAM 策略 `AmazonRDSDirectoryServiceAccess` 的 IAM 角色。该角色允许 Amazon RDS 调用 Amazon Directory Service。

使用 Amazon Web Services Management Console 创建数据库实例并且控制台用户具有 `iam:CreateRole` 权限时，控制台将自动创建此角色。在这种情况下，角色名为 `rds-directoryservice-kerberos-access-role`。否则，您必须手动创建 IAM 角

色。在创建该 IAM 角色时，请选择 `Directory Service`，然后将 Amazon 托管策略 `AmazonRDSDirectoryServiceAccess` 附加到该角色。

有关为服务创建 IAM 角色的更多信息，请参阅 IAM 用户指南中的创建向Amazon服务委托权限的角色。

> **Note**
>
> 用于 RDS for SQL Server 的 Windows 身份验证的 IAM 角色不能用于 RDS for MySQL。

或者，您可以创建具有所需权限的角色，而不是使用托管 IAM 策略 `AmazonRDSDirectoryServiceAccess`。在这种情况下，IAM 角色必须具有以下 IAM 信任策略。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "directoryservice.rds.amazonaws.com",
          "rds.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

角色还必须具有以下 IAM 角色策略。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "ds:DescribeDirectories",
        "ds:AuthorizeApplication",
        "ds:UnauthorizeApplication",
```

```
        "ds:GetAuthorizedApplicationDetails"
      ],
    "Effect": "Allow",
    "Resource": "*"
    }
  ]
}
```

步骤 3：创建和配置用户

您可以使用“Active Directory 用户和计算机”工具创建用户。该工具是 Active Directory Domain Services 和 Active Directory Lightweight Directory Services 工具的一部分。用户表示有权访问您的目录的独立个人或实体。

要在 Amazon Directory Service 目录中创建用户，您必须连接到基于 Microsoft Windows 的 Amazon EC2 实例。此实例必须是 Amazon Directory Service 目录的成员，并以具有创建用户权限的用户身份登录。有关更多信息，请参阅Amazon Managed Microsoft AD Directory Service 管理指南中的管理 Amazon 中的用户和组。

步骤 4：创建或修改 MySQL 数据库实例

创建或修改 MySQL 数据库实例，以便与目录一起使用。您可以使用控制台、CLI 或 RDS API 将数据库实例与目录关联。您可以通过下列方式之一来执行该操作：

- 使用控制台、create-db-instance CLI 命令或 CreateDBInstance RDS API 操作创建新的 MySQL 数据库实例。

  有关说明，请参阅创建 Amazon RDS 数据库实例。
- 使用控制台、modify-db-instance CLI 命令或 ModifyDBInstance RDS API 操作修改现有 MySQL 数据库实例。

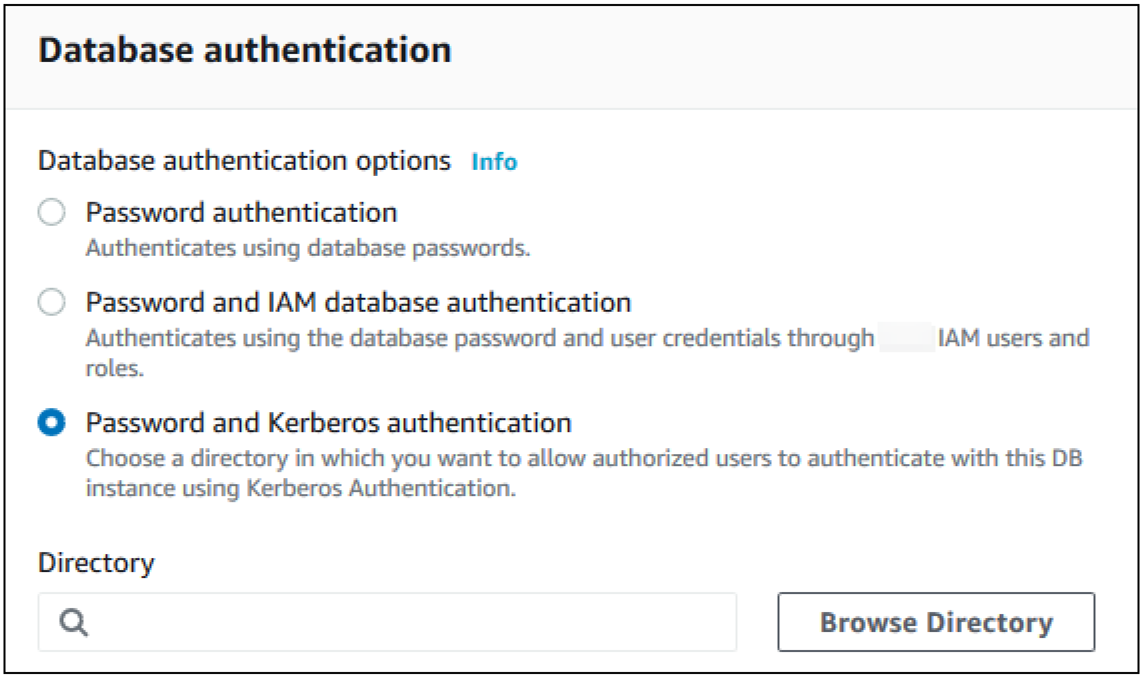
  有关说明，请参阅修改 Amazon RDS 数据库实例。
- 使用控制台、restore-db-instance-from-db-snapshot CLI 命令或 RestoreDBInstanceFromDBSnapshot RDS API 操作，从数据库快照还原 MySQL 数据库实例。

  有关说明，请参阅还原到数据库实例。
- 使用控制台、restore-db-instance-to-point-in-time CLI 命令或 RestoreDBInstanceToPointInTime RDS API 操作，将 MySQL 数据库实例还原到某个时间点。

  有关说明，请参阅将 Amazon RDS 的数据库实例还原到指定时间。

只有 VPC 中的 MySQL 数据库实例才支持 Kerberos 身份验证。数据库实例可以与目录在同一 VPC 中或在不同 VPC 中。数据库实例必须使用允许在目录的 VPC 中传出的安全组，以便数据库实例可与目录通信。

在使用控制台创建、修改或还原数据库实例时，请选择数据库身份验证部分中的密码和 Kerberos 身份验证。选择 Browse Directory (浏览目录)，然后选择目录或选择 Create a new directory (创建新目录)。



使用 Amazon CLI 或 RDS API 时，将数据库实例与目录关联。数据库实例需要以下参数才能使用您创建的域目录：

- 对于 `--domain` 参数，请使用创建目录时生成的域标识符（“d-*”标识符）。
- 对于 `--domain-iam-role-name` 参数，请使用您使用托管 IAM 策略 `AmazonRDSDirectoryServiceAccess` 创建的角色。

例如，以下 CLI 命令会修改数据库实例以使用目录。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier mydbinstance \
    --domain d-ID \
    --domain-iam-role-name role-name
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier mydbinstance ^
    --domain d-ID ^
    --domain-iam-role-name role-name
```

> **Important**
>
> 如果您修改数据库实例以启用 Kerberos 身份验证，请在进行更改之后重启数据库实例。

步骤 5：创建 Kerberos 身份验证 MySQL 登录名

使用 Amazon RDS 主用户凭证连接到 MySQL 数据库实例，就像连接到任何其他数据库实例一样。数据库实例加入到 Amazon Managed Microsoft AD 域。这样，您可以从您域中的 Active Directory 用户预配置 MySQL 登录名和用户。数据库权限是通过授予这些登录名并从这些登录名撤消的标准 MySQL 权限来管理的。

您可以允许 Active Directory 用户使用 MySQL 进行身份验证。为此，首先使用 Amazon RDS 主用户凭证连接到 MySQL 数据库实例，就像连接到任何其他数据库实例一样。登录之后，通过运行以下命令，在 MySQL 中使用 PAM（可插入身份验证模块），创建在外部进行身份验证的用户。将 *testuser* 替换为用户名。

```
CREATE USER 'testuser'@'%' IDENTIFIED WITH 'auth_pam';
```

域中用户（人和应用程序）现在均可从联接域的客户端计算机使用 Kerberos 身份验证连接到数据库实例。

> **Important**
>
> 我们强烈建议客户端在使用 PAM 身份验证时使用 SSL/TLS 连接。如果不使用 SSL/TLS 连接，在某些情况下密码可能会以明文形式发送。要为您的 AD 用户要求 SSL/TLS 加密的连接，请运行以下命令，并将 *testuser* 替换为用户名：
>
> ```
> ALTER USER 'testuser'@'%' REQUIRE SSL;
> ```
>
> 有关更多信息，请参阅 [Amazon RDS 上 MySQL 数据库实例的 SSL/TLS 支持]。

在域中管理数据库实例

您可以使用 CLI 或 RDS API 来管理您的数据库实例及其与托管的 Active Directory 的关系。例如，您可以关联一个 Active Directory 以进行 Kerberos 身份验证，并取消关联一个 Active Directory 以禁用 Kerberos 身份验证。您也可以将由一个 Active Directory 在外部进行身份验证的数据库实例移到另一个 Active Directory。

例如，使用 Amazon RDS API，您可以执行下列操作：

- 要重新尝试为失败的成员启用 Kerberos 身份验证，请使用 `ModifyDBInstance` API 操作并指定当前成员的目录 ID。
- 要为成员资格更新 IAM 角色名称，请使用 `ModifyDBInstance` API 操作并指定当前成员资格的目录 ID 和新的 IAM 角色。
- 要在数据库实例上禁用 Kerberos 身份验证，请使用 `ModifyDBInstance` API 操作并指定 none 作为域参数。
- 要将数据库实例从一个域移至另一个域，请使用 `ModifyDBInstance` API 操作并指定新域的域标识符作为域参数。
- 要列出每个数据库实例的成员资格，请使用 `DescribeDBInstances` API 操作。

了解域成员资格

在创建或修改数据库实例后，它将成为域的成员。您可以通过运行 [describe-db-instances](https://docs.aws.amazon.com/cli/latest/reference/rds/describe-db-instances.html) CLI 命令来查看数据库实例的域成员身份状态。数据库实例的状态可以是以下状态之一：

- `kerberos-enabled` – 数据库实例已启用 Kerberos 身份验证。
- `enabling-kerberos` – Amazon 正在此数据库实例上启用 Kerberos 身份验证。
- `pending-enable-kerberos` – 启用 Kerberos 身份验证正在此数据库实例上等待处理。
- `pending-maintenance-enable-kerberos` – Amazon将尝试在下一个计划的维护时段在数据库实例上启用 Kerberos 身份验证。
- `pending-disable-kerberos` – 禁用 Kerberos 身份验证正在此数据库实例上等待处理。
- `pending-maintenance-disable-kerberos` – Amazon将尝试在下一个计划的维护时段在数据库实例上禁用 Kerberos 身份验证。
- `enable-kerberos-failed` – 出现一个配置问题，导致Amazon无法在数据库实例上启用 Kerberos 身份验证。在重新发出数据库实例修改命令之前检查并修复配置。
- `disabling-kerberos` – Amazon 正在此数据库实例上禁用 Kerberos 身份验证。

启用 Kerberos 身份验证的请求可能因网络连接问题或不正确的 IAM 角色而失败。例如，假设您创建数据库实例或修改现有数据库实例，但启用 Kerberos 身份验证的尝试失败。如果发生这种情况，请重新发出修改命令或修改新创建的数据库实例以加入域。

使用 Kerberos 身份验证连接到 MySQL

要使用 Kerberos 身份验证连接到 MySQL，您必须使用 Kerberos 身份验证类型进行登录。

要创建可以使用 Kerberos 身份验证连接到的数据库用户，请在 `IDENTIFIED WITH` 语句上使用 `CREATE USER` 子句。有关说明，请参阅步骤 5：创建 Kerberos 身份验证 MySQL 登录名。

为避免发生错误，请使用 MariaDB `mysql` 客户端。您可以在 https://downloads.mariadb.org/ 下载 MariaDB 软件。

在命令提示符下，连接到其中一个与 MySQL 数据库实例关联的端点。按照 连接到 MySQL 数据库实例中的常规过程进行操作。当系统提示您输入密码时，请输入与该用户名关联的 Kerberos 密码。

还原 MySQL 数据库实例并将其添加到域中

您可还原数据库快照或为 MySQL 数据库实例完成时间点还原，然后将其添加到域中。数据库实例还原后，使用 步骤 4：创建或修改 MySQL 数据库实例 中介绍的过程修改此实例以将数据库实例添加到域中。

Kerberos 身份验证 MySQL 限制

以下限制适用于 MySQL 的 Kerberos 身份验证：

- 仅支持 Amazon Managed Microsoft AD。但是，您可以将 RDS for MySQL 数据库实例加入到同一 Amazon Web Services 区域中不同账户拥有的共享托管式 Microsoft AD 域。
- 启用该功能后，您必须重启数据库实例。
- 域名长度不能超过 61 个字符。
- 您无法同时启用 Kerberos 身份验证和 IAM 身份验证。为 MySQL 数据库实例选择一种或另一种身份验证方法。
- 启用该功能后，请勿修改数据库实例端口。
- 请勿将 Kerberos 身份验证与只读副本一起使用。
- 如果您为使用 Kerberos 身份验证的 MySQL 数据库实例启用了自动次要版本升级，则必须关闭 Kerberos 身份验证，然后在自动升级结束后将该功能重新打开。有关自动升级次要版本的更多信息，请参阅 RDS for MySQL 的自动次要版本升级。

- 要删除启用此功能的数据库实例，请先禁用该功能。为此，请对数据库实例运行 `modify-db-instance` CLI 命令并为 `--domain` 参数指定 `none`。

  如果您使用 CLI 或 RDS API 删除启用此功能的数据库实例，则预期会发生延迟。
- RDS for MySQL 不支持跨本地或自托管 AD 与 Amazon Managed Microsoft AD 之间的林信任进行 Kerberos 身份验证。

使用 Amazon RDS 优化读取提高 RDS for MySQL 的查询性能

使用 Amazon RDS 优化读取，您可以更快地处理 RDS for MySQL 查询。与不使用 RDS 优化型读取功能的数据库实例或集群相比，使用此功能的 RDS for MySQL 数据库实例或多可用区数据库集群可以实现最多快 2 倍的查询处理速度。

主题

RDS 优化读取概述

当您使用开启了 RDS 优化型读取功能的 RDS for MySQL 数据库实例或多可用区数据库集群时，该实例或集群会通过使用实例存储实现更快的查询性能。实例存储为数据库实例或多可用区数据库集群提供临时性块级存储。存储位于物理上附加到主机服务器的非易失性存储规范（NVMe）固态硬盘（SSD）上。此存储针对低延迟、高随机输入/输出性能和高速连续读取吞吐量进行了优化。

默认情况下，当数据库实例或多可用区数据库集群使用带有实例存储的数据库实例类（例如 db.m5d 或 db.m6gd）时，RDS 优化型读取功能处于开启状态。使用 RDS 优化读取，一些临时对象存储在实例存储中。这些临时对象包括内部临时文件、内部磁盘上的临时表、内存映射文件和二进制日志（binlog）缓存文件。有关实例存储的更多信息，请参阅《适用于 Linux 实例的 Amazon Elastic Compute Cloud 用户指南》中的 Amazon EC2 实例存储。

在 MySQL 中生成用于查询处理的临时对象的工作负载可以利用实例存储来加快查询处理速度。此类工作负载包括涉及排序、哈希聚合、高负载联接、公用表表达式（CTE）的查询以及对未索引列的查询。无论用于持久 Amazon EBS 存储的存储配置如何，这些实例存储卷都能提供更高的 IOPS 和性能。由于 RDS 优化读取将对临时对象的操作分载到实例存储，因此永久存储（Amazon EBS）的每秒进行读写操作的次数（IOPS）或吞吐量现在可用于对永久对象进行的操作。这些操作包括常规数据文件读取和写入以及后台引擎操作，例如刷新和插入缓冲区合并。

> **Note**
>
> 手动和自动 RDS 快照仅包含用于永久对象的引擎文件。在实例存储中创建的临时对象不包含在 RDS 快照中。

RDS 优化读取的使用案例

如果您的工作负载严重依赖于临时对象（例如内部表或文件）来执行查询，那么启用 RDS 优化读取会给您带来好处。以下使用案例是 RDS 优化读取的候选使用案例：

- 使用复杂的公用表表达式（CTE）、派生表和分组操作运行分析查询的应用程序
- 通过未优化的查询提供大量读取流量的只读副本
- 运行涉及复杂操作（例如，带 `GROUP BY` 和 `ORDER BY` 子句的查询）的按需或动态报告查询的应用程序
- 使用内部临时表进行查询处理的工作负载

  您可以监控引擎状态变量 `created_tmp_disk_tables`，以确定在数据库实例上创建的基于磁盘的临时表的数量。
- 直接或在过程中创建大型临时表以存储中间结果的应用程序
- 对非索引列执行分组或排序的数据库查询

RDS 优化读取的最佳实践

对于 RDS 优化读取使用以下最佳实践：

- 为只读查询添加重试逻辑，以防它们在执行过程中由于实例存储已满而失败。
- 使用 CloudWatch 指标 `FreeLocalStorage` 监控实例存储上的可用存储空间。如果由于数据库实例上的工作负载导致实例存储达到其极限，请修改数据库实例以使用更大的数据库实例类。
- 当您的数据库实例或多可用区数据库集群有足够的内存，但仍达到实例存储上的存储限制时，请增加 `binlog_cache_size` 值以将特定于会话的二进制日志条目保留在内存中。此配置可防止将二进制日志条目写入磁盘上的临时二进制日志缓存文件。

  `binlog_cache_size` 参数是特定于会话的。您可以更改每个新会话的值。此参数的设置可以提高高峰工作负载期间数据库实例的内存利用率。因此，请考虑根据应用程序的工作负载模式和数据库实例上的可用内存增加参数值。

- 对于 MySQL 8.0 及更低版本，binlog_format 参数的默认值为 MIXED。根据事务的大小，将 binlog_format 设置为 ROW 可能会导致实例存储上的二进制日志缓存文件很大。对于 MySQL 8.4 及更高版本，binlog_format 参数的默认值为 ROW。
- 将 internal_tmp_mem_storage_engine 参数设置为 TempTable，然后将 temptable_max_mmap 参数设置为与实例存储上的可用存储大小相匹配。
- 避免在单个事务中进行批量更改。这些类型的事务会在实例存储上生成大型二进制日志缓存文件，并在实例存储已满时导致问题。考虑将写入分成多个小事务，以显著减少二进制日志缓存文件的存储使用量。
- 对 binlog_error_action 参数使用 ABORT_SERVER 的默认值。这样做可以避免在启用备份的数据库实例上进行二进制日志记录时出现问题。

使用 RDS 优化读取

当您在单可用区数据库实例部署、多可用区数据库实例部署或多可用区数据库集群部署中预调配具有以下数据库实例类之一的 RDS for MySQL 数据库实例时，该数据库实例会自动使用 RDS 优化型读取功能。

要启用 RDS 优化读取，请执行以下操作之一：

- 使用其中一个数据库实例类创建 RDS for MySQL 数据库实例或多可用区数据库集群。有关更多信息，请参阅 创建 Amazon RDS 数据库实例。
- 修改现有 RDS for MySQL 数据库实例或多可用区数据库集群以使用其中一个数据库实例类。有关更多信息，请参阅 修改 Amazon RDS 数据库实例。

在支持其中一个或多个数据库实例类（具有本地 NVMe SSD 存储）的所有 Amazon Web Services 区域 RDS 中，均可使用 RDS 优化型读取功能。有关数据库实例类的信息，请参阅 the section called “数据库实例类”。

数据库实例类可用性因 Amazon Web Services 区域而异。要确定特定 Amazon Web Services 区域中是否支持数据库实例类，请参阅the section called “确定 Amazon Web Services 区域 中的数据库实例类支持”。

如果您不想使用 RDS 优化型读取特征，请修改您的数据库实例或多可用区数据库集群，使其不使用支持该特征的数据库实例类。

监控使用 RDS 优化读取的数据库实例

您可以使用以下 CloudWatch 指标监控使用 RDS 优化读取的数据库实例：

- `FreeLocalStorage`
- `ReadIOPSLocalStorage`
- `ReadLatencyLocalStorage`
- `ReadThroughputLocalStorage`
- `WriteIOPSLocalStorage`
- `WriteLatencyLocalStorage`
- `WriteThroughputLocalStorage`

这些指标提供有关可用实例存储的存储空间、IOPS 和吞吐量的数据。有关这些指标的更多信息，请参阅 Amazon RDS 的 Amazon CloudWatch 实例级指标。

RDS 优化读取的限制

以下限制适用于 RDS 优化读取：

- 以下版本支持 RDS 优化读取：
  - RDS for MySQL 8.0.28 版本以及更高的主要和次要版本

  有关 RDS for MySQL 版本的信息，请参阅 Amazon RDS 上的 MySQL 版本。
- 您无法在支持 RDS 优化读取的数据库实例类上将临时对象的位置更改为永久存储（Amazon EBS）。
- 在数据库实例上启用二进制日志记录时，最大事务大小受实例存储大小的限制。在 MySQL 中，任何需要的存储空间比 `binlog_cache_size` 值更多的会话都会将事务更改写入临时二进制日志缓存文件，这些文件是在实例存储上创建的。
- 实例存储已满时，事务可能会失败。

使用适用于 MySQL 的 RDS 优化型写入功能提高写入性能

您可以使用适用于 MySQL 的 RDS 优化写入功能来提高写入事务的性能。当您的 RDS for MySQL 数据库使用 RDS 优化写入时，它最多可以实现高两倍的写入事务吞吐量。

主题

RDS 优化写入概述

当您启用 RDS 优化写入时，您的 RDS for MySQL 数据库在将数据刷新到持久存储时仅写入一次，无需双写缓冲区。数据库继续为可靠的数据库事务提供 ACID 属性保护，同时提高性能。

关系数据库（如 MySQL）为可靠的数据库事务提供原子性、一致性、隔离性和持久性的 ACID 属性。为了帮助提供这些属性，MySQL 使用一个名为双写缓冲区的数据存储区域，以防止出现不完整页面写入错误。这些错误发生在数据库更新页面时出现硬件故障的情况下，例如停电时。MySQL 数据库可以检测不完整页面写入，并使用双写缓冲区中的页面副本进行恢复。虽然这种技术提供了保护，但也会导致额外的写入操作。有关 MySQL 双写缓冲区的更多信息，请参阅 MySQL 文档中的双写缓冲区。

开启 RDS 优化写入后，RDS for MySQL 数据库将在数据刷新到持久存储时仅写入一次，而不使用双写缓冲区。如果您在 RDS for MySQL 数据库上运行写入密集型工作负载，RDS 优化写入非常有用。具有写入密集型工作负载的数据库示例包括支持数字支付、金融交易和游戏应用程序的数据库。

这些数据库在使用 Amazon Nitro 系统的数据库实例类上运行。由于这些系统中的硬件配置，数据库可以在一个步骤中可靠而持久地将 16KiB 页直接写入数据文件。Amazon Nitro 系统使 RDS 优化写入成为可能。

您可以设置新的数据库参数 `rds.optimized_writes` 来控制适用于 RDS for MySQL 数据库的 RDS 优化写入功能。可在 RDS for MySQL 版本 8.0 和 RDS for MySQL 版本 8.4 的数据库参数组中访问此参数。使用以下值设置参数：

- `AUTO` – 如果数据库支持 RDS 优化写入，则开启此功能。如果数据库不支持 RDS 优化写入，请将其关闭。此设置是默认设置。
- `OFF` – 关闭 RDS 优化写入，即使数据库支持此功能也不例外。

如果您的现有数据库的引擎版本、数据库实例类和/或文件系统格式不支持 RDS 优化写入，则可以通过创建蓝绿部署来启用该功能。有关更多信息，请参阅 the section called “在现有数据库上启用”。

如果您将配置为使用 RDS 优化写入的 RDS for MySQL 数据库迁移到不支持该特征的数据库实例类，RDS 会自动对该数据库关闭 RDS 优化写入。

关闭 RDS 优化写入时，数据库使用 MySQL 双写缓冲区。

要确定 RDS for MySQL 数据库是否使用 RDS 优化写入，请查看数据库的 `innodb_doublewrite` 参数的当前值。如果数据库使用 RDS 优化写入，则此参数设置为 `FALSE`（`0`）。

使用 RDS 优化写入

使用 RDS 控制台、Amazon CLI 或 RDS API 创建 RDS for MySQL 数据库时，可以开启 RDS 优化写入。在数据库创建过程中，当以下两个条件都适用时，RDS 优化写入功能会自动开启：

- 您可以指定支持 RDS 优化写入的数据库引擎版本和数据库实例类。
  - RDS for MySQL 版本 8.0.30 及更高版本支持 RDS 优化写入。有关 RDS for MySQL 版本的信息，请参阅 Amazon RDS 上的 MySQL 版本。
  - 使用以下数据库实例类的 RDS for MySQL 数据库支持 RDS 优化写入：
    - db.m7i
    - db.m7g
    - db.m6g
    - db.m6gd
    - db.m6i
    - db.m5
    - db.m5d
    - db.r7i
    - db.r7g
    - db.r6g
    - db.r6gd
    - db.r6i
    - db.r5
    - db.r5b
    - db.r5d

  - db.x2idn
  - db.x2iedn

  有关数据库实例类的信息，请参阅 the section called “数据库实例类”。

  数据库实例类可用性因 Amazon Web Services 区域而异。要确定特定 Amazon Web Services 区域中是否支持数据库实例类，请参阅the section called “确定 Amazon Web Services 区域 中的数据库实例类支持”。

  要将数据库升级为支持 RDS 优化写入的数据库实例类，您可以创建蓝绿部署。有关更多信息，请参阅 the section called “在现有数据库上启用”。

- 在与数据库关联的参数组中，`rds.optimized_writes` 参数设置为 `AUTO`。在默认参数组中，此参数始终设置为 `AUTO`。

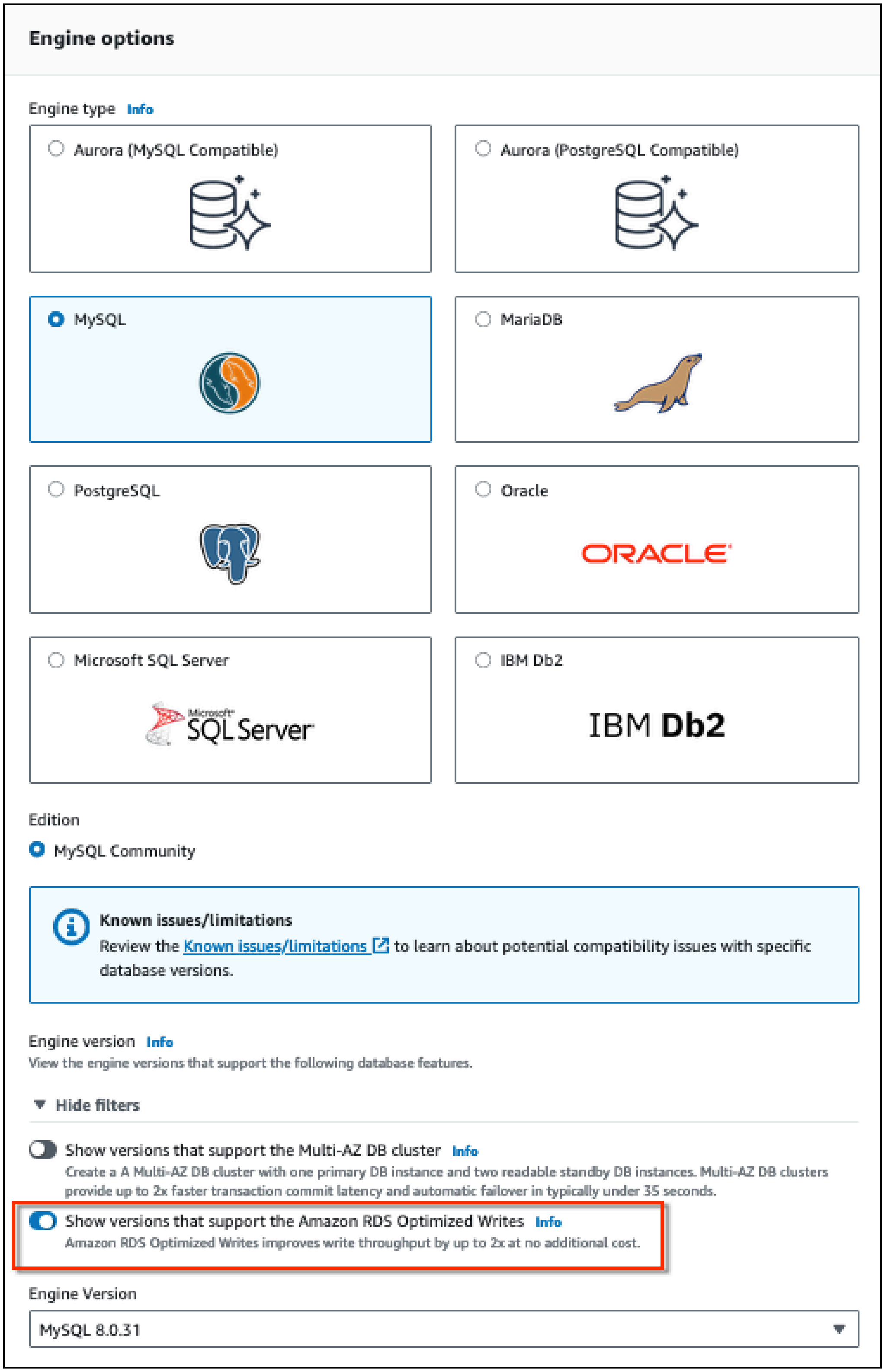
如果您需要使用支持 RDS 优化型写入特征的数据库引擎版本和数据库实例类，但您不想使用此特征，则在创建数据库时指定一个自定义参数组。在此参数组中，将 `rds.optimized_writes` 参数设置为 `OFF`。如果您希望数据库稍后使用 RDS 优化写入，可以将此参数设置为 `AUTO` 以将其开启。有关创建自定义参数组和设置参数的信息，请参阅 Amazon RDS 的参数组。

有关创建数据库实例的信息，请参阅创建 Amazon RDS 数据库实例。

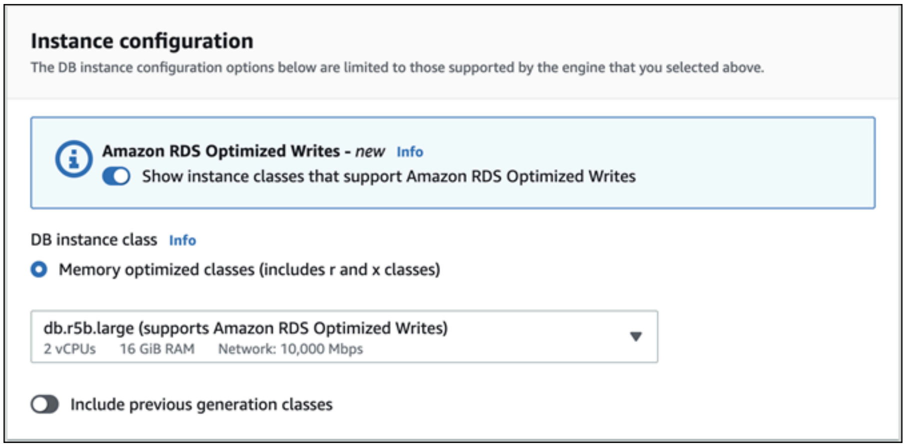
控制台

当您使用 RDS 控制台创建 RDS for MySQL 数据库时，您可以筛选支持 RDS 优化写入的数据库引擎版本和数据库实例类。开启筛选条件后，您可以从可用的数据库引擎版本和数据库实例类中进行选择。

要选择支持 RDS 优化写入的数据库引擎版本，请在 Engine version（引擎版本）中筛选支持该功能的 RDS for MySQL 数据库引擎版本，然后选择版本。



在 Instance configuration（实例配置）部分中，筛选支持 RDS 优化写入的数据库实例类，然后选择数据库实例类。



做出这些选择后，您可以选择符合要求的其他设置，并使用控制台完成创建 RDS for MySQL 数据库。

Amazon CLI

要使用 Amazon CLI 创建数据库实例，请运行 create-db-instance 命令。确保 `--engine-version` 和 `--db-instance-class` 值支持 RDS 优化写入。此外，确保与数据库实例关联的参数组将 `rds.optimized_writes` 参数设置为 `AUTO`。以下示例将默认参数组与数据库实例关联。

Example 创建使用 RDS 优化写入的数据库实例

对于 Linux、macOS 或 Unix：

```
aws rds create-db-instance \
    --db-instance-identifier mydbinstance \
    --engine mysql \
    --engine-version 8.0.30 \
    --db-instance-class db.r5b.large \
    --manage-master-user-password \
    --master-username admin \
    --allocated-storage 200
```

对于 Windows：

```
aws rds create-db-instance ^
    --db-instance-identifier mydbinstance ^
    --engine mysql ^
    --engine-version 8.0.30 ^
    --db-instance-class db.r5b.large ^
    --manage-master-user-password ^
    --master-username admin ^
```

```
    --allocated-storage 200
```

RDS API

您可以使用 CreateDBInstance 操作创建数据库实例。使用此操作时，请确保 EngineVersion 和 DBInstanceClass 值支持 RDS 优化写入。此外，确保与数据库实例关联的参数组将 rds.optimized_writes 参数设置为 AUTO。

在现有数据库上启用 RDS 优化写入

要修改现有 RDS for MySQL 数据库以开启 RDS 优化写入功能，数据库必须使用支持的数据库引擎版本和数据库实例类创建。此外，数据库必须是在 2022 年 11 月 27 日发布 RDS 优化写入之后创建的，因为所需的底层文件系统配置与此次发布前创建的数据库的配置不兼容。如果满足上述条件，则可以通过将 rds.optimized_writes 参数设置为 AUTO 来开启 RDS 优化写入。

如果您的数据库不是使用支持的引擎版本、实例类或文件系统配置创建的，则可以使用 RDS 蓝绿部署来迁移到支持的配置。创建蓝绿部署时，请执行以下操作：

- 选择在绿色数据库上启用优化写入，然后指定支持 RDS 优化写入的引擎版本和数据库实例类。有关支持的引擎版本和实例类的列表，请参阅使用 RDS 优化写入。
- 在存储下，选择升级存储文件系统配置。此选项将数据库升级到兼容的底层文件系统配置。

创建蓝绿部署时，如果将 rds.optimized_writes 参数设置为 AUTO，则在绿色环境中将自动启用 RDS 优化写入。然后，您可以切换蓝绿部署，以将绿色环境升级为新的生产环境。

有关更多信息，请参阅 the section called “创建蓝绿部署”。

RDS 优化写入的限制

当您从快照中还原 RDS for MySQL 数据库时，仅当满足以下所有条件时，才能为数据库开启 RDS 优化写入：

- 快照是从支持 RDS 优化写入的数据库创建的。
- 快照是从发布 RDS 优化型写入功能之后创建的数据库中创建的。
- 快照还原为支持 RDS 优化写入的数据库。
- 还原的数据库与一个参数组相关联，该参数组的 rds.optimized_writes 参数设置为 AUTO。

升级 RDS for MySQL 数据库引擎

如果 Amazon RDS 支持数据库引擎的新版本，您可以将您的数据库实例升级到新版本。MySQL 数据库有两种升级方式：主要版本升级和次要版本升级。

主要版本升级。

主要版本升级会包含不与现有应用程序向后兼容的数据库更改。因此，您必须手动为数据库实例执行主要版本升级。您可以通过修改数据库实例来启动主要版本升级。在执行主要版本升级之前，建议您按照RDS for MySQL 的主要版本升级中的说明操作。

对于多可用区数据库实例部署的主要版本升级，Amazon RDS 会同时升级主要副本和备用副本。在升级完成之前，数据库实例将不可用。对于多可用区数据库集群部署的主要版本升级，Amazon RDS 一次升级一个集群成员实例。

> **Tip**
>
> 您可以使用蓝绿部署，最大限度地减少主要版本升级所需的停机时间。有关更多信息，请参阅 使用 Amazon RDS 蓝绿部署进行数据库更新。

次要版本升级

次要版本升级 仅包含与现有应用程序向后兼容的更改。您可以通过修改数据库实例来手动启动次要版本升级。您也可以在创建或修改数据库实例时启用自动次要版本升级选项。这样做意味着 Amazon RDS 会在测试并批准新版本后自动升级您的数据库实例。有关执行升级的信息，请参阅升级数据库实例引擎版本。

当您对多可用区数据库集群执行次要版本升级时，Amazon RDS 将一次升级一个读取器数据库实例。然后，其中一个读取器数据库实例将切换为新的写入器数据库实例。然后，Amazon RDS 升级旧的写入器实例（现在是读取器实例）。

> **Note**
>
> 多可用区数据库实例部署的次要版本升级的停机时间可能会持续几分钟。多可用区数据库集群通常可将次要版本升级的停机时间缩短至大约 35 秒。与 RDS 代理一起使用时，您可以将停机时间进一步缩短到一秒或更短。有关更多信息，请参阅 Amazon RDS 代理。或者，您可以使用开源数据库代理，例如 ProxySQL、PgBouncer 或 Amazon Advanced JDBC Wrapper Driver。

如果 MySQL 数据库实例使用只读副本，则必须在升级源实例之前升级所有只读副本。

主题

MySQL 升级的注意事项

在升级流程期间，Amazon RDS 会创建两个或更多数据库快照。在进行任何升级更改之前，Amazon RDS 最多会拍摄两个数据库实例快照。如果无法完成数据库的升级，那么，就可以还原这些快照之一，来创建一个运行旧版本的数据库实例。升级完成后，Amazon RDS 会为数据库实例拍摄另一个快照。无论 Amazon Backup 是否管理数据库实例的备份，Amazon RDS 都会拍摄这些快照。

> **Note**
>
> 如果您将数据库实例的备份保留期设置为大于 0 的数字，则 Amazon RDS 只需创建数据库快照。要更改您的备份保留期，请参阅 修改 Amazon RDS 数据库实例。

升级完成后，您无法恢复为数据库引擎的以前版本。如果要返回以前版本，则还原创建的第一个数据库快照以新建数据库实例。

您可以控制何时将数据库实例升级到 Amazon RDS 支持的新版本。此级别的控制可帮助您保留与特定数据库版本的兼容性，并在生产中部署前使用应用程序测试新版本。在您准备好后，可在与时间表最搭配的时间执行版本升级。

如果数据库实例使用只读复制，则必须在升级源实例之前升级所有只读副本。

查找有效的升级目标

使用 Amazon Web Services Management Console 升级数据库实例时，它会显示数据库实例的有效升级目标。您还可以运行以下 Amazon CLI 命令来标识数据库实例的有效升级目标：

对于 Linux、macOS 或 Unix：

```
aws rds describe-db-engine-versions \
  --engine mysql \
  --engine-version version_number \
  --query "DBEngineVersions[*].ValidUpgradeTarget[*].{EngineVersion:EngineVersion}" --
output text
```

对于 Windows：

```
aws rds describe-db-engine-versions ^
  --engine mysql ^
  --engine-version version_number ^
  --query "DBEngineVersions[*].ValidUpgradeTarget[*].{EngineVersion:EngineVersion}" --
output text
```

例如，要标识 MySQL 版本 8.0.28 数据库实例的有效升级目标，请运行以下 Amazon CLI 命令：

对于 Linux、macOS 或 Unix：

```
aws rds describe-db-engine-versions \
  --engine mysql \
  --engine-version 8.0.28 \
  --query "DBEngineVersions[*].ValidUpgradeTarget[*].{EngineVersion:EngineVersion}" --
output text
```

对于 Windows：

```
aws rds describe-db-engine-versions ^
  --engine mysql ^
  --engine-version 8.0.28 ^
  --query "DBEngineVersions[*].ValidUpgradeTarget[*].{EngineVersion:EngineVersion}" --
output text
```

MySQL 版本号

RDS for MySQL 数据库引擎的版本编号序列采用 major.minor.patch.YYYYMMDD 或 major.minor.patch 形式，例如 8.0.33.R2.20231201 或 5.7.44。使用的格式取决于 MySQL 引擎版本。有关 RDS 扩展支持版本号的信息，请参阅Amazon RDS Extended Support 版本命名。

major

主要版本号是版本号的整数和第一个小数点后的部分，例如 8.0。主要版本升级会增加版本号的主要版本部分。例如，从 5.7.44 升级到 8.0.33 是主要版本升级，其中 5.7 和 8.0 为主要版本号。

minor

次要版本号是版本号的第三部分，例如 8.0.33 中的 33。

patch

补丁是版本号的第四部分，例如 8.0.33.R2 中的 R2。RDS 补丁版本包括在发布后添加到次要版本的重要错误修复。

YYYYMMDD

日期是版本号的第五部分，例如 8.0.33.R2.20231201 中的 20231201。RDS 日期版本是一个安全补丁，它包括在发布后添加到次要版本的重要安全修复程序。它不包括任何可能更改引擎行为的修复程序。

下表说明了 RDS for MySQL 版本 8.4 的命名方案。

8.4 次要版本	命名模式
≥ 3	新的数据库实例使用 major.minor.patch.YYMMDD，例如 8.4.3.R2.20241201。 在下一次主要版本或次要版本升级之前，现有数据库实例可能使用 major.minor.patch，例如 8.4.3.R2。

下表说明了 RDS for MySQL 版本 8.0 的命名方案。

8.0 次要版本	命名模式
≥ 33	新的数据库实例使用 major.minor.patch.YYMMDD，例如 8.0.33.R2.20231201。 在下一次主要版本或次要版本升级之前，现有数据库实例可能使用 major.minor.patch，例如 8.0.33.R2。
< 33	现有数据库实例使用 major.minor.patch，例如 8.0.32.R2。

下表说明了 RDS for MySQL 版本 5.7 的命名方案。

5.7 次要版本	命名模式
≥ 42	新的数据库实例使用 major.minor.patch.YYMMDD，例如 5.7.42.R2.20231201。 在下一次主要版本或次要版本升级之前，现有数据库实例可能使用 major.minor.patch，例如 5.7.42.R2。

RDS for MySQL 中的 RDS 版本号

RDS 版本号使用 *major*.*minor*.*patch* 或 *major*.*minor*.*patch*.*YYYYMMDD* 命名方案。Amazon RDS 扩展支持版本使用 *minor-RDS.YYYYMMDD* 次要版本命名方案。

RDS 补丁版本包括在发布后添加到次要版本的重要错误修复。RDS 日期版本（*YYYYMMDD*）是一个安全补丁。安全补丁不包括任何可能更改引擎行为的修复程序。有关 RDS 扩展支持版本号的信息，请参阅Amazon RDS Extended Support 版本命名。

您可以通过以下 SQL 查询找出 RDS for MySQL 数据库的 RDS 版本号：

```
mysql> select mysql.rds_version();
```

例如，查询 RDS for MySQL 8.0.34 数据库将返回以下输出：

```
+---------------------+
| mysql.rds_version() |
```

```
+----------------------+
| 8.0.34.R2.20231201   |
+----------------------+
1 row in set (0.01 sec)
```

RDS for MySQL 的主要版本升级

Amazon RDS 支持 MySQL 数据库引擎的以下主要版本就地升级：

- MySQL 5.7 到 MySQL 8.0
- MySQL 8.0 到 MySQL 8.4

> **Note**
>
> 您只能使用最新一代和当前一代数据库实例类创建 MySQL 版本 5.7、8.0 和 8.4 数据库实例。在某些情况下，您希望将在上一代数据库实例类上运行的数据库实例升级到具有更高 MySQL 引擎版本的数据库实例。在这些情况下，先修改数据库实例以使用最新一代或当前一代数据库实例类。执行此操作之后，您可以修改数据库实例以使用更高的 MySQL 数据库引擎版本。有关 Amazon RDS 数据库实例类的信息，请参阅数据库实例类。

主题

MySQL 主要版本升级概述

主要版本升级会包含不与现有应用程序向后兼容的数据库更改。因此，Amazon RDS 不会自动应用主要版本升级；您必须手动修改数据库实例。建议您在将任何升级应用于生产实例前全面测试这些升级。

要执行主要版本升级，请首先执行任何可用的操作系统更新。操作系统更新完成后，请升级到各个主要版本，例如从 5.7 升级到 8.0，然后再升级到 8.0 到 8.4。有关升级 RDS for MySQL 多可用区数据库集群的信息，请参阅升级 Amazon RDS 的多可用区数据库集群的引擎版本。在 2014 年 4 月 24 日之前创建的 MySQL 数据库实例会显示可用的操作系统更新，直到更新安装完成。有关操作系统升级的更多信息，请参阅 对数据库实例应用更新。

在 MySQL 的主要版本升级期间，如有必要，Amazon RDS 会运行 MySQL 二进制文件 `mysql_upgrade` 以便升级表。此外，Amazon RDS 会在主要版本升级期间清空 `slow_log` 和 `general_log` 表。要保留日志信息，请在升级主要版本之前保存日志内容。

MySQL 主要版本升级通常在大约 10 分钟内完成。由于数据库实例类大小不同或由于实例未遵循Amazon RDS 的最佳实践中的特定操作指南，某些升级过程可能需要更长时间。如果您从 Amazon RDS 控制台升级数据库实例，则数据库实例的状态会标明升级完成的时间。如果您使用 Amazon Command Line Interface (Amazon CLI) 进行升级，请使用 describe-db-instances 命令并选中 `Status` 值。

升级预检查

Amazon RDS 在升级之前会运行预检查以检查是否有不兼容问题。这些不兼容问题因要升级到的 MySQL 版本不同而异。

预检查包括 MySQL 内的一些预检查和 Amazon RDS 团队专门创建的一些预检查。有关 MySQL 提供的预检查的信息，请参阅升级检查程序实用工具。

在为了升级而停止数据库实例之前先运行预检查，这意味着它们在运行时不会造成任何停机。如果预检查发现不兼容问题，Amazon RDS 会在停止数据库实例之前自动取消升级。Amazon RDS 还会针对不兼容问题生成事件。有关 Amazon RDS 事件的更多信息，请参阅 使用 Amazon RDS 事件通知。

Amazon RDS 在日志文件 `PrePatchCompatibility.log` 中记录有关每项不兼容性的详细信息。在大部分情况下，日志条目包括用于纠正不兼容性的 MySQL 文档的链接。有关查看日志文件的更多信息，请参阅 查看和列出数据库日志文件。

由于预检查的性质，它们会分析数据库中的对象。此分析会导致资源消耗并增加完成升级的时间。

主题

- 从 MySQL 8.0 升级到 8.4 的预检查
- 从 MySQL 5.7 升级到 8.0 的预检查

从 MySQL 8.0 升级到 8.4 的预检查

MySQL 8.4 与 MySQL 8.0 存在一定的不兼容性。在从 MySQL 8.0 升级到 MySQL 8.4 时，这些不兼容性会引起问题。因此，为了让升级成功，可能需要对数据库做一些准备。以下是这些不一致项的一般列表：

- 不得有使用过时的数据类型或函数的表。

- 触发器不得具有缺失的或空的定义程序或无效的创建上下文。
- 不得出现关键字或保留关键字违规情况。MySQL 8.4 中可能会保留一些以前未保留的关键字。

  有关更多信息，请参阅 MySQL 文档中的关键字和保留关键字。
- MySQL 8.0 `mysql` 系统数据库中不得有与 MySQL 8.4 数据字典使用的表同名的表。
- `sql_mode` 系统变量设置中不得定义过时的 SQL 模式。
- 不得有包含超过 255 个字符或 1020 个字节的单个 `ENUM` 或 `SET` 列元素的表或存储过程。
- 您的 MySQL 8.0 安装不得使用 MySQL 8.4 不支持的功能。

  有关更多信息，请参阅 MySQL 文档中的 Features removed in MySQL 8.4。
- 不得有超过 64 个字符的外键约束名称。
- 对于改进的 Unicode 支持，请查看以下信息：
  - 请考虑将使用 `utf8mb3` 字符集的对象转换为使用 `utf8mb4` 字符集。`utf8mb3` 字符集已弃用。
  - 请考虑对字符集引用使用 `utf8mb4` 而不是 `utf8`，因为 `utf8` 当前是 `utf8mb3` 字符集的别名。如果可能，请先将 `utf8` 更改为 `utf8mb4`，然后再升级数据库。
  - 对于 `utf8mb3`，由于较旧的客户端可能会收到未知字符集错误，因此请在升级数据库之前升级数据库客户端。

  有关更多信息，请参阅 MySQL 文档中的 utf8mb3 字符集（3 字节 UTF-8 Unicode 编码）。

  要更改字符集，您可以手动对数据库执行备份、还原和复制。您也可以使用 Amazon RDS 蓝绿部署。有关更多信息，请参阅 使用 Amazon RDS 蓝绿部署进行数据库更新。

当您开始从 MySQL 8.0 更新到 8.4 时，Amazon RDS 会自动运行预检查，以便检测这些不兼容性。有关升级到 MySQL 8.4 的信息，请参阅 MySQL 文档中的 Upgrading MySQL。

这些预检查是必需的。您不能选择跳过它们。预检查提供以下好处：

- 它们让您可以在升级期间避免出现计划外停机。
- 如果存在不一致项，Amazon RDS 将阻止升级并提供日志以供您参阅。然后，您可以使用日志，通过减少不一致性来准备数据库以升级到 MySQL 8.4。有关消除不兼容情况的详细信息，请参阅 MySQL 文档中的 Preparing your installation for upgrade。

从 MySQL 5.7 升级到 8.0 的预检查

MySQL 8.0 与 MySQL 5.7 存在一定的不兼容性。在从 MySQL 5.7 升级到 MySQL 8.0 时，这些不兼容性会引起问题。因此，为了让升级成功，可能需要对数据库做一些准备。以下是这些不一致项的一般列表：

- 不得有使用过时的数据类型或函数的表。
- 不得有孤立的 *.frm 文件。
- 触发器不得具有缺失的或空的定义程序或无效的创建上下文。
- 不得有使用不支持本机分区的存储引擎的分区表。
- 不得出现关键字或保留关键字违规情况。MySQL 8.0 中可能会保留一些以前未保留的关键字。

  有关更多信息，请参阅 MySQL 文档中的关键字和保留关键字。
- MySQL 5.7 `mysql` 系统数据库中不得有与 MySQL 8.0 数据字典使用的表同名的表。
- `sql_mode` 系统变量设置中不得定义过时的 SQL 模式。
- 不得有包含超过 255 个字符或 1020 个字节的单个 `ENUM` 或 `SET` 列元素的表或存储过程。
- 在升级到 MySQL 8.0.13 或更高版本之前，不得有驻留在共享 InnoDB 表空间中的表分区。
- MySQL 8.0.12 或更低版本中不得有对 `ASC` 子句使用 `DESC` 或 `GROUP BY` 限定符的查询和存储程序定义。
- 您的 MySQL 5.7 安装不得使用 MySQL 8.0 不支持的功能。

  有关更多信息，请参阅 MySQL 文档中的 MySQL 8.0 中删除的功能。
- 不得有超过 64 个字符的外键约束名称。
- 对于改进的 Unicode 支持，请查看以下信息：
  - 请考虑将使用 `utf8mb3` 字符集的对象转换为使用 `utf8mb4` 字符集。`utf8mb3` 字符集已弃用。
  - 请考虑对字符集引用使用 `utf8mb4` 而不是 `utf8`，因为 `utf8` 当前是 `utf8mb3` 字符集的别名。如果可能，请先将 `utf8` 更改为 `utf8mb4`，然后再升级数据库。
  - 对于 `utf8mb3`，由于较旧的客户端可能会收到未知字符集错误，因此请在升级数据库之前升级数据库客户端。

  有关更多信息，请参阅 MySQL 文档中的 utf8mb3 字符集（3 字节 UTF-8 Unicode 编码）。

  要更改字符集，您可以手动对数据库执行备份、还原和复制。您也可以使用 Amazon RDS 蓝绿部署。有关更多信息，请参阅 使用 Amazon RDS 蓝绿部署进行数据库更新。

当您开始从 MySQL 5.7 更新到 8.0 时，Amazon RDS 会自动运行预检查，以便检测这些不兼容性。有关升级到 MySQL 8.0 的信息，请参阅 MySQL 文档中的升级 MySQL。

这些预检查是必需的。您不能选择跳过它们。预检查提供以下好处：

- 它们让您可以在升级期间避免出现计划外停机。
- 如果存在不一致项，Amazon RDS 将阻止升级并提供日志以供您参阅。然后，您可以使用日志，通过减少不一致性来准备数据库以升级到 MySQL 8.0。有关消除不兼容性的详细信息，请参阅 MySQL 文档中的准备安装以进行升级和 MySQL Server 博客上的升级到 MySQL 8.0？ 以下是您需要了解的内容…。

升级失败后回滚

当将数据库实例从 MySQL 版本 5.7 升级到 MySQL 版本 8.0 或从 MySQL 版本 8.0 升级到 8.4 时，升级可能会失败。尤其是，如果数据字典中包含预检查未发现的不兼容性，则可能会失败。在这种情况下，数据库无法在新的 MySQL 8.0 或 8.4 版本中成功启动。此时，升级所进行的更改就会 Amazon RDS 回滚。回滚后，MySQL 数据库实例将运行原始版本：

- MySQL 版本 8.0（用于从 MySQL 8.4 回滚）
- MySQL 版本 5.7（用于从 MySQL 8.0 回滚）

当升级失败并回滚时，Amazon RDS 生成一个 ID 为 RDS-EVENT-0188 的事件。

通常，升级失败是因为数据库实例中的数据库与目标 MySQL 版本之间的元数据存在不兼容性。升级失败时，您可以在 `upgradeFailure.log` 文件中查看有关这些不兼容性的详细信息。在尝试再次升级之前，请先解决不兼容问题。

尝试升级和回滚失败时，您的数据库实例会重新启动。任何待处理的参数更改都将在重新启动期间应用，并在回滚后保留。

有关升级到 MySQL 8.0 的更多信息，请参阅 MySQL 文档中的以下主题：

- 准备安装以进行升级
- 升级到 MySQL 8.0？ 您需要知道这些...

有关升级到 MySQL 8.4 的更多信息，请参阅 MySQL 文档中的 Preparing Your Installation for Upgrade。

测试 RDS for MySQL 升级

在数据库实例上执行主要版本升级前，彻底地测试数据库，了解是否与新版本兼容。此外，彻底地测试访问数据库的所有应用程序，了解是否与新版本兼容。建议您使用以下过程。

测试主要版本升级

1. 检查新版数据库引擎的升级文档以了解是否存在可能会影响数据库或应用程序的兼容性问题：
   - MySQL 5.7 中的变化
   - MySQL 8.0 中的变化
   - MySQL 8.4 中的变化
2. 如果数据库实例是自定义数据库参数组的成员，就必须使用兼容新主要版本的现有设置创建一个新的数据库参数组。在升级测试实例时，应指定新的数据库参数组，以便升级测试可确保其运行正常。有关创建数据库参数组的详细信息，请参阅 Amazon RDS 的参数组。
3. 为要升级的数据库实例创建数据库快照。有关更多信息，请参阅“为 Amazon RDS 的单可用区数据库实例创建数据库快照”。
4. 恢复数据库快照，以便创建一个新的测试数据库实例。有关更多信息，请参阅“还原到数据库实例”。
5. 使用下方详细介绍的某种方法，修改此新测试数据库实例以将其升级到新版本。如果您在步骤 2 中创建了一个新的参数组，则指定该参数组。
6. 计算升级的实例使用的存储容量以决定升级是否需要额外的存储容量。
7. 对升级的数据库实例运行所需次数的质量保证测试以确保数据库和应用程序可正常使用新版本。实施任何必需的新测试，评估您在步骤 1 中发现的任何兼容性问题的影响。测试所有的存储过程和功能。将应用程序的测试版本指向所升级的数据库实例。
8. 如果所有测试通过，那么请在生产数据库实例上执行升级。建议您直到确认一切正常后，再允许对数据库实例执行写入操作。

升级 MySQL 数据库实例

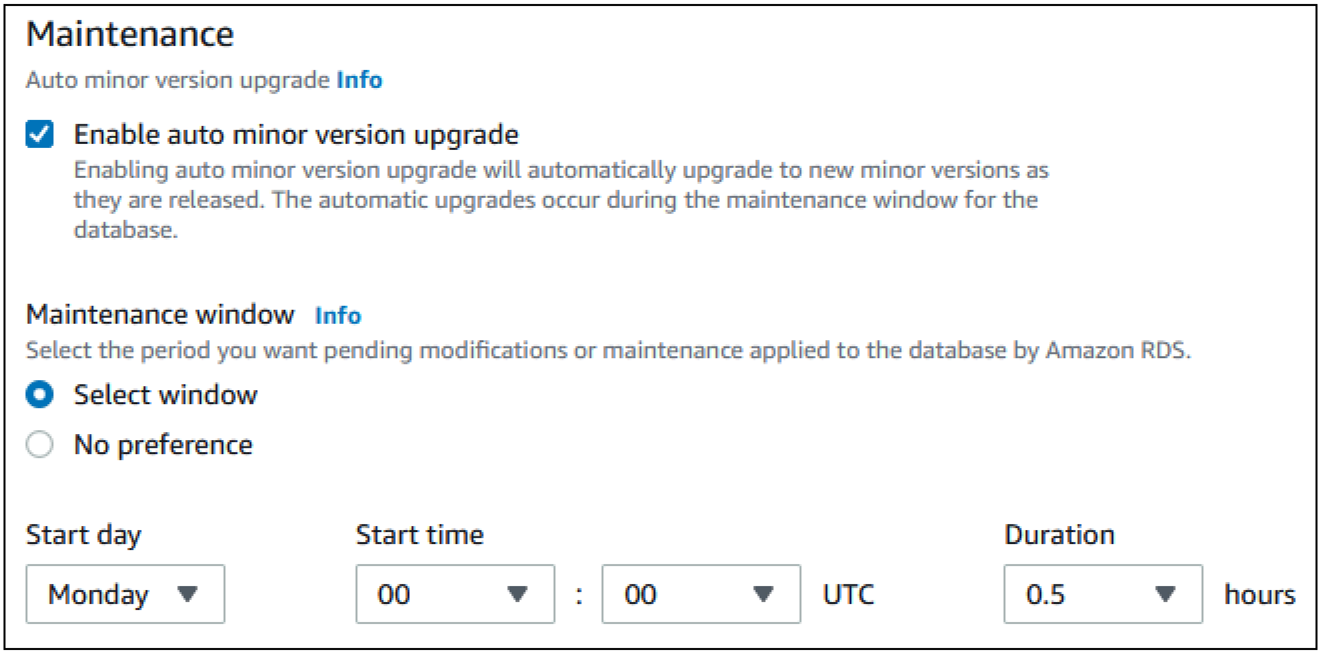
有关手动或自动升级 MySQL 数据库实例的信息，请参阅升级数据库实例引擎版本。

RDS for MySQL 的自动次要版本升级

如果在创建或修改数据库实例时指定以下设置，则可以让数据库实例自动升级。

- 自动次要版本升级设置已启用。
- 备份保留期设置需大于 0。

在 Amazon Web Services Management Console 中，这些设置位于 Additional configuration（其他配置）下。下图显示了 Auto minor version upgrade（自动次要版本升级）设置。



有关这些设置的更多信息，请参阅 数据库实例的设置。

对于某些 Amazon Web Services 区域中的某些 RDS for MySQL 主要版本，RDS 将一个次要版本指定为自动升级版本。在 Amazon RDS 测试并批准次要版本后，将在维护时段内自动进行次要版本升级。RDS 不会自动将发布的较新次要版本设置为自动升级版本。在 RDS 指定较新的自动升级版本之前，将会考虑一些条件，例如：

- 已知的安全问题
- MySQL 社区版本中的错误
- 发布次要版本后的整体队列稳定性

您可以运行以下 Amazon CLI 命令，以确定特定 Amazon Web Services 区域中指定 MySQL 次要版本的当前自动次要升级目标版本。

对于 Linux、macOS 或 Unix：

```
aws rds describe-db-engine-versions \
```

```
--engine mysql \
--engine-version minor_version \
--region region \
--query "DBEngineVersions[*].ValidUpgradeTarget[*].
{AutoUpgrade:AutoUpgrade,EngineVersion:EngineVersion}" \
--output text
```

对于 Windows：

```
aws rds describe-db-engine-versions ^
--engine mysql ^
--engine-version minor_version ^
--region region ^
--query "DBEngineVersions[*].ValidUpgradeTarget[*].
{AutoUpgrade:AutoUpgrade,EngineVersion:EngineVersion}" ^
--output text
```

例如，以下 Amazon CLI 命令可确定美国东部（俄亥俄州）Amazon Web Services 区域 (us-east-2) 中 MySQL 次要版本 8.0.11 的自动次要升级目标。

对于 Linux、macOS 或 Unix：

```
aws rds describe-db-engine-versions \
--engine mysql \
--engine-version 8.0.11 \
--region us-east-2 \
--query "DBEngineVersions[*].ValidUpgradeTarget[*].
{AutoUpgrade:AutoUpgrade,EngineVersion:EngineVersion}" \
--output table
```

对于 Windows：

```
aws rds describe-db-engine-versions ^
--engine mysql ^
--engine-version 8.0.11 ^
--region us-east-2 ^
--query "DBEngineVersions[*].ValidUpgradeTarget[*].
{AutoUpgrade:AutoUpgrade,EngineVersion:EngineVersion}" ^
--output table
```

输出类似于以下内容。

```
-----------------------------------
|    DescribeDBEngineVersions     |
+--------------+------------------+
|  AutoUpgrade |  EngineVersion   |
+--------------+------------------+
|  False       |  8.0.15          |
|  False       |  8.0.16          |
|  False       |  8.0.17          |
|  False       |  8.0.19          |
|  False       |  8.0.20          |
|  False       |  8.0.21          |
|  True        |  8.0.23          |
|  False       |  8.0.25          |
+--------------+------------------+
```

在此示例中，AutoUpgrade 值对于 MySQL 版本 8.0.23 为 True。因此，自动次要升级目标是 MySQL 版本 8.0.23，它在输出中突出显示。

如果满足以下条件，则会在维护时段内自动升级 MySQL 数据库实例：

- 自动次要版本升级设置已启用。
- 备份保留期设置需大于 0。
- 数据库实例运行的次要数据库引擎版本低于当前的自动升级次要版本。

有关更多信息，请参阅“自动升级次要引擎版本”。

使用只读副本在升级 RDS for MySQL 数据库时减少停机时间

在大多数情况下，蓝绿部署是升级 MySQL 数据库实例时减少停机时间的最佳选项。有关更多信息，请参阅 使用 Amazon RDS 蓝绿部署进行数据库更新。

如果您无法使用蓝绿部署，并且 MySQL 数据库实例当前与生产应用程序结合使用，则可使用以下过程来升级数据库实例的数据库版本。此过程可以减少应用程序的停机时间。

使用只读副本，您可以提前执行大多数维护步骤，并在实际停机期间尽量减少必要的更改。利用此技巧，您可以在不对现有数据库实例进行任何更改的情况下测试和准备新的数据库实例。

以下过程显示了从 MySQL 5.7 版升级到 MySQL 8.0 版的示例。您可以使用相同的常规步骤升级到其他主要版本。您可以使用相同的常规步骤升级到其他主要版本。

> **Note**
>
> 当您从 MySQL 版本 5.7 升级到 MySQL 版本 8.0 或从 MySQL 版本 8.0 升级到 MySQL 版本 8.4 时，请在执行升级之前完成预检查。有关更多信息，请参阅从 MySQL 5.7 升级到 8.0 的预检查 和从 MySQL 8.0 升级到 8.4 的预检查。

在使用数据库实例时升级 MySQL 数据库

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 创建 MySQL 5.7 数据库实例的只读副本。该过程可创建数据库的可升级副本。可能还存在数据库实例的其他只读副本。

    a. 在控制台上，选择数据库，然后选择您要升级的数据库实例。
    b. 对于 Actions (操作)，选择 Create read replica (创建只读副本)。
    c. 为只读副本提供数据库实例标识符的值，并确保数据库实例类和其他设置与您的 MySQL 5.7 数据库实例匹配。
    d. 选择 Create read replica (创建只读副本)。
3. （可选）创建只读副本且状态显示为可用之后，将只读副本转换为多可用区部署并启用备份。

    默认情况下，创建的只读副本禁用了备份。由于只读副本最终成为生产数据库实例，因此最佳实践是配置多可用区部署并启用备份。

    a. 在控制台上，选择数据库，然后选择您刚创建的只读副本。
    b. 选择修改。
    c. 对于多可用区部署，请选择创建备用实例。
    d. 对于 Backup Retention Period（备份保留期），请选择正的非零值（如 3 天），然后选择 Continue（继续）。
    e. 对于修改计划，选择立即应用。
    f. 选择修改数据库实例。
4. 只读副本状态显示可用后，请将只读副本升级到 MySQL 8.0：

    a. 在控制台上，选择数据库，然后选择您刚创建的只读副本。
    b. 选择修改。
    c. 对于数据库引擎版本，选择升级目标版本 MySQL 8.0 版，然后选择继续。

   d. 对于修改计划，选择立即应用。

   e. 选择 Modify DB instance (修改数据库实例)开始升级。

5. 升级完成并且状态显示为可用之后，验证升级的只读副本是否与源 MySQL 5.7 数据库实例保持同步。要进行验证，请连接到只读副本并运行 `SHOW REPLICA STATUS` 命令。如果 `Seconds_Behind_Master` 字段为 `0`，则复制保持最新。

   > **Note**
   >
   > 以前的 MySQL 版本使用的是 `SHOW SLAVE STATUS`，而不是 `SHOW REPLICA STATUS`。如果您使用的 MySQL 版本低于 8.0.23，那么请使用 `SHOW SLAVE STATUS`。

6. （可选）创建只读副本的只读副本。

   如果您希望数据库实例在提升为独立数据库实例后拥有只读副本，则可以立即创建只读副本。

   a. 在控制台上，选择数据库，然后选择您刚刚升级的只读副本。

   b. 对于 Actions (操作)，选择 Create read replica (创建只读副本)。

   c. 为只读副本提供数据库实例标识符的值，并确保数据库实例类和其他设置与您的 MySQL 5.7 数据库实例匹配。

   d. 选择 Create read replica (创建只读副本)。

7. （可选）为只读副本配置自定义数据库参数组。

   如果您希望数据库实例在提升为独立数据库实例后使用自定义参数组，则可以立即创建数据库参数组，并将其与只读副本相关联。

   a. 为 MySQL 8.0 创建自定义数据库参数组。有关说明，请参阅在 Amazon RDS 中创建数据库参数组。

   b. 修改要在刚创建的数据库参数组中更改的参数。有关说明，请参阅在 Amazon RDS 中修改数据库参数组中的参数。

   c. 在控制台中，选择数据库，然后选择只读副本。

   d. 选择修改。

   e. 对于数据库参数组，请选择刚创建的 MySQL 8.0 数据库参数组，然后选择继续。

   f. 对于修改计划，选择立即应用。

   g. 选择 Modify DB instance (修改数据库实例)开始升级。

8. 使您的 MySQL 8.0 只读副本成为独立的数据库实例。

> **Important**
>
> 将 MySQL 8.0 只读副本提升为独立的数据库实例后，它不再是 MySQL 5.7 数据库实例的副本。建议您在源 MySQL 8.0 数据库实例处于只读模式并且所有写入操作都暂停的维护时段期间，提升 MySQL 5.7 只读副本。提升完成之后，您可以将写入操作定向到已升级的 MySQL 8.0 数据库实例，以确保不会丢失写入操作。
>
> 此外，建议您在提升 MySQL 8.0 只读副本之前，对 MySQL 8.0 只读副本执行所有必需的数据定义语言 (DDL) 操作。例如，创建索引。此方法可避免在 MySQL 8.0 只读副本提升之后对其性能造成任何负面影响。要提升只读副本，请使用以下过程。

a. 在控制台上，选择数据库，然后选择您刚刚升级的只读副本。
b. 对于操作，请选择提升。
c. 选择是，为只读副本实例启用自动备份。有关更多信息，请参阅“备份简介”。
d. 选择 Continue (继续)。
e. 选择 Promote Read Replica。

9. 您现在有了 MySQL 数据库的升级版本。此时，您可以将应用程序定向到新的 MySQL 8.0 数据库实例。

升级 MySQL 数据库快照引擎版本

借助 Amazon RDS，您可以为 MySQL 数据库实例创建存储卷数据库快照。创建数据库快照时，快照基于数据库实例使用的引擎版本。您可以升级数据库快照的引擎版本。

对于 RDS for MySQL，可以将版本 5.7 快照升级到版本 8.0，或将版本 8.0 快照升级到版本 8.4。您可以升级加密或未加密的数据库快照。

要查看 RDS for MySQL 数据库快照的可用引擎版本，请使用以下 Amazon CLI 示例。

```
aws rds describe-db-engine-versions --engine mysql --include-all --engine-
version example-engine-version --query "DBEngineVersions[*].ValidUpgradeTarget[*].
{EngineVersion:EngineVersion}" --output text
```

如果您看不到快照的结果，表明您的引擎版本可能已被弃用。如果您的引擎版本被弃用，我们建议您升级到最新的主要版本升级目标或该版本的其它可用升级目标之一。有关更多信息，请参阅 使用 RDS for MySQL 不支持的引擎版本的数据库快照的升级选项。

将升级到新引擎版本的数据库快照还原后，请确保测试升级是否成功。有关主要版本升级的更多信息，请参阅the section called “升级 MySQL 数据库引擎”。要了解如何还原数据库快照，请参阅 the section called “还原到数据库实例”。

> **Note**
>
> 您无法升级在自动备份过程中创建的自动数据库快照。

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 升级数据库快照。

Console

要使用 Amazon Web Services Management Console升级数据库快照引擎版本，请使用以下过程。

升级数据库快照

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择快照。
3. 选择要升级的快照。

4. 对于 Actions (操作)，选择 Upgrade snapshot (升级快照)。此时会显示 Upgrade snapshot (升级快照) 页面。
5. 选择要升级到的 New engine version (新引擎版本)。
6. 选择 Save changes (保存更改) 以升级快照。

   在升级过程中，将为此数据库快照禁用所有快照操作。此外，数据库快照状态由可用变为正在升级，完成后变为活动。如果由于快照损坏问题导致无法升级数据库快照，状态将变为不可用。此状态的快照无法恢复。

   > **Note**
   >
   > 如果数据库快照升级失败，则快照将回滚到具有原始版本的原始状态。

Amazon CLI

要将数据库快照升级到新的数据库引擎版本，请运行 Amazon CLI modify-db-snapshot 命令。

选项

- `--db-snapshot-identifier` – 要升级的数据库快照的标识符。标识符必须是唯一的 Amazon 资源名称 (ARN)。有关更多信息，请参阅"Amazon RDS 中的 Amazon 资源名称（ARN）"。
- `--engine-version` – 升级后的数据库快照的引擎版本。

Example

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-snapshot \

    --db-snapshot-identifier my_db_snapshot \
    --engine-version new_version
```

对于 Windows：

```
aws rds modify-db-snapshot ^
    --db-snapshot-identifier my_db_snapshot ^
    --engine-version new_version
```

Amazon RDS API

要将数据库快照升级到新的数据库引擎版本，请调用 RDS API ModifyDBSnapshot 操作。

参数

- `DBSnapshotIdentifier` – 要升级的数据库快照的标识符。标识符必须是唯一的 Amazon 资源名称 (ARN)。有关更多信息，请参阅“Amazon RDS 中的 Amazon 资源名称（ARN）”。
- `EngineVersion` – 升级后的数据库快照的引擎版本。

使用 RDS for MySQL 不支持的引擎版本的数据库快照的升级选项

下表显示了可以从 RDS for MySQL 数据库快照不支持的引擎版本升级到哪些引擎版本。

> **Note**
> 您可能需要多次升级数据库快照，才能升级到所选的引擎版本。

数据库快照引擎版本	可供升级的引擎版本
5.5.8	5.5.62、5.6.51
5.5.12	5.5.62、5.6.51
5.5.20	5.5.62、5.6.51
5.5.23	5.5.62、5.6.51
5.5.25a	5.5.62、5.6.51
5.5.27	5.5.62、5.6.51
5.5.31	5.5.62、5.6.51
5.5.33	5.5.62、5.6.51
5.5.37	5.5.62、5.6.51
5.5.38	5.5.62、5.6.51

数据库快照引擎版本	可供升级的引擎版本
5.5.40	5.5.62、5.6.51
5.5.40a	5.5.62、5.6.51
5.5.40b	5.5.62、5.6.51
5.5.41	5.5.62、5.6.51
5.5.42	5.5.62、5.6.51
5.5.59	5.5.62、5.6.51
5.6.12	5.6.51、5.7.44
5.6.13	5.6.51、5.7.44
5.6.17	5.6.51、5.7.44
5.6.19	5.6.51、5.7.44
5.6.19a	5.6.51、5.7.44
5.6.19b	5.6.51、5.7.44
5.6.21	5.6.51、5.7.44
5.6.21b	5.6.51、5.7.44
5.6.22	5.6.51、5.7.44
5.6.23	5.6.51、5.7.44
5.6.27	5.6.51、5.7.44
5.6.27a	5.6.51、5.7.44
5.7.10	5.7.44、5.7.44-rds.20240408、5.7.44-rds.20240529、5.7.44-rds.20250103、5.7.44-rds.20250213、8.0.32、8.0.33、8.0.34、8.0.35、8.0.36、8.0.37、8.0.39、8.0.40、8.0.41

数据库快照引擎版本	可供升级的引擎版本
5.7.11	5.7.44、5.7.44-rds.20240408、5.7.44-rds.20240529、5.7.44-rds.20250103、5.7.44-rds.20250213、8.0.32、8.0.33、8.0.34、8.0.35、8.0.36、8.0.37、8.0.39、8.0.40、8.0.41
5.7.12	5.7.44、5.7.44-rds.20240408、5.7.44-rds.20240529、5.7.44-rds.20250103、5.7.44-rds.20250213、8.0.32、8.0.33、8.0.34、8.0.35、8.0.36、8.0.37、8.0.39、8.0.40、8.0.41

将数据导入到 Amazon RDS for MySQL 数据库实例

您可以使用多种不同的技术将数据导入到 RDS for MySQL 数据库实例。最佳方法取决于多种因素：

- 数据的来源
- 数据量
- 一次性导入，还是持续导出
- 停机时间量

如果还要随数据一起迁移应用程序，则务必考虑停机时间量。

下表中列出了将数据导入到 RDS for MySQL 数据库实例的技巧：

来源	数据量	一次性或持续	应用程序停机时间	方法	更多信息
本地或 Amazon EC2 上的现有 MySQL 数据库	任何	一次性	一段时间	创建本地数据库的备份，将其存储在 Amazon S3 上，然后将该备份文件还原到运行 MySQL 的新 Amazon RDS 数据库实例上。	将备份还原到 Amazon RDS for MySQL 数据库实例
本地或 Amazon EC2 上的现有 MySQL 数据库	任何	持续	最低	配置复制，使用现有 MySQL 数据库作为复制源。	配置与外部源实例之间的二进制日志文件位置复制 将数据导入 Amazon

来源	数据量	一次性或持续	应用程序停机时间	方法	更多信息
					RDS for MySQL 数据库实例并减少停机时间
任何现有数据库	任何	一次性或持续	最低	使用 Amazon Database Migration Service 可在最少停机时间下迁移数据库，对于很多数据库引擎，将继续进行持续复制。	Amazon Database Migration Service 用户指南中的什么是 Amazon Database Migration Service 和将与 MySQL 兼容的数据库作为 Amazon DMS 的目标
现有 MySQL 数据库实例	任何	一次性或持续	最低	为正在进行的复制创建只读副本。提升只读副本用于一次性创建新数据库实例。	使用数据库实例只读副本

来源	数据量	一次性或持续	应用程序停机时间	方法	更多信息
现有 MySQL 数据库	小型	一次性	一段时间	使用命令行实用程序，将数据直接复制到您的 MySQL 数据库实例。	将数据从外部 MySQL 数据库导入到 Amazon RDS for MySQL 数据库实例
未存储在现有数据库中的数据	中	一次性	一段时间	使用 MySQL `LOAD DATA LOCAL INFILE` 语句创建平面文件并导入这些文件。	从任意源将数据导入 Amazon RDS for MySQL 数据库实例

> **Note**
>
> `mysql` 系统数据库包含登录到数据库实例并访问数据所需的身份验证和授权信息。在数据库实例中删掉、改动、重命名或截取 `mysql` 数据库中的表、数据或其他内容可能会导致错误，并可能导致无法访问数据库实例和数据。如果发生此情况，您可以使用 Amazon CLI restore-db-instance-from-db-snapshot 命令从快照中还原数据库实例。您可以使用 Amazon CLI restore-db-instance-to-point-in-time 命令恢复数据库实例。

为 MySQL 导入数据的注意事项

以下内容包含与将数据加载到 MySQL 中相关的技术信息。此内容面向熟悉 MySQL 服务器架构的用户。

二进制日志记录

与禁用日志记录相比，启用二进制日志记录会降低数据加载性能，并且需要多达四倍的额外磁盘空间。用于加载数据的事务大小直接影响系统性能和磁盘空间需求，也即，事务越大，需要的资源越多。

事务大小

事务大小会影响 MySQL 数据加载的以下方面：

- 资源消耗
- 磁盘空间利用率
- 恢复进程
- 恢复时间
- 输入格式（平面文件或 SQL）

本部分介绍了事务大小如何影响二进制日志记录以及让案例在大数据加载过程中禁用二进制日志记录。可以通过设置 Amazon RDS 自动备份保留期来启用和禁用二进制日志记录。非零值会启用二进制日志记录，零会将其禁用。有关更多信息，请参阅 备份保留期。

本节还介绍大型事务对 InnoDB 的影响以及将事务大小保持为较小的重要性。

小型事务

对于小型事务，二进制日志记录会使加载数据所需的磁盘写入次数翻倍。这种影响会严重降低其他数据库会话的性能，并增加数据加载所需的时间。所经历的降级在一定程度上取决于以下因素：

- 上传速率
- 在加载期间发生的其它数据库活动
- Amazon RDS 数据库实例的容量

此外，二进制日志占用的磁盘空间大致等于加载的数据量，直至备份和移除日志。Amazon RDS 通过频繁备份和移除二进制日志来尽量减少占用的磁盘空间量。

大型事务

对于大型事务，二进制日志记录会使 IOPS 和磁盘使用量增加两倍，原因如下：

- 二进制日志缓存将事务数据临时存储在磁盘上。
- 此缓存会随着事务大小而增长，这会占用磁盘空间。
- 当事务（提交或回滚）完成时，系统会将缓存复制到二进制日志。

此过程会创建数据的三个副本：

- 原始数据
- 磁盘上的缓存
- 最终二进制日志条目

每个写入操作都会产生额外的 IO，从而进一步影响性能。

因此，与禁用日志记录相比，二进制日志记录需要三倍的磁盘空间。例如，在单个事务中加载 10 GiB 的数据会创建三个副本：

- 表数据为 10 GiB
- 二进制日志缓存为 10 GiB
- 二进制日志文件为 10 GiB

所需的临时磁盘空间总量为 30 GiB。

磁盘空间重要注意事项：

- 缓存文件会一直存在，直到会话结束或新事务创建另一个缓存。
- 二进制日志会保留直至对其进行备份，可能会在很长一段时间内占用 20 GiB（缓存和日志）。

如果您使用 `LOAD DATA LOCAL INFILE` 来加载数据，数据恢复会创建第四个副本，以防数据库必须从加载前进行的备份中恢复。在恢复期间，MySQL 将二进制日志中的数据提取到平面文件中。然后，MySQL 运行 `LOAD DATA LOCAL INFILE`。在前一个示例的基础上，此恢复需要的临时磁盘空间总量为 40 GiB，也即表、缓存、日志和本地文件各为 10 GiB。如果没有至少 40 GiB 的可用磁盘空间，则恢复将失败。

优化大数据加载

对于大数据加载，请禁用二进制日志记录，以减少开销和磁盘空间要求。可以通过将备份保留期设置为 0 来禁用二进制日志记录。完成加载后，将备份保留期还原为适当的非零值。有关更多信息，请参阅修改 Amazon RDS 数据库实例和设置表中的备份保留期。

> **Note**
> 如果数据库实例是只读副本的源数据库实例，则您无法将备份保留期设置为 0。

在加载数据之前，我们建议您创建数据库快照。有关更多信息，请参阅 管理手动备份。

InnoDB

以下有关撤消日志记录和恢复选项的信息支持将 InnoDB 事务保持为较小，以优化数据库性能。

了解 InnoDB 撤消日志记录

撤消是一种日志记录机制，可启用事务回滚并支持多版本并发控制（MVCC）。

对于 MySQL 5.7 及更低版本，撤消日志存储在 InnoDB 系统表空间（通常为 ibdata1）中，并保留至清除线程移除它们为止。因此，大数据加载事务可能会导致系统表空间变得相当大，并占用磁盘空间，除非重新创建数据库，否则无法回收这些磁盘空间。

对于所有 MySQL 版本，清除线程必须等待以移除所有撤消日志，直到最旧的活动事务提交或回滚。如果数据库在加载过程中正在处理其它事务，则其撤消日志也会累积并且无法移除，即使事务提交并且没有其它事务需要 MVCC 的撤消日志，也是如此。在这种情况下，所有事务（包括只读事务）都会变慢。之所以发生这种变慢过程，是因为所有事务都访问任何事务（而不仅仅是加载事务）更改的所有行。实际上，事务必须扫描一些撤消日志，在撤消日志清理期间，长时间运行的加载事务已阻止清除这些撤消日志。这会影响访问已修改行的任何操作的性能。

InnoDB 事务恢复选项

尽管 InnoDB 优化了提交操作，但大型事务回滚速度很慢。为了加快恢复速度，请执行时间点故障恢复或还原数据库快照。有关更多信息，请参阅时间点故障恢复和还原到数据库实例。

数据导入格式

MySQL 支持两种数据导入格式：平面文件和 SQL。查看有关每种格式的信息，以确定最适合您需求的选项。

平面文件

对于小型事务，请使用 `LOAD DATA LOCAL INFILE` 加载平面文件。与使用 SQL 相比，这种数据导入格式具有以下优势：

- 减少网络流量
- 降低数据传输成本
- 减少了数据库处理开销
- 处理速度更快

`LOAD DATA LOCAL INFILE` 会将整个平面文件作为一个事务进行加载。将各个文件的大小保持为较小可获得以下优势：

- 恢复功能：可以保持跟踪哪些文件已加载。如果在加载期间出现问题，可以从停止处继续加载。可能需要将一些数据重新传输到 Amazon RDS，但使用小文件，就可以将重新传输的量减少至最低水平。
- 并行数据加载：如果 IOPS 和网络带宽足以支持单个文件加载，那么并行加载就可以节省时间。
- 加载速率控制：如果数据加载对其它进程产生负面影响，则可以通过增加文件之间的间隔来控制加载速率。

大型事务会降低使用 `LOAD DATA LOCAL INFILE` 导入数据的优势。当您无法将大量数据分解成较小的文件时，可以考虑使用 SQL。

SQL

与平面文件相比，SQL 的一个主要优势为：您可以轻松地将事务大小保持为较小。但是，SQL 的加载时间可能比平面文件长得多。此外，在发生故障后，可能难以确定从哪里恢复，而您无法重新启动 mysqldump 文件。如果加载 mysqldump 文件时出现故障，您必须修改或替换该文件，然后才能恢复加载。或者，在纠正故障的原因后，您可以恢复到加载前的时间点，并重新发送该文件。有关更多信息，请参阅 时间点故障恢复。

使用 Amazon RDS 数据库快照提供数据库检查点

如果您在没有二进制日志记录的情况下长时间（如数小时或数天）加载数据，请使用数据库快照提供定期检查点来实现数据安全。每个数据库快照都会创建数据库实例的一致副本，该副本在系统故障或数据损坏事件期间用作恢复点。由于数据库快照速度很快，因此频繁的检查点对加载性能的影响微乎其微。您可以在不影响数据库持久性或恢复能力的情况下，删除以前的数据库快照。有关数据库快照的更多信息，请参阅管理手动备份。

减少数据库加载时间

以下各项是减少加载时间的其它技巧：

- 在将数据加载到 MySQL 数据库之前，请创建所有二级索引。与其它数据库系统不同，MySQL 在添加或修改二级索引时会重建整个表。此过程会创建一个包含索引更改的新表，复制所有数据，然后删除原始表。
- 按主键顺序加载数据。对于 InnoDB 表，这可以将加载时间缩短 75% 至 80%，并将数据文件大小减少 50%。
- 通过将 `foreign_key_checks` 设置为 `0` 来禁用外键约束。对于使用 `LOAD DATA LOCAL INFILE` 加载的平面文件，这通常是必需的。对于任何加载，禁用外键检查会加快数据加载速度。加载完成后，通过将 `foreign_key_checks` 设置为 `1` 来重新启用约束并验证数据。
- 并行加载数据，除非接近资源限制。要实现跨多个表段的并行加载，请在适当时使用分区表。
- 要减少 SQL 执行开销，请将多条 `INSERT` 语句合并为单个多值 `INSERT` 操作。`mysqldump` 自动实现此优化。
- 通过将 `innodb_flush_log_at_trx_commit` 设置为 `0` 来减少 InnoDB 日志 IO 操作。加载完成后，将 `innodb_flush_log_at_trx_commit` 还原为 `1`。

> **Warning**
>
> 将 `innodb_flush_log_at_trx_commit` 设置为 `0` 会导致 InnoDB 每秒都刷新其日志，而不是在每次提交时刷新。此设置可以提高性能，但在系统故障期间可能会有事务丢失的风险。

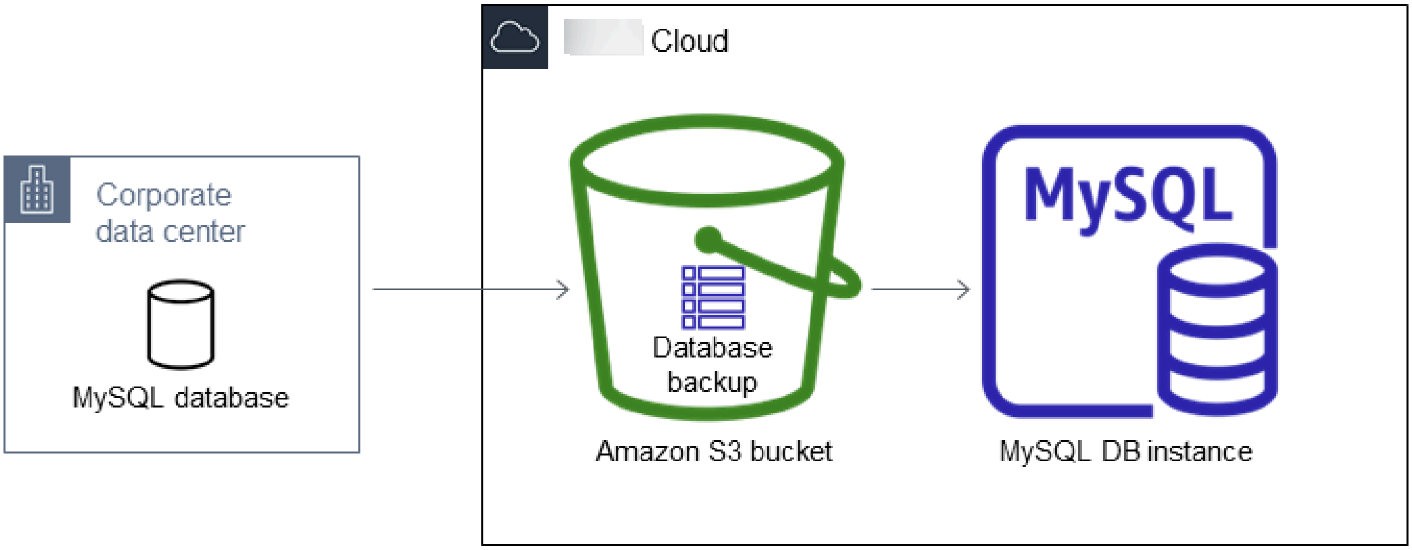
- 如果您要将数据加载到没有只读副本的数据库实例，请将 `sync_binlog` 设置为 `0`。加载完成后，将 `sync_binlog parameter` 还原为 `1`。
- 在将数据库实例转换为多可用区部署之前，将数据加载到单可用区实例中。如果数据库实例已经使用多可用区部署，我们建议您不要切换为单可用区部署来加载数据。这样做所带来的改进微不足道。

将备份还原到 Amazon RDS for MySQL 数据库实例

Amazon RDS 支持使用备份文件导入 MySQL 数据库。您可以创建数据库的备份，将备份文件存储在 Amazon S3 上，然后将该备份文件还原到运行 MySQL 的新 Amazon RDS 数据库实例上。Amazon RDS 支持在所有 Amazon Web Services 区域中从 Amazon S3 导入备份文件。

本节中介绍的方案将还原本地数据库的备份。只要数据库可供访问，您就可以将此技术用于其它位置（例如 Amazon EC2 或其它云服务）中的数据库。

下图显示了支持的方案。



如果在创建、复制和还原备份文件时，您的本地数据库可能处于脱机状态，则我们建议您使用备份文件将数据库导入到 Amazon RDS。如果数据库无法脱机，则可以使用以下方法之一：

- 二进制日志：首先，将备份文件从 Amazon S3 导入到 Amazon RDS，如本主题所述。然后，使用二进制日志（binlog）复制来更新数据库。有关更多信息，请参阅 配置与外部源实例之间的二进制日志文件位置复制。
- Amazon Database Migration Service：使用 Amazon Database Migration Service 将数据库迁移到 Amazon RDS。有关更多信息，请参阅什么是 Amazon Database Migration Service？

将备份文件从 Amazon S3 导入到 Amazon RDS 的设置概述

要将备份文件从 Amazon S3 导入到 Amazon RDS，您需要以下组件：

- 用于存储备份文件的 Amazon S3 存储桶。

  如果您已有一个 Amazon S3 存储桶，则可以使用该存储桶。如果没有 Amazon S3 存储桶，请创建一个新的存储桶。有关更多信息，请参阅创建存储桶。
- Percona XtraBackup 创建的本地数据库备份。

  有关更多信息，请参阅 创建数据库备份。
- Amazon Identity and Access Management（IAM）角色，支持 Amazon RDS 访问 S3 存储桶。

  如果您已有 IAM 角色，则可以使用该角色并向其附加信任和权限策略。有关更多信息，请参阅 手动创建 IAM 角色。

如果您没有 IAM 角色，则有两个选项：

- 您可以手动创建新的 IAM 角色。有关更多信息，请参阅 手动创建 IAM 角色。
- 您可以选择让 Amazon RDS 为您创建新的 IAM 角色。如果您希望 Amazon RDS 为您创建新的 IAM 角色，请按照将数据从 Amazon S3 导入到新的 MySQL 数据库实例一节中使用 Amazon Web Services Management Console的过程进行操作。

创建数据库备份

可以使用 Percona XtraBackup 软件创建备份。建议您使用最新的 Percona XtraBackup 版本。您可以通过 Percona 网站上的 Software Downloads 安装 Percona XtraBackup。

> **Warning**
>
> 创建数据库备份时，XtraBackup 可能会将凭证保存在 xtrabackup_info 文件中。务必确认 xtrabackup_info 文件中的 `tool_command` 设置不包含任何敏感信息。

您使用的 Percona XtraBackup 版本取决于您要备份的 MySQL 版本。

- MySQL 8.4：使用 Percona XtraBackup 版本 8.4。
- MySQL 8.0：使用 Percona XtraBackup 版本 8.0。

> **Note**
>
> Percona XtraBackup 8.0.12 及更高版本支持迁移所有 MySQL 8.0 版本。如果要迁移到 RDS for MySQL 8.0.32 或更高版本，则必须使用 Percona XtraBackup 8.0.12 或更高版本。

- MySQL 5.7：使用 Percona XtraBackup 版本 2.4。

您可以使用 Percona XtraBackup 来创建 MySQL 数据库文件的完整备份。或者，如果已使用 Percona XtraBackup 备份 MySQL 数据库文件，您可以上传现有的完整和增量备份目录和文件。

有关使用 Percona XtraBackup 备份数据库的更多信息，请参阅 Percona 网站上的 Percona XtraBackup - Documentation。

使用 Percona XtraBackup 创建完整备份

要为 Amazon RDS 可从 Amazon S3 中还原的 MySQL 数据库文件创建完整备份，请使用 Percona XtraBackup 实用程序 (`xtrabackup`)。

例如，以下命令创建 MySQL 数据库备份，并将这些文件存储在 `/on-premises/s3-restore/backup` 文件夹中。

```
xtrabackup --backup --user=myuser --password=password --target-dir=/on-premises/s3-
restore/backup
```

如果您想将备份压缩到单个文件中（如果需要，您可以稍后将其拆分为多个文件），可以根据 MySQL 版本将备份保存为以下格式之一：

- Gzip (.gz)：适用于 MySQL 5.7 及更低版本
- tar (.tar)：适用于 MySQL 5.7 及更低版本
- Percona xbstream (.xbstream)：适用于所有 MySQL 版本

> **Note**
> Percona XtraBackup 8.0 及更高版本仅支持使用 Percona xbstream 进行压缩。

MySQL 5.7 及更低版本

以下命令为您的 MySQL 数据库创建一个拆分成多个 Gzip 文件的备份。将值替换为您自己的信息。

```
xtrabackup --backup --user=my_user --password=password --stream=tar \
   --target-dir=/on-premises/s3-restore/backup | gzip - | split -d --bytes=500MB \
   - /on-premises/s3-restore/backup/backup.tar.gz
```

MySQL 5.7 及更低版本

以下命令为您的 MySQL 数据库创建一个拆分成多个 tar 文件的备份。将值替换为您自己的信息。

```
xtrabackup --backup --user=my_user --password=password --stream=tar \
   --target-dir=/on-premises/s3-restore/backup | split -d --bytes=500MB \
   - /on-premises/s3-restore/backup/backup.tar
```

所有 MySQL 版本

以下命令为您的 MySQL 数据库创建一个拆分成多个 xbstream 文件的备份。将值替换为您自己的信息。

```
xtrabackup --backup --user=myuser --password=password --stream=xbstream \
   --target-dir=/on-premises/s3-restore/backup | split -d --bytes=500MB \
   - /on-premises/s3-restore/backup/backup.xbstream
```

> **Note**
>
> 如果您看到以下错误，则可能是由于命令中混用了文件格式所致：
>
> ```
> ERROR:/bin/tar: This does not look like a tar archive
> ```

通过 Percona XtraBackup 使用增量备份

如果您已使用 Percona XtraBackup 对 MySQL 数据库文件进行完整备份和增量备份，则无需创建完整备份和将备份文件上传到 Amazon S3。相反，为了节省时间，请将现有的备份目录和文件复制到 Amazon S3 存储桶中。有关使用 Percona XtraBackup 创建增量备份的更多信息，请参阅 Percona 网站上的 [Create an incremental backup]。

在将现有的完整备份和增量备份文件复制到 Amazon S3 存储桶时，您必须以递归方式复制基目录的内容。这些内容同时包括完整备份以及所有增量备份目录和文件。此副本必须在 Amazon S3 存储桶中保留目录结构。Amazon RDS 将循环访问所有文件和目录。Amazon RDS 使用每个增量备份中包含的 `xtrabackup-checkpoints` 文件来标识基本目录，并按日志序列号（LSN）范围对增量备份进行排序。

Percona XtraBackup 的备份注意事项

Amazon RDS 根据文件名使用您的备份文件。根据文件格式使用相应的文件扩展名来命名备份文件。例如，使用 `.xbstream` 来表示使用 Percona xbstream 格式存储的文件。

Amazon RDS 按照字母顺序以及自然数字顺序使用您的备份文件。为确保按正确的顺序写入和命名备份文件，请在发出 `xtrabackup` 命令时使用 `split` 选项。

Amazon RDS 不支持使用 Percona XtraBackup 创建的部分备份。在备份数据库的源文件时，不能使用以下选项创建部分备份：

- `--tables`
- `--tables-exclude`
- `--tables-file`
- `--databases`
- `--databases-exclude`
- `--databases-file`

手动创建 IAM 角色

如果您没有 IAM 角色，则可以手动创建一个新的角色。但是，如果您使用 Amazon Web Services Management Console还原数据库，我们建议您选择让 Amazon RDS 为您创建这个新的 IAM 角色。要让 Amazon RDS 为您创建此角色，请按照将数据从 Amazon S3 导入到新的 MySQL 数据库实例一节中的过程进行操作。

要手动创建新的 IAM 角色以从 Amazon S3 导入数据库，请创建一个角色，以便将权限从 Amazon RDS 委托给 Amazon S3 存储桶。在创建 IAM 角色时，可附加信任和权限策略。要从 Amazon S3 中导入备份文件，请使用类似于以下示例的信任和权限策略。有关创建角色的更多信息，请参阅创建将权限委派给 Amazon 服务的角色。

信任和权限策略要求您提供 Amazon Resource Name (ARN)。有关 ARN 格式的更多信息，请参阅 Amazon Resource Name (ARN) 和 Amazon 服务命名空间。

Example 用于从 Amazon S3 导入的信任策略

```
{
    "Version": "2012-10-17",
    "Statement":
    [{
        "Effect": "Allow",
        "Principal": {"Service":  "rds.amazonaws.com"},
        "Action": "sts:AssumeRole"
    }]
}
```

Example 用于从 Amazon S3 导入的权限策略 – IAM 用户权限

在以下示例中，将 *iam_user_id* 替换为您自己的值。

```
{
    "Version":"2012-10-17",
    "Statement":
    [
        {
            "Sid":"AllowS3AccessRole",
            "Effect":"Allow",
            "Action":"iam:PassRole",
            "Resource":"arn:aws:iam::iam_user_id:role/S3Access"
        }
    ]
}
```

Example 用于从 Amazon S3 导入的权限策略 – 角色权限

在以下示例中，将 *amzn-s3-demo-bucket* 和 *prefix* 替换为您自己的值。

```
{
    "Version": "2012-10-17",
    "Statement":
    [
        {
        "Effect": "Allow",
        "Action":
            [
                "s3:ListBucket",
                "s3:GetBucketLocation"
            ],
        "Resource": "arn:aws:s3:::amzn-s3-demo-bucket"
        },
        {
        "Effect": "Allow",
        "Action":
            [
                "s3:GetObject"
            ],
        "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/prefix*"
        },
        { // If your bucket is encrypted, include the following permission. This
 permission allows decryption of your Amazon KMS key.
        "Effect": "Allow",
        "Action":
            [
```

```
                "kms:Decrypt"
            ],
        "Resource": [
            "arn:aws:kms:region:customer_id:key/key_id*"
            ]
        }
    ]
}
```

> **Note**
>
> 如果包含文件名前缀，请在前缀后面包含星号 (*)。如果您不想指定前缀，请仅指定星号。

将数据从 Amazon S3 导入到新的 MySQL 数据库实例

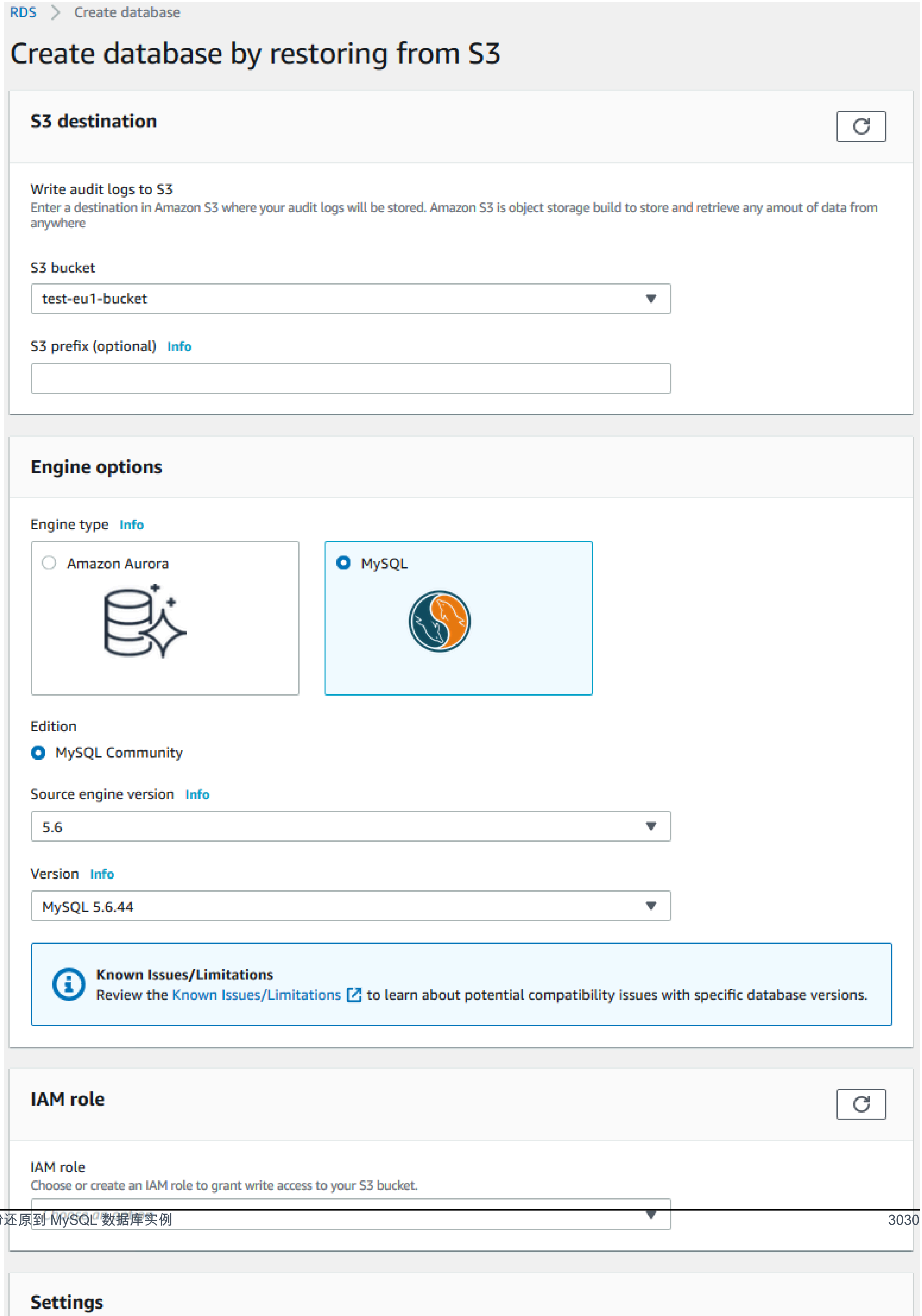
您可以使用Amazon Web Services Management Console、Amazon CLI 或 RDS API 将数据从 Amazon S3 导入到新的 MySQL 数据库实例。

控制台

将数据从 Amazon S3 导入到新的 MySQL 数据库实例

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在 Amazon RDS 控制台的右上角，选择要在其中创建数据库实例的 Amazon Web Services 区域。选择与包含数据库备份的 Amazon S3 桶相同的 Amazon Web Services 区域。
3. 在导航窗格中，选择数据库。
4. 选择从 S3 还原。

   此时将显示通过从 S3 还原创建数据库页面。


RDS > Create database
Create database by restoring from S3
S3 destination
Write audit logs to S3
Enter a destination in Amazon S3 where your audit logs will be stored. Amazon S3 is object storage build to store and retrieve any amout of data from anywhere
S3 bucket
test-eu1-bucket
S3 prefix (optional) Info
Engine options
Engine type Info
Amazon Aurora
MySQL
Edition
MySQL Community
Source engine version Info
5.6
Version Info
MySQL 5.6.44
Known Issues/Limitations
Review the Known Issues/Limitations to learn about potential compatibility issues with specific database versions.
IAM role
IAM role
Choose or create an IAM role to grant write access to your S3 bucket.
Settings

5. 在 S3 源下：

   a. 选择包含备份的 S3 桶。

   b. （可选）对于 S3 前缀，输入存储在 Amazon S3 存储桶中的文件的文件路径前缀。

      如果未指定前缀，则 Amazon RDS 使用 S3 存储桶的根文件夹中的所有文件和文件夹创建数据库实例。如果的确指定了前缀，则 Amazon RDS 使用 S3 存储桶中文件路径以指定前缀开头的文件和文件夹来创建数据库实例。

      例如，您将备份文件存储在 S3 上名为 backups 的子文件夹中，并且具有多组备份文件，每个文件位于其自己的目录（gzip_backup1、gzip_backup2，依此类推）中。在这种情况下，要从 gzip_backup1 文件夹中的文件进行还原，请指定前缀 backups/gzip_backup1。

6. 在引擎选项下：

   a. 对于引擎类型，选择 MySQL。

   b. 对于源引擎版本，请选择源数据库的 MySQL 主版本。

   c. 对于引擎版本，选择您所在Amazon Web Services 区域中 MySQL 主要版本的默认次要版本。

      在Amazon Web Services Management Console中，仅默认的次要版本可用。完成导入后，您可以升级数据库实例。

7. 对于 IAM 角色，创建或选择具有所需信任策略和权限策略的 IAM 角色，即允许 Amazon RDS 访问您的 Amazon S3 存储桶。执行以下操作之一：

   - （推荐）选择创建新角色，然后输入 IAM 角色名称。选择此选项后，Amazon RDS 会自动为您创建具有信任策略和权限策略的角色。
   - 选择现有 IAM 角色。确保此角色符合the section called “手动创建 IAM 角色”中的所有条件。

8. 指定数据库实例信息。有关每项设置的信息，请参阅 数据库实例的设置。

   > **Note**
   >
   > 确保为新的数据库实例分配足够的存储空间，以便成功进行还原操作。
   > 要支持未来自动增长，请在其他存储配置下选择启用存储自动扩缩。

9. 根据需要选择其他设置。

10. 选择创建数据库。

Amazon CLI

要使用 Amazon CLI 将数据从 Amazon S3 导入新的 MySQL 数据库实例，请使用以下选项运行 restore-db-instance-from-s3 命令。有关每项设置的信息，请参阅 数据库实例的设置。

> **Note**
>
> 确保为新的数据库实例分配足够的存储空间，以便成功进行还原操作。
> 要启用存储自动扩缩并支持未来自动增长，请使用 `--max-allocated-storage` 选项。

- `--allocated-storage`
- `--db-instance-identifier`
- `--db-instance-class`
- `--engine`
- `--master-username`
- `--manage-master-user-password`
- `--s3-bucket-name`
- `--s3-ingestion-role-arn`
- `--s3-prefix`
- `--source-engine`
- `--source-engine-version`

Example

对于 Linux、macOS 或 Unix：

```
aws rds restore-db-instance-from-s3 \
    --allocated-storage 250 \
    --db-instance-identifier my_identifier \
    --db-instance-class db.m5.large \
    --engine mysql \
    --master-username admin \
    --manage-master-user-password \
    --s3-bucket-name amzn-s3-demo-bucket \
    --s3-ingestion-role-arn arn:aws:iam::account-number:role/rolename \
    --s3-prefix bucket_prefix \
    --source-engine my_sql \
```

```
    --source-engine-version 8.0.32 \
    --max-allocated-storage 1000
```

对于 Windows：

```
aws rds restore-db-instance-from-s3 ^
    --allocated-storage 250 ^
    --db-instance-identifier my_identifier ^
    --db-instance-class db.m5.large ^
    --engine mysql ^
    --master-username admin ^
    --manage-master-user-password ^
    --s3-bucket-name amzn-s3-demo-bucket ^
    --s3-ingestion-role-arn arn:aws:iam::account-number:role/rolename ^
    --s3-prefix bucket_prefix ^
    --source-engine mysql ^
    --source-engine-version 8.0.32 ^
    --max-allocated-storage 1000
```

RDS API

要使用 Amazon RDS API 将数据从 Amazon S3 导入新的 MySQL 数据库实例，请调用 RestoreDBInstanceFromS3 操作。

将备份文件从 Amazon S3 导入到 Amazon RDS 的限制和注意事项

以下限制和注意事项适用于将备份文件从 Amazon S3 导入到 RDS for MySQL 数据库实例：

- 您只能将数据迁移到新的数据库实例，而不能迁移到现有的数据库实例。
- 您必须使用 Percona XtraBackup 将数据备份到 Amazon S3。有关更多信息，请参阅 创建数据库备份。
- Amazon S3 存储桶和 RDS for MySQL 数据库实例必须在同一个 Amazon Web Services 区域中。
- 您无法从以下来源进行还原：
  - 到 Amazon S3 的数据库实例快照导出。您也无法将数据从数据库实例快照导出迁移到 Amazon S3 存储桶。
  - 加密的源数据库。但是，您可以加密正在迁移的数据。您也可以在迁移过程中不加密数据。
  - MySQL 5.5 或 5.6 数据库。
- RDS for MySQL 不支持将 Percona Server for MySQL 用作源数据库，因为它可能在 `mysql` schema 中包含 `compression_dictionary*` 表。

- RDS for MySQL 不支持主要版本或次要版本的反向迁移。例如，您无法从 MySQL 版本 8.0 迁移到 RDS for MySQL 5.7，也无法从 MySQL 版本 8.0.32 迁移到 RDS for MySQL 版本 8.0.26。
- Amazon RDS 不支持从 Amazon S3 中的 db.t2.micro 数据库实例类上进行导入。不过，您可以还原到不同的数据库实例类，然后更改该数据库实例类。有关实例类的更多信息，请参阅数据库实例类的硬件规格。
- Amazon S3 将上传到 Amazon S3 存储桶的文件大小限制为 5 TB。如果备份文件超过 5 TB，则必须将备份文件拆分为较小的文件。
- Amazon RDS 将上传到 Amazon S3 存储桶的文件数限制为一百万个。如果数据库的备份数据 (包括所有完整和增量备份) 超过 100 万个文件，请使用 Gzip (.gz)、tar (.tar.gz) 或 Percona xbstream (.xbstream) 文件将完整和增量备份文件存储在 Amazon S3 存储桶中。Percona XtraBackup 8.0 仅支持使用 Percona xbstream 进行压缩。
- 要为每个数据库实例提供管理服务，Amazon RDS 会在创建数据库实例时创建 `rdsadmin` 用户。由于 `rdsamin` 是 Amazon RDS 中的保留用户，因此以下限制适用：
  - Amazon RDS 不导入以 `'rdsadmin'@'localhost'` 作为定义者的函数、过程、视图、事件和触发器。有关更多信息，请参阅以 'rdsamin'@'localhost' 作为定义者的存储对象和主用户账户权限。
  - 创建数据库实例时，Amazon RDS 会创建一个具有最大受支持权限的主用户。从备份还原时，Amazon RDS 会自动移除分配给正在导入的用户的任何不受支持的权限。

    要识别可能受此影响的用户，请参阅具有不支持的权限的用户账户。有关 RDS for MySQL 中支持的权限的更多信息，请参阅 RDS for MySQL 的基于角色的权限模型。
- Amazon RDS 不会迁移 `mysql` 架构中用户创建的表。
- 您必须仅通过一个使用默认数据文件名 `ibdata1:12M:autoextend` 的数据文件来配置 `innodb_data_file_path` 参数。可以使用此方法迁移具有两个数据文件或具有不同名称的数据文件的数据库。

  以下示例是 Amazon RDS 不支持的文件名：
  - `innodb_data_file_path=ibdata1:50M`
  - `ibdata2:50M:autoextend`
  - `innodb_data_file_path=ibdata01:50M:autoextend`
- 您无法从在默认 MySQL 数据目录外部定义表的源数据库中迁移。
- 使用此方法的未压缩备份支持的最大大小限制为 64 TiB。对于压缩的备份，考虑到解压缩空间的需求，此限制会降低。在此类情况下，支持的最大备份大小为 `64 TiB - compressed backup size`。

有关 RDS for MySQL 支持的最大数据库大小的信息，请参阅通用型 SSD 存储和预置 IOPS SSD 存储。

- Amazon RDS 不支持导入 MySQL 以及其它外部组件和插件。
- Amazon RDS 不会从您的数据库中还原所有内容。我们建议您从源 MySQL 系统数据库保存数据库架构以及以下项目的值，然后将这些内容添加到已还原的 RDS for MySQL 数据库实例（在创建该实例后）：
  - 用户账户
  - 函数
  - 存储过程
  - 时区信息。从 RDS for MySQL 数据库实例的本地操作系统中加载时区信息。有关更多信息，请参阅 MySQL 数据库实例的本地时区。

以 'rdsamin'@'localhost' 作为定义者的存储对象

Amazon RDS 不导入以 `'rdsadmin'@'localhost'` 作为定义者的函数、过程、视图、事件和触发器。

您可以在源 MySQL 数据库上使用以下 SQL 脚本来列出具有不受支持的定义程序的存储对象。

```
-- This SQL query lists routines with `rdsadmin`@`localhost` as the definer.

SELECT
    ROUTINE_SCHEMA,
    ROUTINE_NAME
FROM
    information_schema.routines
WHERE
    definer = 'rdsadmin@localhost';

-- This SQL query lists triggers with `rdsadmin`@`localhost` as the definer.

SELECT
    TRIGGER_SCHEMA,
    TRIGGER_NAME,
    DEFINER
FROM
    information_schema.triggers
WHERE
    DEFINER = 'rdsadmin@localhost';
```

```
-- This SQL query lists events with `rdsadmin`@`localhost` as the definer.

SELECT
    EVENT_SCHEMA,
    EVENT_NAME
FROM
    information_schema.events
WHERE
    DEFINER = 'rdsadmin@localhost';

-- This SQL query lists views with `rdsadmin`@`localhost` as the definer.
SELECT
    TABLE_SCHEMA,
    TABLE_NAME
FROM
    information_schema.views
WHERE
    DEFINER = 'rdsadmin@localhost';
```

具有不支持的权限的用户账户

如果用户账户的权限不受 RDS for MySQL 支持，则在导入这些账户时将不包含这些不受支持的权限。有关支持的权限列表，请参阅RDS for MySQL 的基于角色的权限模型。

可以在源数据库上运行以下 SQL 查询，以列出具有不受支持的权限的用户账户。

```
SELECT
    user,
    host
FROM
    mysql.user
WHERE
    Shutdown_priv = 'y'
    OR File_priv = 'y'
    OR Super_priv = 'y'
    OR Create_tablespace_priv = 'y';
```

将数据从外部 MySQL 数据库导入到 Amazon RDS for MySQL 数据库实例

您可以将数据从现有 MariaDB 或 MySQL 数据库导入到 RDS for MariaDB 或 RDS for MySQL 数据库实例。为此，请使用 mysqldump 复制数据库，然后通过管道将其直接传输到 RDS for MariaDB

或 RDS for MySQL 数据库实例。mysqldump 命令行实用程序通常用于创建备份以及将数据从一个 MariaDB 或 MySQL 服务器传输到另一个 MariaDB 或 MySQL 服务器。该实用程序包含在 MySQL 和 MariaDB 客户端软件中。

在 MariaDB 10.5 中，客户端被称为 mariadb-dump。从 MariaDB 11.0.1 开始，您必须使用 mariadb-dump 而不是 mysqldump。

> **Note**
>
> 如果您使用 MySQL 数据库实例导入或导出大量数据，则可以使用 xtrabackup 备份文件和 Amazon S3 更可靠、更块地将数据移入和移出 Amazon RDS。有关更多信息，请参阅 将备份还原到 Amazon RDS for MySQL 数据库实例。
> Amazon RDS 不支持 MariaDB 的 xtrabackup 或 mariabackup。Amazon RDS 仅支持从 Amazon S3 为 MySQL 导入。

将数据从外部数据库移动到 Amazon RDS 数据库实例的典型 mysqldump 命令类似以下示例。将值替换为您自己的信息。对于 MariaDB 11.0.1 及更高版本，请将 mysqldump 替换为 mariadb-dump，并将 mysql 替换为 mariadb。

```
mysqldump -u local_user \
    --databases database_name \
    --single-transaction \
    --compress \
    --order-by-primary  \
    --routines=0 \
    --triggers=0 \
    --events=0 \
    -plocal_password | mysql -u RDS_user \
        --port=port_number \
        --host=host_name \
        -pRDS_password
```

> **Important**
>
> 确保 -p 选项和输入的密码之间不留空格。
> 作为安全最佳实践，请指定此示例中显示的提示以外的凭证。

确保您了解以下建议和注意事项：

- 从转储文件中排除以下架构：
  - sys
  - performance_schema
  - information_schema

  默认情况下，mysqldump 和 mariadb-dump 实用程序排除这些架构。
- 如果您需要迁移用户和权限，请考虑使用生成数据控制语言 (DCL) 的工具 (如 pt-show-grants 实用程序) 来重新创建用户和权限。
- 要执行导入，请确保执行此操作的用户有权访问数据库实例。有关更多信息，请参阅 使用安全组控制访问权限。

所用参数如下所示：

- `-u` *`local_user`* – 用于指定用户名。在第一次使用该参数时，请指定您使用 `--databases` 参数确定的本地 MariaDB 或 MySQL 数据库中的用户账户名称。
- `--databases` *`database_name`* – 用于指定本地 MariaDB 或 MySQL 实例上您要导入 Amazon RDS 的数据库的名称。
- `--single-transaction` – 用于确保从本地数据库加载的所有数据都与单一时间点保持一致。如果在 mysqldump 读取数据期间有其他进程更改数据，使用该参数有助于保持数据完整性。
- `--compress` – 用于降低网络带宽消耗，方式为将数据从本地数据库发送到 Amazon RDS 之前压缩数据。
- `--order-by-primary` – 用于减少加载时间，方式为根据主键对每个表中的数据进行排序。
- `--routines`：如果您要复制的数据库中存在诸如存储过程或函数之类的例程，则使用此参数。将此参数设置为 `0`，这将在导入过程中排除例程。稍后在 Amazon RDS 数据库中手动重新创建例程。
- `--triggers`：如果要复制的数据库中存在触发器，则使用此参数。将此参数设置为 `0`，这将在导入过程中排除触发器。稍后在 Amazon RDS 数据库中手动重新创建触发器。
- `--events`：如果要复制的数据库中存在事件，则使用此参数。将此参数设置为 `0`，这将在导入过程中排除事件。稍后在 Amazon RDS 数据库中手动重新创建事件。
- `-p`*`local_password`* – 用于指定密码。在第一次使用该参数时，请为您使用第一个 -u 参数标识的用户账户指定密码。
- `-u` *`RDS_user`* – 用于指定用户名。在第二次使用该参数时，请为您使用 `--host` 参数标识的 MariaDB 或 MySQL 数据库实例指定默认数据库上的用户账户的名称。
- `--port` *`port_number`* – 用于为您的 MariaDB 或 MySQL 数据库实例指定端口。默认情况下该值为 3306，除非您在创建数据库实例时更改了该值。

- `--host` *`host_name`* – 用于从 Amazon RDS 数据库实例端点指定域名系统 (DNS) 名称，例如，`myinstance.123456789012.us-east-1.rds.amazonaws.com`。您可以在 Amazon RDS 控制台上的数据库实例详细信息中找到端点值。
- `-p`*`RDS_password`* – 用于指定密码。在第二次使用该参数时，您为第二个 `-u` 参数确定的用户账户指定密码。

确保在 Amazon RDS 数据库中手动创建任何存储过程、触发器、函数或事件。如果您所复制的数据库中有上述任一对象，则在运行 `mysqldump` 或 `mariadb-dump` 时会排除这些对象。为此，使用 `mysqldump` 或 `mariadb-dump` 命令时请包含以下参数：

- `--routines=0`
- `--triggers=0`
- `--events=0`

示例

以下示例将本地主机上的 `world` 示例数据库复制到 RDS for MySQL 数据库实例中。将值替换为您自己的信息。要将示例数据库复制到 RDS for MariaDB 数据库实例，请将 `mysqldump` 替换为 `mariadb-dump`，并将 `mysql` 替换为 `mariadb`。

对于 Linux、macOS 或 Unix：

```
sudo mysqldump -u local_user \
    --databases world \
    --single-transaction \
    --compress \
    --order-by-primary  \
    --routines=0 \
    --triggers=0 \
    --events=0 \
    -plocal_password | mysql -u rds_user \
        --port=3306 \
        --host=my_instance.123456789012.us-east-1.rds.amazonaws.com \
        -pRDS_password
```

对于 Windows：

在已通过右键单击 Windows 程序菜单上的命令提示符并选择以管理员身份运行打开的命令提示符中，运行以下命令：将值替换为您自己的信息。要将示例数据库复制到 RDS for MariaDB 数据库实例，请将 mysqldump 替换为 mariadb-dump，并将 mysql 替换为 mariadb。

```
mysqldump -u local_user ^
    --databases world ^
    --single-transaction ^
    --compress ^
    --order-by-primary  ^
    --routines=0 ^
    --triggers=0 ^
    --events=0 ^
    -plocal_password | mysql -u RDS_user ^
        --port=3306 ^
        --host=my_instance.123456789012.us-east-1.rds.amazonaws.com ^
        -pRDS_password
```

> **Note**
>
> 作为安全最佳实践，请指定示例中显示的提示以外的凭证。

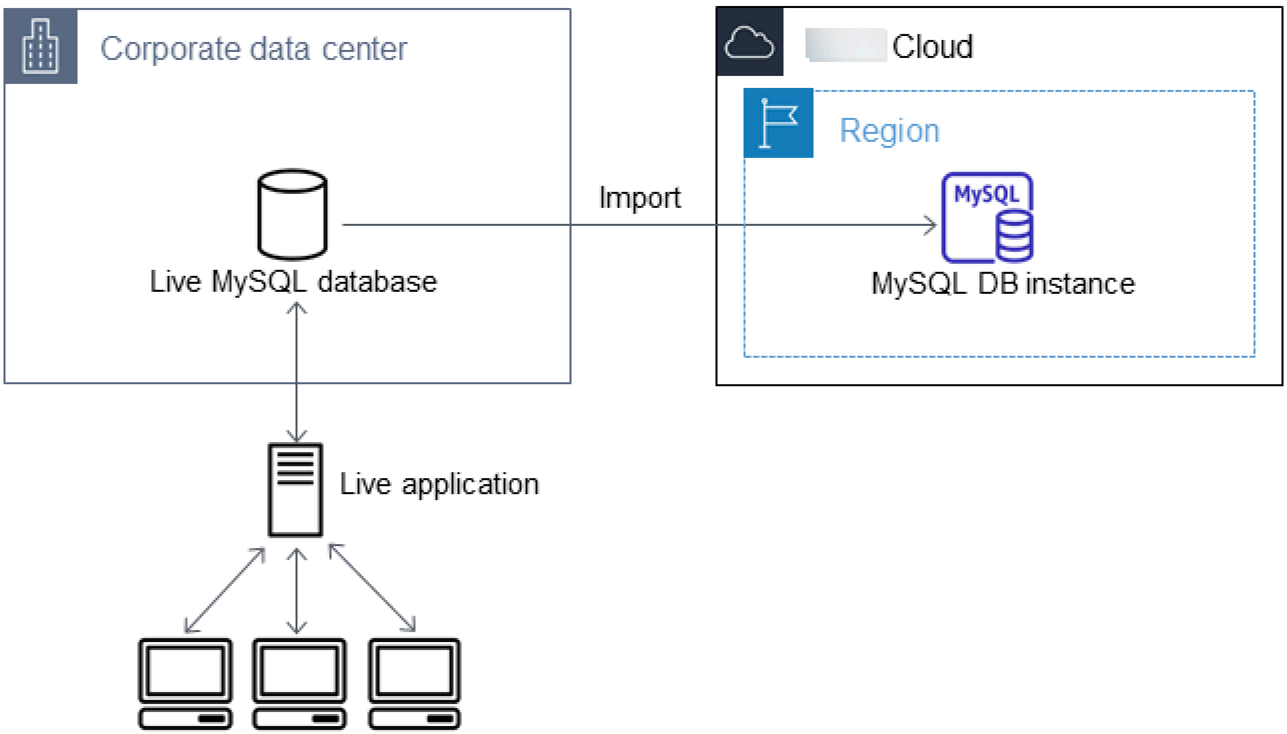
将数据导入 Amazon RDS for MySQL 数据库实例并减少停机时间

有时，您可能需要将数据从支持活动应用程序的外部 MariaDB 或 MySQL 数据库导入到 RDS for MariaDB 数据库实例、RDS for MySQL 数据库实例或 RDS for MySQL 多可用区数据库集群。使用以下过程可最大程度地减小对应用程序可用性产生的影响。如果使用非常大的数据库，该步骤也很有用。使用该过程，您可以通过减少跨网络传送到 Amazon 的数据量以降低导入成本。

在该过程中，您将数据库数据的副本传输到 Amazon EC2 实例，并将数据导入到新的 Amazon RDS 数据库中。然后，使用复制功能使 Amazon RDS 数据库与活动的外部实例保持同步，再将应用程序重新导向到 Amazon RDS 数据库。如果外部实例是 MariaDB 10.0.24 或更高版本，并且目标实例是 RDS for MariaDB，则基于全局事务标识符（GTID）配置 MariaDB 复制。否则，将基于二进制日志坐标配置复制。如果您的外部数据库支持基于 GTID 的复制，我们建议使用它，因为基于 GTID 复制是一种更可靠的方法。有关更多信息，请参阅 MariaDB 文档中的全局事务处理 ID。

> **Note**
>
> 如果您想要将数据导入到 RDS for MySQL 数据库实例中且您的方案支持此操作，则建议使用备份文件和 Amazon S3 将数据移入和移出 Amazon RDS。有关更多信息，请参阅 将备份还原到 Amazon RDS for MySQL 数据库实例。

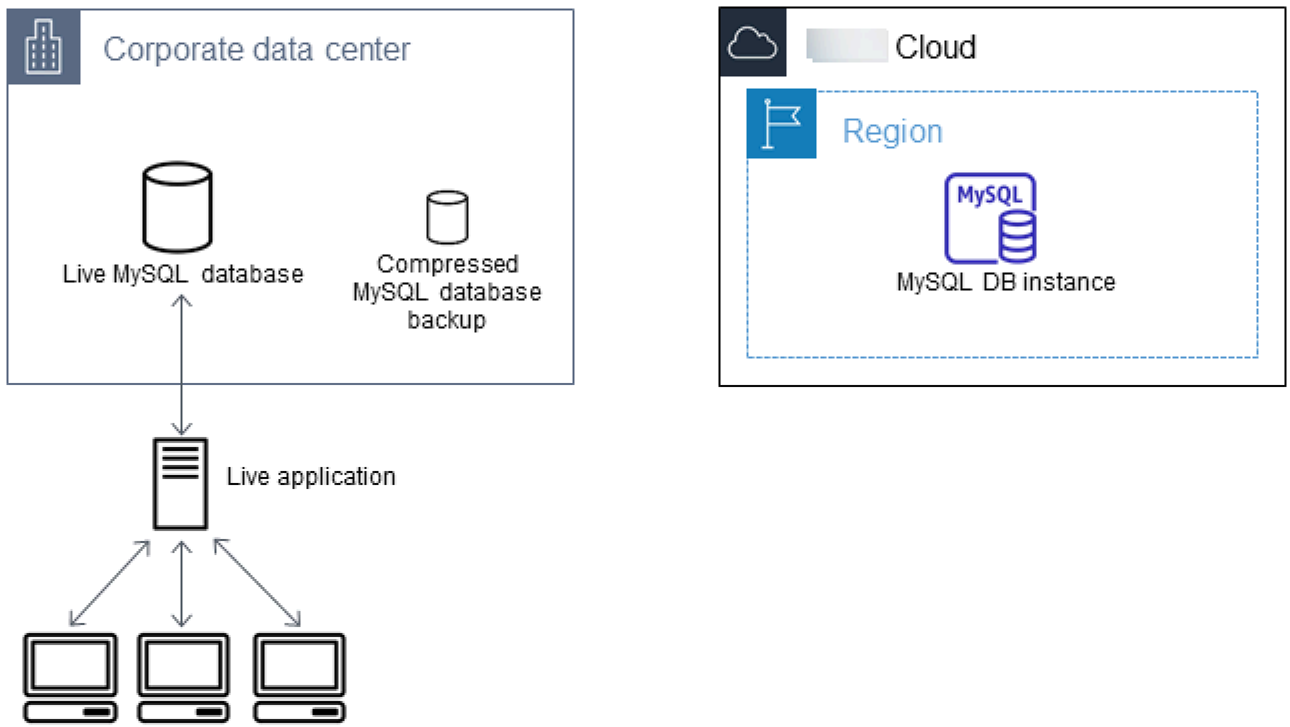
下图显示将外部 MySQL 数据库导入到 Amazon RDS 上的 MySQL 数据库。



任务 1：创建现有数据库的副本

要将大量数据迁移到停机时间最少的 RDS for MariaDB 或 RDS for MySQL 数据库，第一步是创建源数据的副本。

下图显示了创建 MySQL 数据库的备份。



您可以使用 mysqldump 实用程序创建 SQL 或分隔的文本格式的数据库备份。在 MariaDB 10.5 中，客户端被称为 mariadb-dump。从 MariaDB 11.0.1 开始，您必须使用 mariadb-dump 而不是 mysqldump。建议您在非生产环境中对每种格式运行测试，以便了解哪种方法能最大程度地减少 mysqldump 运行的时间。

我们还建议您对照将分隔文本格式用于加载所获得的好处来权衡 mysqldump 性能。使用分隔的文本格式的备份将为所转储的每个表创建制表符分隔的文本文件。为减少导入数据库所需的时间，您可以使用 LOAD DATA LOCAL INFILE 命令并行加载这些文件。有关更多信息，请参阅加载数据（适用于 MariaDB）或加载数据（适用于 MySQL）。

在开始备份操作之前，确保对将复制到 Amazon RDS 的 MariaDB 或 MySQL 数据库设置复制选项。复制选项包括开启二进制日志记录和设置唯一的服务器 ID。设置这些选项将促使您的服务器开始记录数据库事务，并做好在该过程的后面充当源复制实例的准备。

确保您了解以下建议和注意事项：

- 将 --single-transaction 选项与 mysqldump 结合使用，因为它会转储数据库的一致状态。为确保转储文件有效，请勿在 mysqldump 运行时运行数据定义语言 (DDL) 语句。您可以为这些操作计划维护时段。
- 从转储文件中排除以下架构：
  - sys

  - performance_schema
  - information_schema

  默认情况下，mysqldump 和 mariadb-dump 实用程序排除这些架构。

- 如果您需要迁移用户和权限，请考虑使用生成数据控制语言 (DCL) 的工具 (如 pt-show-grants 实用程序) 来重新创建用户和权限。

设置复制选项

1. 编辑 my.cnf 文件。此文件通常位于 /etc 下。

   ```
   sudo vi /etc/my.cnf
   ```

   将 log_bin 和 server_id 选项添加到 [mysqld] 节。log_bin 选项为二进制日志文件提供文件名标识符。server_id 选项为源-副本关系中的服务器提供唯一标识符。

   以下示例显示 my.cnf 文件已更新的 [mysqld] 部分：

   ```
   [mysqld]
   log-bin=mysql-bin
   server-id=1
   ```

   有关更多信息，请参阅 MySQL 文档中的 Setting the Replication Source Configuration。

2. 要使用多可用区数据库集群进行复制，请将 ENFORCE_GTID_CONSISTENCY 和 GTID_MODE 参数设置为 ON。

   ```
   mysql> SET @@GLOBAL.ENFORCE_GTID_CONSISTENCY = ON;
   ```

   ```
   mysql> SET @@GLOBAL.GTID_MODE = ON;
   ```

   使用数据库实例进行复制不需要这些设置。

3. 重新启动 mysql 服务。

   ```
   sudo service mysqld restart
   ```

创建现有数据库的备份副本

1. 使用 mysqldump 实用程序或 mariadb-dump 实用程序创建数据的备份，并指定 SQL 或分隔文本格式。

   对于 MySQL 8.0.25 及更低版本，指定 --master-data=2 以创建可用于启动服务器间复制的备份文件。对于 MySQL 8.0.26 及更高版本，指定 --source-data=2 以创建可用于启动服务器间复制的备份文件。有关更多信息，请参阅 MySQL 文档中的 mysqldump — A Database Backup Program。

   要提高性能和确保数据完整性，请对于 mysqldump 使用 --order-by-primary 和 --single-transaction 选项。

   要避免在备份中包括 MySQL 系统数据库，请不要将 --all-databases 选项与 mysqldump 结合使用。有关更多信息，请参阅 MySQL 文档中的 Creating a Data Snapshot Using mysqldump。

   如有必要，可使用 chmod 来确保要在其中创建备份文件的目录是可写的。

   Important

   在 Windows 中，以管理员身份运行命令窗口。

   - 要生成 SQL 输出，请使用以下命令：

     对于 Linux、macOS 或 Unix：

     ```
     sudo mysqldump \
         --databases database_name \
         --master-data=2  \
         --single-transaction \
         --order-by-primary \
         -r backup.sql \
         -u local_user \
         -ppassword
     ```

     Note

     作为安全最佳实践，请指定示例中显示的提示以外的凭证。

对于 Windows：

```
mysqldump ^
    --databases database_name ^
    --master-data=2  ^
    --single-transaction ^
    --order-by-primary ^
    -r backup.sql ^
    -u local_user ^
    -ppassword
```

> **Note**
>
> 作为安全最佳实践，请指定示例中显示的提示以外的凭证。

- 要生成分隔文本输出，请使用以下命令：

  对于 Linux、macOS 或 Unix：

```
sudo mysqldump \
    --tab=target_directory \
    --fields-terminated-by ',' \
    --fields-enclosed-by '"' \
    --lines-terminated-by 0x0d0a \
    database_name \
    --master-data=2 \
    --single-transaction \
    --order-by-primary \
    -ppassword
```

  对于 Windows：

```
mysqldump ^
    --tab=target_directory ^
    --fields-terminated-by "," ^
    --fields-enclosed-by """ ^
    --lines-terminated-by 0x0d0a ^
    database_name ^
    --master-data=2 ^
    --single-transaction ^
```

```
    --order-by-primary ^
    -ppassword
```

> **Note**
>
> 作为安全最佳实践，请指定示例中显示的提示以外的凭证。
>
> 确保在 Amazon RDS 数据库中手动创建任何存储过程、触发器、函数或事件。如果您所复制的数据库中有上述任一对象，则在运行 `mysqldump` 时排除这些对象。为此，请在 `mysqldump` 命令中包含以下参数：
>
> - `--routines=0`
> - `--triggers=0`
> - `--events=0`

对于 MySQL 8.0.22 及更低版本以及 MariaDB 版本，当您运行 `mysqldump` 并指定分隔文本格式时，将返回 `CHANGE MASTER TO` 注释。此注释包含主日志文件名称和位置。对于 MySQL 8.0.23 及更高版本，如果您使用分隔文本格式运行 `mysqldump`，将返回 `CHANGE REPLICATION SOURCE TO` 注释。此注释包含源日志文件名称和位置。如果外部实例不是 MariaDB 10.0.24 和更高版本，也不是 MySQL 8.0.23 和更高版本，请注意 `MASTER_LOG_FILE` 和 `MASTER_LOG_POS` 的值。设置复制时需要这些值。

对于 MariaDB 版本和 MySQL 8.0.22 及更低版本，将返回以下输出：

```
-- Position to start replication or point-in-time recovery from
--
-- CHANGE MASTER TO MASTER_LOG_FILE='mysql-bin-changelog.000031',
 MASTER_LOG_POS=107;
```

对于 MySQL 8.0.23 及更高版本，将返回以下输出：

```
-- Position to start replication or point-in-time recovery from
--
-- CHANGE SOURCE TO SOURCE_LOG_FILE='mysql-bin-changelog.000031',
 SOURCE_LOG_POS=107;
```

对于 MySQL 8.0.22 及更低版本，如果您使用的是 SQL 格式，则可以在备份文件的 `CHANGE MASTER TO` 注释中获得主日志文件名称和位置。对于 MySQL 8.0.23 及更高版本，如果您使用的

是 SQL 格式，则可以在备份文件的 CHANGE REPLICATION SOURCE TO 注释中获得源日志文件名称和位置。如果外部实例是 MariaDB 10.0.24 或更高版本，则可在下一个步骤中获得 GTID。

2. 如果您使用的外部实例是 MariaDB 10.0.24 或更高版本，请使用基于 GTID 的复制。在外部 MariaDB 实例上运行 SHOW MASTER STATUS 以获取二进制日志文件名称和位置，然后通过在外部 MariaDB 实例上运行 BINLOG_GTID_POS 将它们转换为 GTID。

```
SELECT BINLOG_GTID_POS('binary_log_file_name', binary_log_file_position);
```

请记下返回的 GTID。您需要此 GTID 来配置复制。

3. 压缩复制后的数据可减少将数据复制到 Amazon RDS 数据库所需的网络资源量。记下备份文件的大小。您在确定要创建的 Amazon EC2 实例的大小时需要此信息。完成操作后，请使用 GZIP 或您的首选压缩实用程序来压缩备份文件。

   - 要压缩 SQL 输出，请使用以下命令：

   ```
   gzip backup.sql
   ```

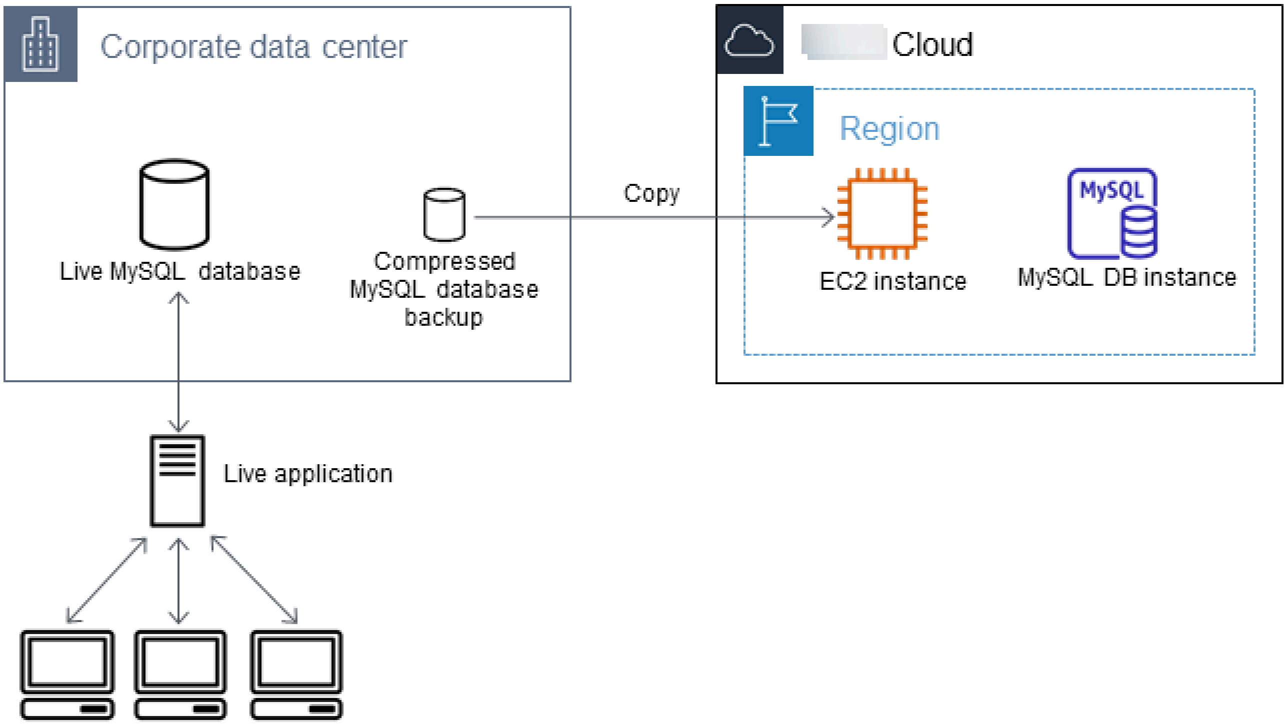
   - 要压缩分隔文本输出，请使用以下命令：

   ```
   tar -zcvf backup.tar.gz target_directory
   ```

任务 2：创建 Amazon EC2 实例和复制压缩的数据库

将压缩的数据库备份文件复制到 Amazon EC2 实例所占用的网络资源少于直接在数据库实例之间复制未压缩的数据所占用的网络资源。当您的数据位于 Amazon EC2 中后，可以直接从中将数据复制到 MariaDB 或 MySQL 数据库。要节省网络资源成本，您的 Amazon EC2 实例必须与 Amazon RDS 数据库实例位于同一 Amazon Web Services 区域内。使 Amazon EC2 实例位于 Amazon RDS 数据库所在的相同 Amazon Web Services 区域内，还可以减少导入期间的网络延迟。

下图显示了将数据库备份复制到 Amazon EC2 实例。



创建 Amazon EC2 实例并复制数据

1. 在您打算在其中创建 Amazon RDS 数据库的 Amazon Web Services 区域中，创建虚拟私有云（VPC）、VPC 安全组和 VPC 子网。确保您的 VPC 安全组的入站规则允许应用程序连接到 Amazon 所需的 IP 地址。您可以指定一系列 IP 地址（例如 `203.0.113.0/24`）或另一个 VPC 安全组。可以使用 Amazon VPC 控制台来创建和管理 VPC、子网和安全组。有关更多信息，请参阅《Amazon Virtual Private Cloud 用户指南》中的 Amazon VPC 入门。
2. 打开 Amazon EC2 控制台，并选择将包含 Amazon EC2 实例和 Amazon RDS 数据库的 Amazon Web Services 区域。使用您在步骤 1 中创建的 VPC、子网和安全组来启动 Amazon EC2 实例。在数据库备份文件未压缩的情况下，确保为该文件选择具有足够存储空间的实例类型。有关 Amazon EC2 实例的详细信息，请参阅《Amazon Elastic Compute Cloud 用户指南》中的开始使用 Amazon EC2。
3. 要从 Amazon EC2 实例连接到 Amazon RDS 数据库，请编辑 VPC 安全组。添加指定 EC2 实例的私有 IP 地址的入站规则。您可以在 EC2 控制台窗口中的 Instance 窗格的 Details 选项卡上找到私有 IP 地址。要编辑 VPC 安全组并添加入站规则，请在 EC2 控制台导航窗格中选择 Security Groups（安全组），然后为 MySQL 或 Aurora 添加指定 EC2 实例的私有 IP 地址的入站规则。要了解如何向 VPC 安全组添加入站规则，请参阅《Amazon Virtual Private Cloud 用户指南》中的安全组规则。

4. 将压缩后的数据库备份文件从本地系统复制到 Amazon EC2 实例中。如有必要，可使用 chmod 来确保您具有 Amazon EC2 实例目标目录的写入权限。您可以使用 scp 或 Secure Shell (SSH) 客户端复制文件。以下命令是一个示例 scp 命令：

```
scp -r -i key pair.pem backup.sql.gz ec2-user@EC2 DNS:/target_directory/backup.sql.gz
```

> **Important**
> 复制敏感数据时，务必使用安全网络传输协议。

5. 连接到 Amazon EC2 实例，并使用以下命令安装最新更新和 MySQL 客户端工具：

```
sudo yum update -y
sudo yum install mysql -y
```

有关更多信息，请参阅《Amazon Elastic Compute Cloud 用户指南》中的连接到您的实例（适用于 Linux 实例）。

> **Important**
> 此示例介绍了如何在适用于 Amazon Linux 发行版的 Amazon Machine Image (AMI) 上安装 MySQL 客户端。此示例不在其它发行版（例如 Ubuntu 或 Red Hat Enterprise Linux）上安装 MySQL 客户端。有关安装 MySQL 的信息，请参阅 MySQL 文档中的 Installing MySQL。

6. 在连接到 Amazon EC2 实例时，请解压缩数据库备份文件。下面是一些命令示例。

- 要解压缩 SQL 输出，请使用以下命令：

```
gzip backup.sql.gz -d
```

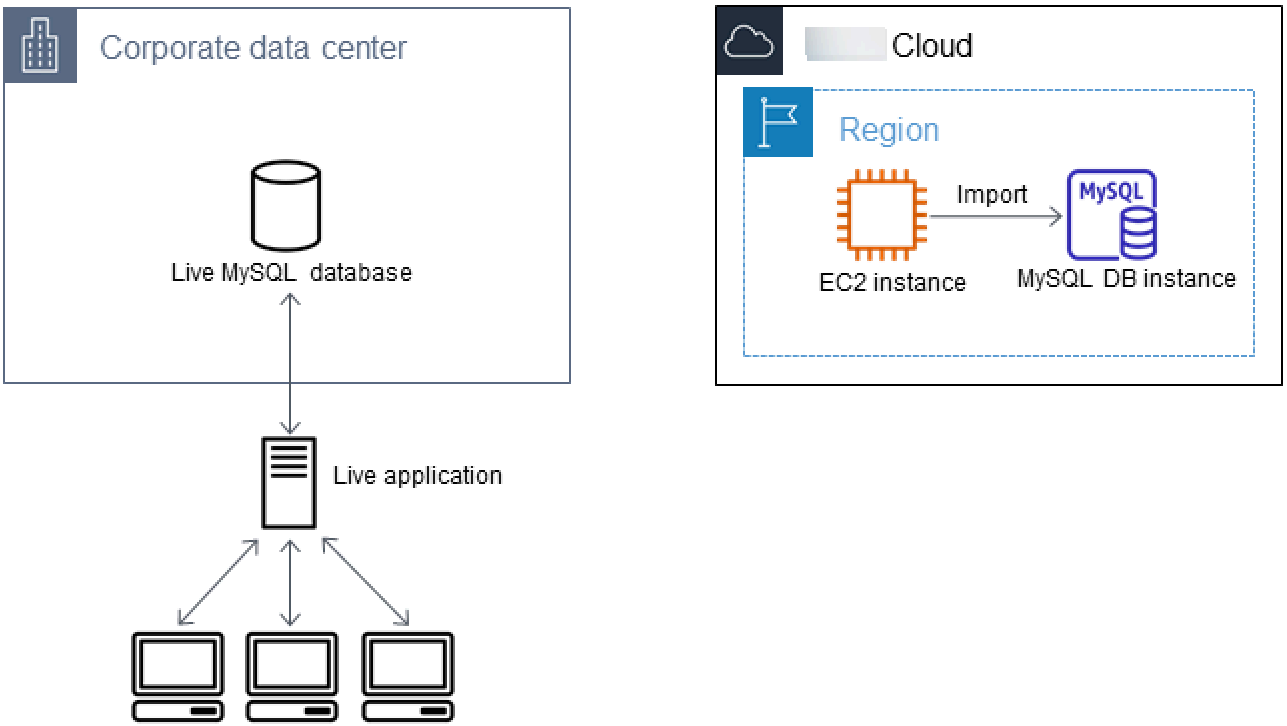
- 要解压缩分隔的文本输出，请使用以下命令：

```
tar xzvf backup.tar.gz
```

任务 3：创建 MySQL 或 MariaDB 数据库，并从 Amazon EC2 实例导入数据

通过在 Amazon EC2 实例所在的相同 Amazon Web Services 区域中创建 RDS for MariaDB 数据库实例、RDS for MySQL 数据库实例或 RDS for MySQL 多可用区数据库集群，与通过互联网相比，可以更快地从 Amazon EC2 导入数据库备份文件。

下图显示了将备份从 Amazon EC2 实例导入到 MySQL 数据库。



创建 MariaDB 或 MySQL 数据库并导入数据

1. 确定支持此 Amazon RDS 数据库的预期工作负载所需的数据库实例类和存储空间量。作为过程的一部分，确定数据加载过程所需的足够空间量和处理能力。此外，确定处理生产工作负载所需的项目。可以根据源 MariaDB 或 MySQL 数据库的大小和资源来进行估计。有关更多信息，请参阅 数据库实例类。
2. 在包含您的 Amazon EC2 实例的 Amazon Web Services 区域中创建数据库实例或多可用区数据库集群。

   要创建 RDS for MySQL 多可用区数据库集群，请按创建 Amazon RDS 的多可用区数据库集群中的说明操作。

   要创建 RDS for MariaDB 或 RDS for MySQL 数据库实例，请按照创建 Amazon RDS 数据库实例中的说明操作并使用以下准则：

- 指定与源数据库实例兼容的数据库引擎版本。
  - 如果源实例是 MySQL 5.5.x，则 Amazon RDS 数据库实例必须是 MySQL。
  - 如果源实例是 MySQL 5.6.x 或 5.7.x，则 Amazon RDS 数据库实例必须是 MySQL 或 MariaDB。
  - 如果源实例是 MySQL 8.0.x，则 Amazon RDS 数据库实例必须是 MySQL 8.0.x。
  - 如果源实例是 MySQL 8.4.x，则 Amazon RDS 数据库实例必须是 MySQL 8.4.x。
  - 如果源实例是 MariaDB 5.5 或更高版本，则 Amazon RDS 数据库实例必须是 MariaDB。
- 指定用于 Amazon EC2 实例的相同虚拟私有云（VPC）和 VPC 安全组。此方式将确保 Amazon EC2 实例和 Amazon RDS 实例通过网络相互可见。确保您的数据库实例可公开访问。要按下一节所述设置源数据库的复制，您的数据库实例必须可公开访问。
- 在导入数据库备份之前，不要配置多个可用区、备份保留或只读副本。在完成该导入操作后，您可以为生产实例配置多可用区和备份保留期。

3. 检查 Amazon RDS 数据库的默认配置选项。如果数据库的默认参数组没有所需的配置选项，请查找具有这些选项的参数组，或创建新的参数组。有关创建参数组的详细信息，请参阅 [Amazon RDS 的参数组]。
4. 以主用户身份连接到新 Amazon RDS 数据库。创建为需要访问数据库实例的管理员、应用程序和服务提供支持所需的用户。Amazon RDS 数据库的主机名为此数据库实例的端点值（不包含端口号），例如 `mysampledb.123456789012.us-west-2.rds.amazonaws.com`。您可以在 Amazon RDS 控制台上的数据库详细信息中找到端点值。
5. 连接到 Amazon EC2 实例。有关更多信息，请参阅《Amazon Elastic Compute Cloud 用户指南》中的[连接到您的实例]（适用于 Linux 实例）。
6. 使用 `mysql` 命令以远程主机身份从 Amazon EC2 实例连接到 Amazon RDS 数据库。以下命令是一个示例：

   ```
   mysql -h host_name -P 3306 -u db_master_user -p
   ```

   host_name 是 Amazon RDS 数据库端点。
7. 在 `mysql` 提示符下，运行 `source` 命令并向它传递您的数据库转储文件名。此命令将数据加载到 Amazon RDS 数据库实例。

   - 对于 SQL 格式，请使用以下命令：

     ```
     mysql> source backup.sql;
     ```

- 对于分隔文本格式，请先创建数据库（如果它不是您在设置 Amazon RDS 数据库时创建的默认数据库）。

  ```
  mysql> create database database_name;
  mysql> use database_name;
  ```

  然后，创建表。

  ```
  mysql> source table1.sql
  mysql> source table2.sql
  etc...
  ```

  然后，导入数据。

  ```
  mysql> LOAD DATA LOCAL INFILE 'table1.txt' INTO TABLE table1 FIELDS TERMINATED BY
   ',' ENCLOSED BY '"' LINES TERMINATED BY '0x0d0a';
  mysql> LOAD DATA LOCAL INFILE 'table2.txt' INTO TABLE table2 FIELDS TERMINATED BY
   ',' ENCLOSED BY '"' LINES TERMINATED BY '0x0d0a';
  etc...
  ```

  要提高性能，您可以从多个连接并行执行这些操作，以便同时创建并加载所有表。

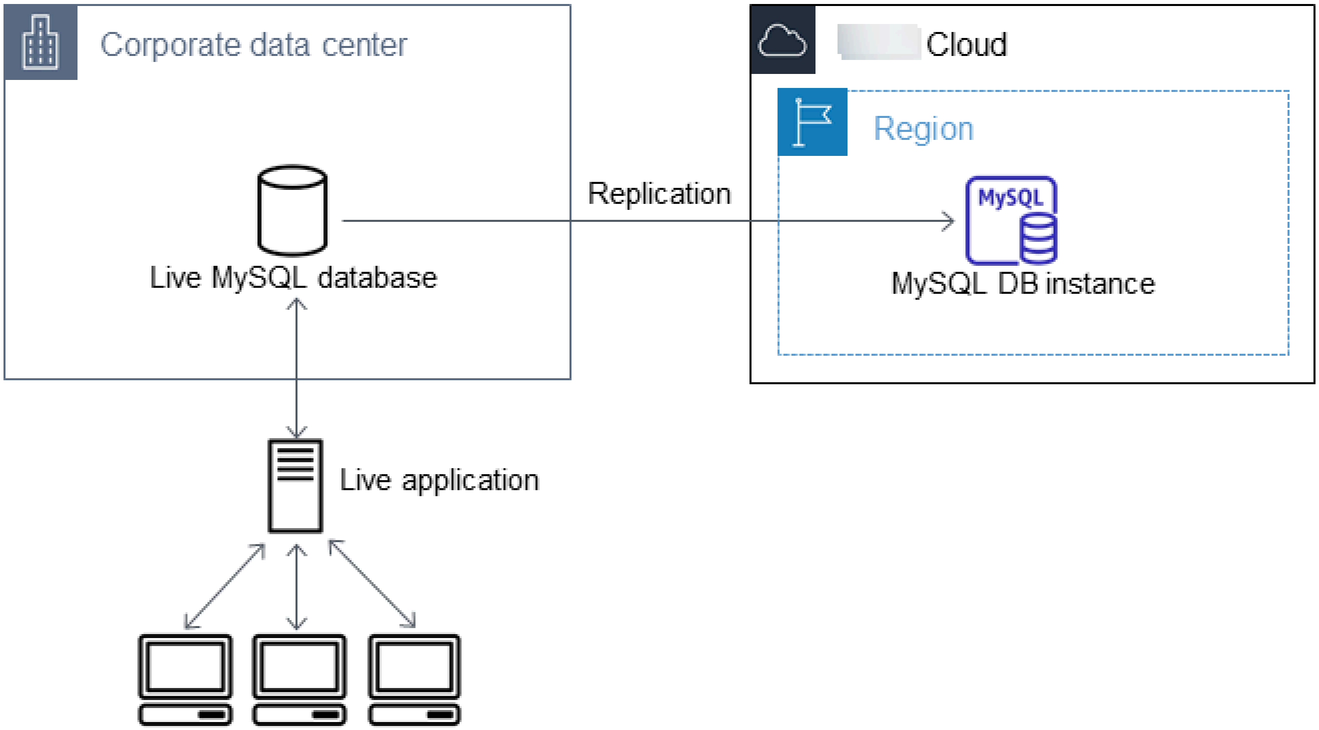
  > **Note**
  >
  > 如果您在最初转储表时已将任何数据格式化选项与 `mysqldump` 结合使用，则务必将相同的选项与 `LOAD DATA LOCAL INFILE` 结合使用，来确保正确地解读数据文件内容。

8. 对导入的数据库中的一个或两个表运行简单 `SELECT` 查询，来验证导入是否成功。

如果您不再需要该过程中使用的 Amazon EC2 实例，请终止 EC2 实例以减少Amazon资源使用。要终止 EC2 实例，请参阅《Amazon Elastic Compute Cloud 用户指南》中的终止实例。

任务 4：将数据从外部数据库复制到新的 Amazon RDS 数据库

在将数据复制并传输到 MariaDB 或 MySQL 数据库期间，您的源数据库可能已更新。这样，您可以使用复制功能使复制的数据库与源数据库保持同步。



对 Amazon RDS 数据库启动复制功能所需的权限受到限制且对 Amazon RDS 主用户不可用。因此，请使用适用于您的主要引擎版本的 Amazon RDS 存储过程：

- mysql.rds_set_external_master（RDS for MariaDB 和 RDS for MySQL 主要版本 8.0 及更低版本）
- mysql.rds_set_external_source（RDS for MySQL 主要版本 8.4 及更高版本）
- mysql.rds_set_external_master_gtid（配置复制）和 mysql.rds_start_replication（开始复制）

启动复制

在任务 1 中，当您设置复制选项时，您开启了二进制日志记录，并为源数据库设置了唯一服务器 ID。现在您可以将 Amazon RDS 数据库设置为副本，并将活动数据库作为源数据库实例。

1. 在 Amazon RDS 控制台中，将托管源数据库的服务器的 IP 地址添加到 Amazon RDS 数据库的 VPC 安全组。有关配置 VPC 安全组的更多信息，请参阅《Amazon Virtual Private Cloud 用户指南》中的配置安全组规则。

   您可能还需要配置本地网络，以允许来自 Amazon RDS 数据库的 IP 地址的连接，以便它能与源实例进行通信。要查找 Amazon RDS 数据库的 IP 地址，请使用 `host` 命令：

   ```
   host host_name
   ```

host_name 是 Amazon RDS 数据库端点中的 DNS 名称，例如 `myinstance.123456789012.us-east-1.rds.amazonaws.com`。您可以在 Amazon RDS 控制台上的数据库实例详细信息中找到端点值。

2. 通过使用所选的客户端，连接到源实例并创建将用于复制的用户。此账户仅用于复制，并且必须仅供您的域使用以增强安全性。以下命令是一个示例：

```
CREATE USER 'repl_user'@'mydomain.com' IDENTIFIED BY 'password';
```

> **Note**
>
> 作为安全最佳实践，请指定除此处所示提示以外的凭证。

3. 对于源 实例，向复制用户授予 `REPLICATION CLIENT` 和 `REPLICATION SLAVE` 权限。例如，要为您的域的“`repl_user`”用户授予对所有数据库的 `REPLICATION CLIENT` 和 `REPLICATION SLAVE` 权限，请发出以下命令。

```
GRANT REPLICATION CLIENT, REPLICATION SLAVE ON *.* TO 'repl_user'@'mydomain.com';
```

4. 如果您使用 SQL 格式创建备份文件，并且外部实例不是 MariaDB 10.0.24 或更高版本，请通过运行以下命令查看该文件的内容：

```
cat backup.sql
```

该文件包括 `CHANGE MASTER TO` 注释，其中包含主日志文件名称和位置。在将 `--master-data` 选项与 `mysqldump` 结合使用时，此注释将包含在备份文件中。请注意 `MASTER_LOG_FILE` 和 `MASTER_LOG_POS` 的值。

```
--
-- Position to start replication or point-in-time recovery from
--

-- CHANGE MASTER TO MASTER_LOG_FILE='mysql-bin-changelog.000031', MASTER_LOG_POS=107;
```

如果您使用分隔文本格式创建备份文件，并且外部实例不是 MariaDB 10.0.24 或更高版本，您应该已在任务 1 下“创建现有数据库的备份副本”过程的步骤 1 中获得二进制日志坐标。

如果外部实例是 MariaDB 10.0.24 或更高版本，您应该已在任务 1 下“创建现有数据库的备份副本”过程的步骤 2 中获得作为复制起点的 GTID。

5. 将 Amazon RDS 数据库设置为副本。如果外部实例不是 MariaDB 10.0.24 或更高版本，请使用适当的 Amazon RDS 存储过程，以主用户身份连接到 Amazon RDS 数据库，并将源数据库标识为源复制实例。

   • mysql.rds_set_external_master（RDS for MariaDB 和 RDS for MySQL 主要版本 8.0 及更低版本）
   • mysql.rds_set_external_source（RDS for MySQL 主要版本 8.4 及更高版本）

   如果您具有 SQL 格式的备份文件，则使用您在步骤 4 中确定的主日志文件名和主日志位置。如果您使用的是分隔文本格式，则使用您在创建备份文件时确定的名称和位置。下面是一些命令示例：

   MySQL 8.4 及更高版本

```
CALL mysql.rds_set_external_source ('myserver.mydomain.com', 3306,
    'repl_user', 'password', 'mysql-bin-changelog.000031', 107, 1);
```

   MariaDB 以及 MySQL 8.0 和更低版本

```
CALL mysql.rds_set_external_master ('myserver.mydomain.com', 3306,
    'repl_user', 'password', 'mysql-bin-changelog.000031', 107, 1);
```

   Note

   作为安全最佳实践，请指定除此处所示提示以外的凭证。

   如果外部实例是 MariaDB 10.0.24 或更高版本，请使用 mysql.rds_set_external_master_gtid 命令以主用户身份连接到 Amazon RDS 数据库，并将源数据库标识为源复制实例。使用您在任务 1 下“创建现有数据库的备份副本”过程的步骤 2 中确定的 GTID。以下命令是一个示例：

```
CALL mysql.rds_set_external_master_gtid ('source_server_ip_address', 3306,
 'ReplicationUser', 'password', 'GTID', 1);
```

   source_server_ip_address 是源复制实例的 IP 地址。当前不支持 EC2 私有 DNS 地址。

> **Note**
>
> 作为安全最佳实践，请指定除此处所示提示以外的凭证。

6. 在 Amazon RDS 数据库上，要开始复制，请运行以下 mysql.rds_start_replication 命令：

```
CALL mysql.rds_start_replication;
```

7. 在 Amazon RDS 数据库上，要确定副本何时与源复制实例实现同步，请运行 SHOW REPLICA STATUS 命令。`SHOW REPLICA STATUS` 命令的结果包括 `Seconds_Behind_Master` 字段。当 `Seconds_Behind_Master` 字段返回 0 时，副本将与源复制实例保持同步。

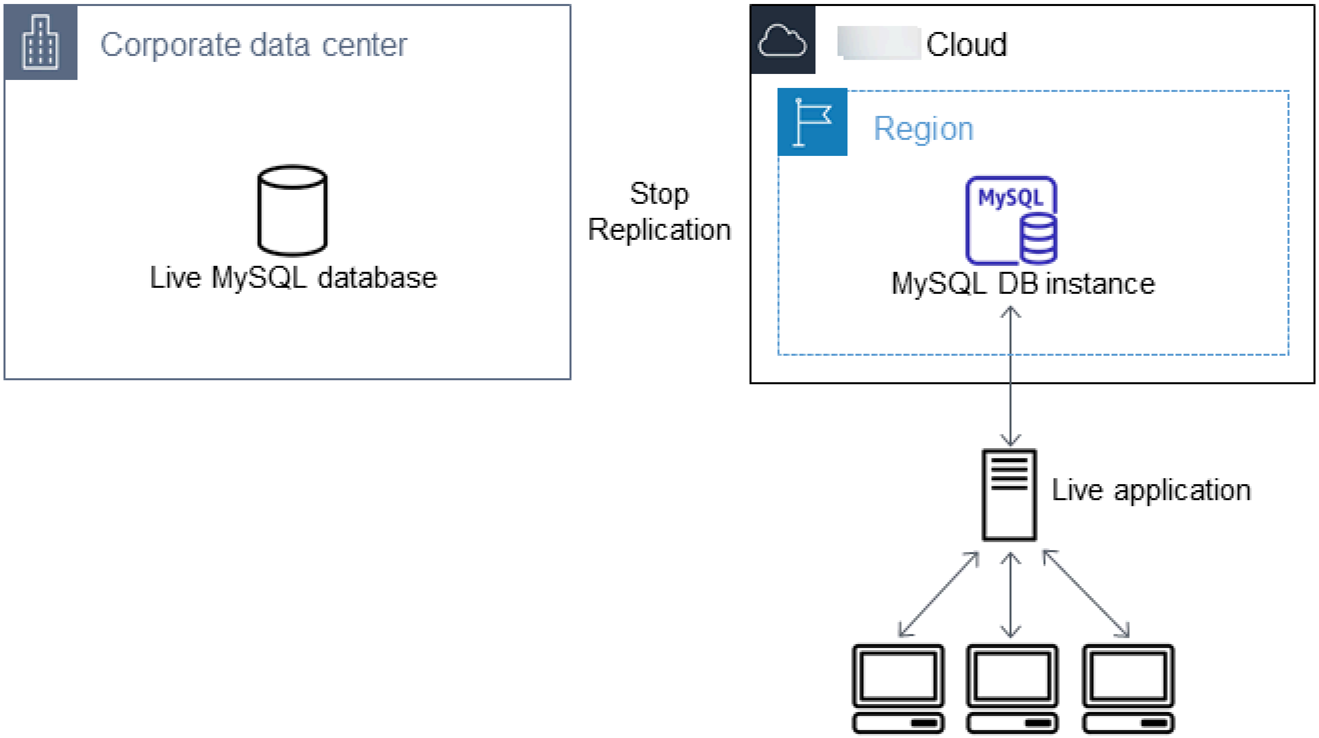
> **Note**
>
> 以前的 MySQL 版本使用的是 `SHOW SLAVE STATUS`，而不是 `SHOW REPLICA STATUS`。如果您使用的 MySQL 版本低于 8.0.23，那么请使用 `SHOW SLAVE STATUS`。

   对于 MariaDB 10.5、10.6 或 10.11 数据库实例，请运行 mysql.rds_replica_status 存储过程而不是 MySQL 命令。

8. 在 Amazon RDS 数据库同步后，开启自动备份以便您可以在需要时还原该数据库。您可以使用 Amazon RDS 控制台开启或修改 Amazon RDS 数据库的自动备份。有关更多信息，请参阅 备份简介。

任务 5：将您的活动应用程序重定向到 Amazon RDS 实例

在 MariaDB 或 MySQL 数据库与源复制实例保持同步后，您现在可以更新活动的应用程序以使用 Amazon RDS 实例。



将活动应用程序重新导向到 MariaDB 或 MySQL 数据库并停止复制

1. 要添加 Amazon RDS 数据库的 VPC 安全组，请添加托管应用程序的服务器的 IP 地址。有关修改 VPC 安全组的更多信息，请参阅《Amazon Virtual Private Cloud 用户指南》中的配置安全组规则。
2. 验证 SHOW REPLICA STATUS 命令结果中的 `Seconds_Behind_Master` 字段是否为 0，这表示副本与源复制实例保持同步。

```
SHOW REPLICA STATUS;
```

> **Note**
>
> 以前的 MySQL 版本使用的是 `SHOW SLAVE STATUS`，而不是 `SHOW REPLICA STATUS`。如果您使用的 MySQL 版本低于 8.0.23，那么请使用 `SHOW SLAVE STATUS`。

对于 MariaDB 10.5、10.6 或 10.11 数据库实例，请运行 mysql.rds_replica_status 过程而不是 MySQL 命令。

3. 在事务完成后关闭与源的所有连接。
4. 更新应用程序以使用 Amazon RDS 数据库。此更新通常涉及更改连接设置以标识 Amazon RDS 数据库的主机名和端口、用于进行连接的用户账户和密码以及要使用的数据库。

5. 连接到数据库实例。

   对于多可用区数据库集群，连接至写入器数据库实例。
6. 使用以下 mysql.rds_stop_replication 命令停止 Amazon RDS 实例的复制：

```
CALL mysql.rds_stop_replication;
```

7. 在 Amazon RDS 数据库上运行相应的 Amazon RDS 存储过程以重置复制配置，以便不再将该实例标识为副本。
   - mysql.rds_reset_external_master（RDS for MariaDB 和 RDS for MySQL 主要版本 8.0 及更低版本）
   - mysql.rds_reset_external_source（RDS for MySQL 主要版本 8.4 及更高版本）

   MySQL 8.4 及更高版本

```
CALL mysql.rds_reset_external_source;
```

   MariaDB 以及 MySQL 8.0 和更低版本

```
CALL mysql.rds_reset_external_master;
```

8. 开启其他 Amazon RDS 特征，例如多可用区支持和只读副本。有关更多信息，请参阅配置和管理 Amazon RDS 的多可用区部署和使用数据库实例只读副本。

从任意源将数据导入 Amazon RDS for MySQL 数据库实例

以下步骤将引导您完成将数据导入到 Amazon RDS 数据库实例的过程：

步骤 1：创建包含待加载数据的平面文件

使用通用格式存储待加载的数据，如逗号分隔值 (CSV)。每个表都必须拥有其自己的文件，您不能将多个表的数据合并到同一文件。每个文件的名称都必须与对应的表相同。可以根据个人喜好设置文件扩展名。例如，如果表名称为 `sales`，文件名可能为 `sales.csv` 或 `sales.txt`。

如果可能，请按照正在加载的表的主键对数据进行排序。这样做可以显著优化加载时间并尽可能减少磁盘存储要求。

该过程的速度和效率取决于能否将文件的大小保持在较小状态。如果在未压缩状态下单个文件的大小超过 1 GiB，则需将此文件分割为多个文件，然后分别加载各个文件。

在类 Unix 的系统（包括 Linux）中，请使用 split 命令。例如，以下命令会将 sales.csv 文件分割为多个小于 1 GiB 的文件，且只在换行符处进行分割 (-C 1024m)。新文件的名称包括升序数字后缀。以下命令生成名称如 sales.part_00 和 sales.part_01 的文件。

```
split -C 1024m -d sales.csv sales.part_
```

其他操作系统也具有类似的实用程序。

您可以将平面文件存储在任何地方。但是，当在步骤 5 中加载数据时，必须从文件存在的相同位置调用 mysql Shell，或者在运行 LOAD DATA LOCAL INFILE 时使用文件的绝对路径。

步骤 2：阻止任何应用程序访问目标数据库实例

开始大量加载前，阻止所有应用程序活动访问您打算加载到的目标数据库实例。建议您这样做，特别是当其他会话将要修改正在加载的表或引用的表时。这可以降低加载过程中发生约束冲突的风险并提高加载性能。它还可以将数据库实例还原到加载前的某一时间点，而不会丢失未参与加载的进程所做的更改。

当然，这可能并不可行或不现实。如果您无法在加载前阻止应用程序访问数据库实例，应采取步骤以确保数据的可用性和完整性。所需的具体步骤在很大程度上根据特定的使用案例和现场要求而有所不同。

步骤 3：创建数据库快照

如果您计划将数据加载到不包含数据的新数据库实例，可以跳过此步骤。否则，我们建议您在加载数据之前和之后创建目标 Amazon RDS 数据库实例的数据库快照。Amazon RDS 数据库快照是数据库实例的完整备份，可用于将数据库实例还原到已知状态。在启动数据库快照时，针对数据库实例的 I/O 操作将在备份数据库过程中暂停。

通过在即将开始加载前创建数据库快照，您可以在需要时将数据库还原到加载前的状态。通过使用在加载后立即创建的数据库快照，您可以在遇到问题时不必再次加载数据。还可以在加载后使用数据库快照将数据导入到新的数据库实例中。

以下示例运行 Amazon CLI create-db-snapshot 命令来创建 AcmeRDS 实例的数据库快照，并为该数据库快照指定标识符 "preload"。

对于 Linux、macOS 或 Unix：

```
aws rds create-db-snapshot \
    --db-instance-identifier AcmeRDS \
```

```
    --db-snapshot-identifier preload
```

对于 Windows：

```
aws rds create-db-snapshot ^
    --db-instance-identifier AcmeRDS ^
    --db-snapshot-identifier preload
```

也可以使用从数据库快照还原功能，以创建用于试运行的测试数据库实例或撤消在加载期间进行的更改。

请务必注意，从数据库快照还原数据库将创建新的数据库实例，与所有数据库实例一样，该数据库实例具有唯一的标识符和端点。要还原数据库实例但不改变端点，请先删除该数据库实例，以使您能够重新使用此端点。

例如，要创建一个适用于试运行或其他测试的数据库实例，您可为数据库实例指定它自己的标识符。在示例中，标识符为 AcmeRDS-2"。该示例使用与 AcmeRDS-2 关联的端点连接到数据库实例。有关更多信息，请参阅 restore-db-instance-from-db-snapshot。

对于 Linux、macOS 或 Unix：

```
aws rds restore-db-instance-from-db-snapshot \
    --db-instance-identifier AcmeRDS-2 \
    --db-snapshot-identifier preload
```

对于 Windows：

```
aws rds restore-db-instance-from-db-snapshot ^
    --db-instance-identifier AcmeRDS-2 ^
    --db-snapshot-identifier preload
```

要重新使用现有的端点，请首先删除该数据库实例，然后为还原的数据库指定相同的标识符。有关更多信息，请参阅 delete-db-instance。

以下示例还在删除数据库实例之前为其制作最终的数据库快照。您可以自由选择，但建议您这样做。

对于 Linux、macOS 或 Unix：

```
aws rds delete-db-instance \
```

```
    --db-instance-identifier AcmeRDS \
    --final-db-snapshot-identifier AcmeRDS-Final

aws rds restore-db-instance-from-db-snapshot \
    --db-instance-identifier AcmeRDS \
    --db-snapshot-identifier preload
```

对于 Windows：

```
aws rds delete-db-instance ^
    --db-instance-identifier AcmeRDS ^
    --final-db-snapshot-identifier AcmeRDS-Final

aws rds restore-db-instance-from-db-snapshot ^
    --db-instance-identifier AcmeRDS ^
    --db-snapshot-identifier preload
```

步骤 4（可选）：关闭 Amazon RDS 自动备份

> **Warning**
>
> 如果需要执行时间点故障恢复，请勿关闭自动备份。

关闭自动备份是一种性能优化，加载数据时不需要这样做。关闭自动备份会清除所有现有备份。因此，在关闭自动备份后，将无法使用时间点故障恢复功能。关闭自动备份不会影响手动数据库快照。仍然可以还原所有现有的手动数据库快照。

关闭自动备份可减少约 25% 的加载时间，并减少加载期间所需的存储空间。如果您计划将数据加载到不包含任何数据的新数据库实例，关闭备份就可轻松加速加载并避免使用备份所需的额外存储空间。但是，在某些情况下，您可能会计划将数据加载到已包含数据的数据库实例。如果是这样，请权衡关闭备份的优势与无法执行时间点恢复功能的影响。

数据库实例默认开启自动备份 (保留期为一天)。要关闭自动备份，请将备份保留期设置为零。加载后，可以将备份保留期设置为非零值，重新开启自动备份。要开启或关闭备份，Amazon RDS 将关闭并重新启动数据库实例，以开启或关闭 MariaDB 或 MySQL 日志记录。

运行 Amazon CLI `modify-db-instance` 命令，将备份保留期设置为零并立即应用更改。要将保留期设置为零，需要重新启动数据库实例，因此，等到重新启动完成后再继续操作。有关更多信息，请参阅 modify-db-instance。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier AcmeRDS \
    --apply-immediately \
    --backup-retention-period 0
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier AcmeRDS ^
    --apply-immediately ^
    --backup-retention-period 0
```

您可以使用 Amazon CLI describe-db-instances 命令检查数据库实例的状态。以下示例显示 AcmeRDS 数据库实例的数据库实例状态：

```
aws rds describe-db-instances --db-instance-identifier AcmeRDS --query "*[].
{DBInstanceStatus:DBInstanceStatus}"
```

当数据库实例状态为 available 时，您已准备好进行下一步。

步骤 5：加载数据

要将平面文件中的行读取到数据库表中，请使用 MySQL LOAD DATA LOCAL INFILE 语句。

> **Note**
>
> 必须从平面文件存在的相同位置调用 mysql Shell，或者在运行 LOAD DATA LOCAL INFILE 时使用文件的绝对路径。

以下示例说明如何将数据从名为 sales.txt 的文件加载到数据库中名为 Sales 的表中：

```
mysql> LOAD DATA LOCAL INFILE 'sales.txt' INTO TABLE Sales FIELDS TERMINATED BY ' '
 ENCLOSED BY '' ESCAPED BY '\\';
Query OK, 1 row affected (0.01 sec)
Records: 1  Deleted: 0  Skipped: 0  Warnings: 0
```

有关 LOAD DATA 语句的更多信息，请参阅 MySQL 文档中的 LOAD DATA Statement。

步骤 6：重新开启 Amazon RDS 自动备份

如果您在步骤 4 中关闭了 Amazon RDS 自动备份，则在加载完成后，通过将备份保留期重新设置为其预加载值来开启自动备份。如步骤 4 中所述，Amazon RDS 会重新启动数据库实例，因此，请为短暂中断做好准备。

以下示例运行 Amazon CLI [modify-db-instance](https://docs.aws.amazon.com/cli/latest/reference/rds/modify-db-instance.html) 命令来开启 `AcmeRDS` 数据库实例的自动备份，并将保留期设置为一天：

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier AcmeRDS \
    --backup-retention-period 1 \
    --apply-immediately
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier AcmeRDS ^
    --backup-retention-period 1 ^
    --apply-immediately
```

在 Amazon RDS 中使用 MySQL 复制

一般使用只读副本配置 Amazon RDS 数据库实例之间的复制。有关只读副本的一般信息，请参阅使用数据库实例只读副本。有关使用 Amazon RDS for MySQL 上的只读副本的特定信息，请参阅 使用 MySQL 只读副本。

您可以使用全局事务标识符 (GTID) 与 RDS for MySQL 之间进行复制。有关更多信息，请参阅 使用基于 GTID 的复制。

还可以设置 RDS for MySQL 数据库实例与 Amazon RDS 外部的 MariaDB 或 MySQL 实例之间的复制。有关配置外部源复制的信息，请参阅配置与外部源实例之间的二进制日志文件位置复制。

对于其中任何复制选项，您可以使用基于行的复制、基于语句的复制或混合复制。基于行的复制仅复制因 SQL 语句而导致更改的行。基于语句的复制将复制整个 SQL 语句。混合复制尽可能使用基于语句的复制，但当运行对于基于语句的复制不安全的 SQL 语句时切换到基于行的复制。在大多数情况下，建议使用混合复制。数据库实例的二进制日志格式确定复制是基于行、基于语句还是混合的。有关设置二进制日志格式的信息，请参阅为单可用区数据库配置 RDS for MySQL 二进制日志记录。

> **Note**
>
> 您可以配置复制以导入来自 Amazon RDS 外部的 MariaDB 或 MySQL 实例的数据库，或将数据库导出至此类实例。有关更多信息，请参阅将数据导入 Amazon RDS for MySQL 数据库实例并减少停机时间 和使用复制从 MySQL 数据库实例中导出数据。

从快照还原数据库实例或执行时间点恢复后，您可以在 RDS 控制台中查看上次从源数据库恢复的二进制日志位置。在日志和事件下，输入 binlog。二进制日志位置显示在系统注释下。

主题

使用 MySQL 只读副本

接下来，您可以找到有关使用 RDS for MySQL 上的只读副本的特定信息。有关只读副本及其使用说明的一般信息，请参阅使用数据库实例只读副本。

有关 MySQL 只读副本的更多信息，请参阅以下主题。

- 使用 MySQL 配置复制筛选条件
- 使用 MySQL 配置延迟复制
- 使用 MySQL 更新只读副本
- 使用 MySQL 处理多可用区只读副本部署
- 将级联只读副本用于 RDS for MySQL
- 监控 MySQL 只读副本的复制滞后
- 开始和停止 MySQL 只读副本复制
- 排查 MySQL 只读副本问题

使用 MySQL 配置只读副本

请确保在源数据库实例上启用自动备份，然后 MySQL DB 实例才能用作复制源。为此，请将备份保留期设定为非 0 值。此要求也适用于作为一个只读副本的源数据库实例的另一个只读副本。运行任何版本的 MySQL 的只读副本都支持自动备份。可以为 MySQL 数据库实例配置基于二进制日志坐标的复制。

可以在以下版本上使用全局事务标识符（GTIDS）配置复制：

- RDS for MySQL 5.7.44 版本及更高的 5.7 版本
- RDS for MySQL 8.0.28 版本及更高的 8.0 版本
- RDS for MySQL 8.4.3 版本及更高的 8.4 版本

有关更多信息，请参阅 使用基于 GTID 的复制。

您可以从同一区域内的一个数据库实例创建最多 15 个只读副本。为了有效地进行复制，每个只读副本具有的计算和存储资源的量应与源数据库实例的一样多。如果扩展源数据库实例，则还应扩展只读副本。

RDS for MySQL 支持级联只读副本。要了解如何配置级联只读副本，请参阅 将级联只读副本用于 RDS for MySQL。

您可以在引用相同源数据库实例的同时，运行多个只读副本创建和删除操作。当您执行这些操作时，应将每个源实例的只读副本数限制在 15 个以内。

MySQL 数据库实例的只读副本不能使用比其源数据库实例低的数据库引擎版本。

准备使用 MyISAM 的 MySQL 数据库实例

如果 MySQL 数据库实例使用的是 MyISAM 等非事务性引擎，则将需要执行以下步骤才能成功地设置只读副本。需要执行这些步骤，以确保只读副本具有一致的数据副本。如果您的所有表使用的都是事务性引擎 (如 InnoDB)，则无需执行这些步骤。

1. 停止源数据库实例中的非事务性表上的所有数据操作语言 (DML) 和数据定义语言 (DDL) 操作，然后等待它们完成。SELECT 语句可以继续运行。
2. 刷新并锁定源数据库实例中的表。
3. 使用以下各节中的某种方法创建只读副本。
4. 使用如 `DescribeDBInstances` API 操作检查只读副本创建的进度。有只读副本可用后，即解除对源数据库实例的表的锁定，然后继续进行正常的数据库操作。

使用 MySQL 配置复制筛选条件

您可以使用复制筛选条件来指定使用只读副本的数据库和表。复制筛选条件可以将数据库和表包含在复制之中或排除在复制之外。

以下是复制筛选条件的一些使用案例：

- 缩减只读副本的大小。使用复制筛选，您可以排除只读副本上不需要的数据库和表。
- 出于安全原因，将数据库和表从只读副本中排除。
- 在不同只读副本中为特定使用案例复制不同的数据库和表。例如，您可以使用特定的只读副本进行分析或分片。
- 对于在不同 Amazon Web Services 区域中具有只读副本的数据库实例，要在不同的 Amazon Web Services 区域中复制不同的数据库或表。

> **Note**
>
> 您还可以使用复制筛选条件来指定使用入站复制拓扑中裴志伟副本的主 MySQL 数据库实例的数据库和表。有关此配置的更多信息，请参阅配置与外部源实例之间的二进制日志文件位置复制。

主题

- 设置 RDS for MySQL 的复制筛选参数

- RDS for MySQL 的复制筛选限制
- RDS for MySQL 的复制筛选示例
- 查看只读副本的复制筛选条件

设置 RDS for MySQL 的复制筛选参数

要配置复制筛选条件，请在只读副本上设置以下复制筛选参数：

- `replicate-do-db` –将更改复制到指定的数据库。为只读副本设置此参数时，仅复制参数中指定的数据库。
- `replicate-ignore-db` –不将更改复制到指定的数据库。为只读副本设置 `replicate-do-db` 参数时，不会评估此参数。
- `replicate-do-table` –将更改复制到指定的表。为只读副本设置此参数时，仅复制参数中指定的表。此外，设置 `replicate-do-db` 或 `replicate-ignore-db` 参数时，请确保包含指定表的数据库包含在使用只读副本的复制中。
- `replicate-ignore-table` –不将更改复制到指定的表。为只读副本设置 `replicate-do-table` 参数时，不会评估此参数。
- `replicate-wild-do-table` – 根据指定的数据库和表名模式复制表。支持 `%` 和 `_` 通配符。设置 `replicate-do-db` 或 `replicate-ignore-db` 参数时，请确保包含指定表的数据库包含在使用只读副本的复制中。
- `replicate-wild-ignore-table` –不基于指定的数据库和表名模式复制表。支持 `%` 和 `_` 通配符。为只读副本设置 `replicate-do-table` 或 `replicate-wild-do-table` 参数时，不会评估此参数。

将按这些参数列出的顺序对其进行评估。有关这些参数如何运行的更多信息，请参阅 MySQL 文档：

- 有关一般信息，请参阅副本服务器选项和变量。
- 有关如何评估数据库复制筛选参数的信息，请参阅评估数据库级复制和二进制日志记录选项。
- 有关如何评估表复制筛选参数的信息，请参阅评估表级复制选项。

默认情况下，这些参数中的每个参数都具有一个空值。在每个只读副本上，您可以使用这些参数来设置、更改和删除复制筛选条件。设置其中一个参数时，请用逗号将各筛选条件分开。

您可以在 `%` 和 `_` 参数中使用 `replicate-wild-do-table` 和 `replicate-wild-ignore-table` 通配符。`%` 通配符可以匹配任意数量的字符，而 `_` 通配符只能匹配一个字符。

源数据库实例的二进制日志记录格式对于复制非常重要，因为它决定了数据更改的记录。`binlog_format` 参数的设置将决定复制是基于行还是基于语句的复制。有关更多信息，请参阅"为单可用区数据库配置 RDS for MySQL 二进制日志记录"。

> **Note**
>
> 无论源数据库实例上的 `binlog_format` 设置如何，所有数据定义语言 (DDL) 语句都将作为语句进行复制。

RDS for MySQL 的复制筛选限制

以下限制适用于对 RDS for MySQL 进行复制筛选：

- 每个复制筛选参数不得超过 2000 个字符。
- 参数值的复制筛选条件中不支持逗号。在参数列表中，逗号只能用作值分隔符。例如，不支持 `ParameterValue='`a,b`'`，但支持 `ParameterValue='a,b'`。
- 不支持用于二进制日志筛选的 MySQL `--binlog-do-db` 和 `--binlog-ignore-db` 选项。
- 复制筛选不支持 XA 事务。

  有关更多信息，请参阅 MySQL 文档中的XA 事务限制。

RDS for MySQL 的复制筛选示例

要为只读副本配置复制筛选，请修改与只读副本关联的参数组中的复制筛选参数。

> **Note**
>
> 您无法修改默认参数组。如果只读副本使用默认参数组，请创建新的参数组并将其与只读副本关联。有关数据库参数组的更多信息，请参阅 Amazon RDS 的参数组。

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 在参数组中设置参数。有关设置参数的信息，请参阅 在 Amazon RDS 中修改数据库参数组中的参数。在参数组中设置参数时，与参数组关联的所有数据库实例都使用参数设置。如果在参数组中设置复制筛选参数，请确保参数组仅与只读副本关联。将源数据库实例的复制筛选参数留空。

以下示例使用 Amazon CLI 设置参数。这些示例将 ApplyMethod 设置为 immediate，以便在 CLI 命令完成后立即发生参数更改。如果希望在只读副本重新启动后应用待处理的更改，请将 ApplyMethod 设置为 pending-reboot。

以下示例设置了复制筛选条件：

- Including databases in replication
- Including tables in replication
- Including tables in replication with wildcard characters
- Excluding databases from replication
- Excluding tables from replication
- Excluding tables from replication using wildcard characters

Example 将数据库包含在复制之中

以下示例将 mydb1 和 mydb2 数据库包含在复制之内。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-parameter-group \
  --db-parameter-group-name myparametergroup \
  --parameters "ParameterName=replicate-do-
db,ParameterValue='mydb1,mydb2',ApplyMethod=immediate"
```

对于 Windows：

```
aws rds modify-db-parameter-group ^
  --db-parameter-group-name myparametergroup ^
  --parameters "ParameterName=replicate-do-
db,ParameterValue='mydb1,mydb2',ApplyMethod=immediate"
```

Example 将表包含在复制之中

以下示例将数据库 table1 中的 table2 和 mydb1 表包含在复制之中。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-parameter-group \
  --db-parameter-group-name myparametergroup \
  --parameters "ParameterName=replicate-do-
table,ParameterValue='mydb1.table1,mydb1.table2',ApplyMethod=immediate"
```

对于 Windows：

```
aws rds modify-db-parameter-group ^
  --db-parameter-group-name myparametergroup ^
  --parameters "ParameterName=replicate-do-
table,ParameterValue='mydb1.table1,mydb1.table2',ApplyMethod=immediate"
```

Example 使用通配符将表包含在复制之中

以下示例将数据库 `order` 中名称以 `return` 和 mydb 开头的表包含在复制之中。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-parameter-group \
  --db-parameter-group-name myparametergroup \
  --parameters "ParameterName=replicate-wild-do-table,ParameterValue='mydb.order
%,mydb.return%',ApplyMethod=immediate"
```

对于 Windows：

```
aws rds modify-db-parameter-group ^
  --db-parameter-group-name myparametergroup ^
  --parameters "ParameterName=replicate-wild-do-table,ParameterValue='mydb.order
%,mydb.return%',ApplyMethod=immediate"
```

Example 将数据库排除在复制之外

以下示例将 mydb5 和 mydb6 数据库排除在复制之外。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-parameter-group \
  --db-parameter-group-name myparametergroup \
```

```
  --parameters "ParameterName=replicate-ignore-
db,ParameterValue='mydb5,mydb6',ApplyMethod=immediate"
```

对于 Windows：

```
aws rds modify-db-parameter-group ^
  --db-parameter-group-name myparametergroup ^
  --parameters "ParameterName=replicate-ignore-
db,ParameterValue='mydb5,mydb6',ApplyMethod=immediate"
```

Example 将表排除在复制之外

以下示例将数据库 `mydb5` 中的表 `table1` 和数据库 `mydb6` 中的表 `table2` 排除在复制之外。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-parameter-group \
  --db-parameter-group-name myparametergroup \
  --parameters "ParameterName=replicate-ignore-
table,ParameterValue='mydb5.table1,mydb6.table2',ApplyMethod=immediate"
```

对于 Windows：

```
aws rds modify-db-parameter-group ^
  --db-parameter-group-name myparametergroup ^
  --parameters "ParameterName=replicate-ignore-
table,ParameterValue='mydb5.table1,mydb6.table2',ApplyMethod=immediate"
```

Example 使用通配符将表排除在复制之外

以下示例将数据库 `order` 中名称以 `return` 和 `mydb7` 开头的表排除在复制之外。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-parameter-group \
  --db-parameter-group-name myparametergroup \
  --parameters "ParameterName=replicate-wild-ignore-table,ParameterValue='mydb7.order
%,mydb7.return%',ApplyMethod=immediate"
```

对于 Windows：

```
aws rds modify-db-parameter-group ^
  --db-parameter-group-name myparametergroup ^
  --parameters "ParameterName=replicate-wild-ignore-table,ParameterValue='mydb7.order
%,mydb7.return%',ApplyMethod=immediate"
```

查看只读副本的复制筛选条件

您可以通过以下方式查看只读副本的复制筛选条件：

- 检查与只读副本关联的参数组中复制筛选参数的设置。

  有关说明，请参阅在 Amazon RDS 中查看数据库参数组的参数值。
- 在 MySQL 客户端中，连接到只读副本并运行 `SHOW REPLICA STATUS` 语句。

  在输出中，以下字段显示了只读副本的复制筛选条件：
  - `Replicate_Do_DB`
  - `Replicate_Ignore_DB`
  - `Replicate_Do_Table`
  - `Replicate_Ignore_Table`
  - `Replicate_Wild_Do_Table`
  - `Replicate_Wild_Ignore_Table`

  有关这些字段的更多信息，请参阅 MySQL 文档中的检查复制状态。

使用 MySQL 配置延迟复制

可以使用延迟复制作为灾难恢复策略。使用延迟复制，可指定最短的时间（以秒为单位）以延迟从源到只读副本的复制。如果出现灾难（如意外删除了表），可完成以下步骤以快速从灾难恢复：

- 在导致灾难的更改发送到它之前停止复制到只读副本。

  使用 mysql.rds_stop_replication 存储过程停止复制。
- 开始复制并指定复制在日志文件位置处自动停止。

  使用 mysql.rds_start_replication_until 存储过程指定就位于灾难恢复之前的位置。
- 按照将只读副本提升为独立的数据库实例中的说明，将只读副本提升为新的源数据库实例。

> **Note**
>
> - 在 RDS for MySQL 8.4 上，支持针对 MySQL 8.4.3 及更高版本进行延迟复制。在 RDS for MySQL 8.0 上，支持针对 MySQL 8.0.28 及更高版本进行延迟复制。在 RDS for MySQL 5.7 上，支持针对 MySQL 5.7.44 及更高版本进行延迟复制。
> - 使用存储过程配置延迟复制。无法使用Amazon Web Services Management Console、Amazon CLI 或 Amazon RDS API 配置延迟复制。
> - 您可以在以下版本的延迟复制配置中使用基于全局事务标识符（GTID）的复制。
>   - RDS for MySQL 5.7.44 版本及更高的 5.7 版本
>   - RDS for MySQL 8.0.28 版本及更高的 8.0 版本
>   - RDS for MySQL 8.4.3 版本及更高的 8.4 版本
>
>   如果使用基于 GTID 的复制，请使用 mysql.rds_start_replication_until_gtid 存储过程而不是 mysql.rds_start_replication_until 存储过程。有关基于 GTID 的复制的更多信息，请参阅使用基于 GTID 的复制。

主题

- 在创建只读副本时配置延迟复制
- 修改现有只读副本的延迟复制
- 设置停止复制到只读副本的位置
- 提升只读副本

在创建只读副本时配置延迟复制

要为未来通过数据库实例创建的任何只读副本配置延迟复制，请使用 mysql.rds_set_configuration 参数运行 `target delay` 存储过程。

在创建只读副本时配置延迟复制

1. 通过使用 MySQL 客户端，以主用户身份连接到作为只读副本源的 MySQL 数据库实例。
2. 使用 mysql.rds_set_configuration 参数运行 `target delay` 存储过程。

   例如，运行以下存储过程，针对从当前数据库实例创建的任何只读副本，指定将复制延迟至少一小时（3600 秒）。

```
call mysql.rds_set_configuration('target delay', 3600);
```

> **Note**
>
> 运行此存储过程之后，将为使用 Amazon CLI 或 Amazon RDS API 创建的任何只读副本配置延迟指定秒数的复制。

修改现有只读副本的延迟复制

要修改现有只读副本的延迟复制，请运行 mysql.rds_set_source_delay 存储过程。

修改现有只读副本的延迟复制

1. 使用 MySQL 客户端，以主用户的身份连接到只读副本。
2. 使用 mysql.rds_stop_replication 存储过程停止复制。
3. 运行 mysql.rds_set_source_delay 存储过程。

   例如，运行以下存储过程，指定复制到只读副本延迟至少一小时（3600 秒）。

   ```
   call mysql.rds_set_source_delay(3600);
   ```
4. 使用 mysql.rds_start_replication 存储过程开始复制。

设置停止复制到只读副本的位置

停止复制到只读副本之后，可使用 mysql.rds_start_replication_until 存储过程开始复制，然后在指定二进制日志文件位置处停止复制。

开始复制到只读副本并在特定位置处停止复制

1. 通过使用 MySQL 客户端，以主用户的身份连接到源 MySQL 数据库实例。
2. 运行 mysql.rds_start_replication_until 存储过程。

   以下示例将启动复制并复制更改，直到它到达 `120` 二进制日志文件中的 `mysql-bin-changelog.000777` 位置。在灾难恢复方案中，假定位置 `120` 刚好位于灾难之前。

   ```
   call mysql.rds_start_replication_until(
   ```

```
    'mysql-bin-changelog.000777',
    120);
```

当到达停止点时，复制将自动停止。将生成以下 RDS 事件：`Replication has been stopped since the replica reached the stop point specified by the rds_start_replication_until stored procedure`。

提升只读副本

在灾难恢复方案中，当复制停止后，您可以将只读副本提升为新的源数据库实例。有关提升只读副本的信息，请参阅 将只读副本提升为独立的数据库实例。

使用 MySQL 更新只读副本

只读副本旨在支持读取查询，但您可能需要偶尔进行更新。例如，您可能需要添加索引，以优化访问副本的特定类型的查询。

尽管您可以通过将只读副本的数据库参数组中的 `read_only` 参数设置为 `0` 来启用更新，但我们建议您不要这样做，因为如果读取副本与源数据库实例不兼容，则这可能会导致问题。对于维护操作，我们建议您使用蓝绿部署。有关更多信息，请参阅 使用蓝绿部署进行数据库更新。

如果您在只读副本上禁用只读，请尽快将 `read_only` 参数的值改回为 `1`。

使用 MySQL 处理多可用区只读副本部署

您可从单可用区或多可用区数据库实例部署中创建只读副本。您可以使用多可用区部署提高关键数据的持久性和可用性，但无法使用多可用区为只读查询提供辅助服务。您可以改为从大流量、多可用区数据库实例创建只读副本以分载只读查询。如果多可用区部署的源实例故障转移到辅助可用区，则任何关联的只读副本都将自动切换为使用辅助可用区（现在为主可用区）作为其复制源。有关更多信息，请参阅"配置和管理 Amazon RDS 的多可用区部署"。

您可以将只读副本创建为多可用区数据库实例。Amazon RDS 会在另一个可用区中创建您的副本的备用，用于支持副本的故障转移。创建您的只读副本作为多可用区数据库实例与源数据库是否为多可用区数据库实例无关。

将级联只读副本用于 RDS for MySQL

RDS for MySQL 支持级联只读副本。使用级联只读副本，您可以扩展读取操作，而不会增加源 RDS for MySQL 数据库实例的开销。

使用级联只读副本，您的 RDS for MySQL 数据库实例会将数据发送到链中的第一个只读副本。然后，该只读副本将数据发送到链中的第二个副本，依此类推。最终结果是，链中的所有只读副本都具有来自 RDS for MySQL 数据库实例的更改，但不会只在源数据库实例上产生开销。

您可以从源 RDS for MySQL 数据库实例在链中创建一系列最多三个只读副本。例如，假设您具有 RDS for MySQL 数据库实例 `mysql-main`。您可执行以下操作：

- 从 `mysql-main` 开始，创建链中的第一个只读副本 `read-replica-1`。
- 接下来，从 `read-replica-1`，创建链中的下一个只读副本 `read-replica-2`。
- 最后，从 `read-replica-2`，创建链中的第三个只读副本 `read-replica-3`。

除了 `mysql-main` 系列中的第三个级联只读副本之外，您无法创建另一个只读副本。从 RDS for MySQL 源数据库实例到一系列级联只读副本末尾的完整实例系列最多可以包含四个数据库实例。

为了使级联只读副本正常工作，每个源 RDS for MySQL 数据库实例必须开启自动备份。要对只读副本开启自动备份，请先创建只读副本，然后修改只读副本以开启自动备份。有关更多信息，请参阅 创建只读副本。

与任何只读副本一样，您可以升级属于级联一部分的只读副本。从只读副本链中升级只读副本将从链中移除该副本。例如，假设您希望将一些工作负载从 `mysql-main` 数据库实例转移到新实例，以仅供会计部门使用。假设该示例中的链有三个只读副本，您决定升级 `read-replica-2`。该链受到如下影响：

- 升级 `read-replica-2` 会将其从复制链中移除。
  - 现在它是一个完全读/写数据库实例。
  - 它继续复制到 `read-replica-3`，就像在升级之前那样。
- 您的 `mysql-main` 继续复制到 `read-replica-1`。

有关升级只读副本的更多信息，请参阅将只读副本提升为独立的数据库实例。

监控 MySQL 只读副本的复制滞后

对于 MySQL 只读副本，可以在 Amazon CloudWatch 中通过查看 Amazon RDS `ReplicaLag` 指标来监控复制滞后。`ReplicaLag` 指标报告 `Seconds_Behind_Master` 命令的 `SHOW REPLICA STATUS` 字段的值。

MySQL 复制滞后的常见原因如下所示：

- 网络中断。
- 对只读副本上有不同索引的表进行写入操作。如果在只读副本上将 `read_only` 参数设置为 `0`，则在只读副本变得与源数据库实例不兼容时，复制会中断。对只读副本执行维护任务之后，我们建议您将 `read_only` 参数调整回 `1`。
- 使用 MyISAM 等非事务性存储引擎。仅 MySQL 上的 InnoDB 存储引擎支持复制。

当 `ReplicaLag` 指标达到 0 时，即表示副本已赶上源数据库实例进度。如果 `ReplicaLag` 指标返回 -1，则当前未激活复制。`ReplicaLag` = -1 等效于 `Seconds_Behind_Master` = `NULL`。

开始和停止 MySQL 只读副本复制

可通过调用系统存储过程 mysql.rds_stop_replication 和 mysql.rds_start_replication，在 Amazon RDS 数据库实例上停止再重新开始复制过程。对于长时间运行的操作 (如创建大型索引)，在两个 Amazon RDS 实例之间进行复制时可以这样做。在导入或导出数据库时，也需要停止再开始复制。有关更多信息，请参阅"将数据导入 Amazon RDS for MySQL 数据库实例并减少停机时间"和"使用复制从 MySQL 数据库实例中导出数据"。

如果复制连续 30 天停止，不论是手动还是由于复制错误，Amazon RDS 将终止源数据库实例与所有只读副本之间的复制。这样做是为了防止源数据库实例上的存储需求增长以及长故障转移时间。只读副本数据库实例仍可用。但是无法恢复复制，因为在终止复制后，已从源数据库实例中删除只读副本所需的二进制日志。您可为源数据库实例创建新的只读副本来重新建立复制。

排查 MySQL 只读副本问题

对于 MySQL 数据库实例，在某些情况下，只读副本将显示只读副本与其源数据库实例之间的复制错误或数据不一致项（或两者）。如果在只读副本或源数据库实例失败期间未刷新某些二进制日志 (binlog) 事件或 InnoDB 重做日志，则会出现此问题。在这些情况下，请手动删除并重新创建只读副本。您可通过设置以下参数值来降低发生这种情况的可能性：`sync_binlog=1` 和 `innodb_flush_log_at_trx_commit=1`。这些设置可能降低性能，因此，请先测试其影响，然后在生产环境中实施更改。

> **Warning**
>
> 建议在与源数据库实例关联的参数组中保留以下参数值：`sync_binlog=1` 和 `innodb_flush_log_at_trx_commit=1`。这些是动态参数。如果不想使用这些设置，建议在对源数据库实例执行可能导致其重启的任何操作之前，临时设置这些值。这些操作包括但不限于重启、使用故障转移重启、升级数据库版本以及更改数据库实例类或其存储。同样的建议也适用于创建源数据库实例的新只读副本。

如果不遵循此指示，则将增加只读副本显示只读副本与其源数据库实例之间的复制错误或数据不一致项（或两者）的风险。

MySQL 的复制技术是异步的。由于它们是异步的，因此，源数据库实例上偶发的 BinLogDiskUsage 会增多，而只读副本上应有 ReplicaLag。例如，对源数据库实例的大量写入操作可以并行进行。与之对比的是，对只读副本的写入操作使用单个 I/O 线程串行进行，这会导致源实例与只读副本之间存在滞后。有关 MySQL 文档中只读副本的更多信息，请参阅复制实施详细信息。

您可通过多种方式来减少对源数据库实例的更新与对只读副本的后续更新之间的滞后，例如：

- 将只读副本的存储大小和数据库实例类调整到与源数据库实例类似。
- 确保源数据库实例和只读副本使用的数据库参数组中的参数设置相兼容。有关更多信息和示例，请参阅本部分后面的有关 max_allowed_packet 参数的讨论。

Amazon RDS 监控只读副本的复制状态，如果由于任何原因停止复制，则将只读副本实例的 Replication State 字段更新为 Error。可能会有这样的例子，在您的只读副本上运行的 DML 查询与对源数据库实例的更新冲突。

可通过查看 Replication Error 字段，检查 MySQL 引擎引发的关联错误的详细信息。还生成指示只读副本状态的事件，包括 RDS-EVENT-0045、RDS-EVENT-0046 和 RDS-EVENT-0047。有关这些事件和事件订阅的详细信息，请参阅 使用 Amazon RDS 事件通知。如果返回 MySQL 错误消息，则检查 MySQL 错误消息文档中的错误编号。

一个可导致复制出错的常见问题是只读副本的 max_allowed_packet 参数的值小于源数据库实例的 max_allowed_packet 参数的值。max_allowed_packet 参数是您可以在数据库参数组中设置的自定义参数。您可以使用 max_allowed_packet 指定可在数据库上运行的 DML 代码的最大大小。有时候，与只读副本关联的数据库参数组中的 max_allowed_packet 值，要小于与源数据库实例关联的数据库参数组中的 max_allowed_packet 值。在这些情况下，复制过程可能会引发错误 Packet bigger than 'max_allowed_packet' bytes 并停止复制。如需修复错误，请让源数据库实例和只读副本使用具有相同 max_allowed_packet 参数值的数据库参数组。

其他可导致复制错误的常见情况包括：

- 对只读副本上的表进行写入操作。在某些情况下，您可能会在只读副本上创建索引，而该索引不同于源数据库实例上的索引。如果执行此操作，请将 read_only 参数设置为 0 以创建索引。如果您要写入到只读副本上的表，则在只读副本变得与源数据库实例不兼容时，复制会中断。对只读副本执行维护任务之后，我们建议您将 read_only 参数调整回 1。

- 使用非事务性存储引擎，如 MyISAM。只读副本需要使用事务性存储引擎。仅 MySQL 上的 InnoDB 存储引擎支持复制。
- 使用不安全的不确定性查询，如 `SYSDATE()`。有关更多信息，请参阅确定二进制日志记录中的安全和不安全语句。

如果您确定可安全跳过错误，那么可以按照跳过 RDS for MySQL 的当前复制错误部分中描述的步骤操作。否则，您可以先删除只读副本。然后，您可以使用相同的数据库实例标识符创建实例，以使终端节点保持与旧只读副本的终端节点相同。如果复制错误得到纠正，则 `Replication State` 将更改为 replicating。

使用基于 GTID 的复制

以下内容说明了如何在 Amazon RDS for MySQL 数据库实例间使用采用二进制日志（binlog）复制的全局事务标识符（GTID）。

如果您使用的是二进制日志复制，不熟悉 MySQL 的基于 GTID 的复制，请参阅 MySQL 文档中的 Replication with global transaction identifiers。

以下版本支持基于 GTID 的复制：

- 所有 RDS for MySQL 8.4 版本
- 所有 RDS for MySQL 8.0 版本
- 所有 RDS for MySQL 5.7 版本

复制配置中的所有 MySQL 数据库实例均必须满足该版本要求。

主题

全局事务标识符 (GTID) 概述

全局事务标识符 (GTID) 是为提交的 MySQL 事务生成的唯一标识符。您可以使用 GTID 让二进制日志复制的故障排除更加简单便捷。

MySQL 使用两种不同类型的事务进行二进制日志复制：

- GTID 事务 – 由 GTID 标识的事务。
- 匿名事务 – 未分配 GTID 的事务。

在复制配置中，GTID 在所有数据库实例中是唯一的。GTID 简化了复制配置，因为在使用它们时，您不必引用日志文件位置。通过使用 GTID，还可以更轻松地跟踪复制的事务并确定源实例和副本是否一致。

您可以使用基于 GTID 的复制，通过 RDS for MySQL 只读副本复制数据。您可以在创建新的只读副本时配置基于 GTID 的复制，也可以转换现有的只读副本以使用基于 GTID 的复制。

您还可以在与 RDS for MySQL 之间的延迟复制配置中使用基于 GTID 的复制。有关更多信息，请参阅使用 MySQL 配置延迟复制。

基于 GTID 的复制的参数

可以使用以下参数配置基于 GTID 的复制。

参数	有效值	描述
`gtid_mode`	`OFF`, `OFF_PERMISSIVE` ，`ON_PERMISSIVE` ，`ON`	`OFF` 指定新事务是匿名事务（即，没有 GTID），并且事务必须是匿名事务才能复制。 `OFF_PERMISSIVE` 指定新事务是匿名事务，但可以复制所有事务。 `ON_PERMISSIVE` 指定新事务是 GTID 事务，但可以复制所有事务。 `ON` 指定新事务是 GTID 事务，并且事务必须是 GTID 事务才能复制。

参数	有效值	描述
enforce_gtid_consistency	OFF, ON, WARN	OFF 允许事务违反 GTID 一致性。 ON 禁止事务违反 GTID 一致性。 WARN 允许事务违反 GTID 一致性，但在违反一致性时生成警告。

> **Note**
> 在 Amazon Web Services Management Console中，gtid_mode 参数显示为 gtid-mode。

对于基于 GTID 的复制，请为数据库实例或只读副本的参数组使用这些设置：

- ON 和 ON_PERMISSIVE 仅适用于从 RDS 数据库实例的传出复制。这两个值都可以让 RDS 数据库实例为复制的事务使用 GTID。ON 要求目标数据库也使用基于 GTID 的复制。ON_PERMISSIVE 让基于 GTID 的复制成为目标数据库上的可选项。
- OFF_PERMISSIVE（如果设置）表明您的 RDS 数据库实例可以接受来自源数据库的传入复制。无论源数据库是否使用基于 GTID 的复制，它们都可以这样做。
- OFF（如果设置）表明您的 RDS 数据库实例只接受来自不使用基于 GTID 的复制的源数据库的传入复制。

有关参数组的更多信息，请参阅 Amazon RDS 的参数组。

为 RDS for MySQL 的新只读副本启用基于 GTID 的复制

在为 RDS for MySQL 数据库实例启用基于 GTID 的复制时，将自动为数据库实例的只读副本配置基于 GTID 的复制。

为新的只读副本启用基于 GTID 的复制

1. 确保与数据库实例关联的参数组具有以下参数设置：

   - gtid_mode – ON 或 ON_PERMISSIVE
   - enforce_gtid_consistency – ON

有关使用参数组设置配置参数的更多信息，请参阅 Amazon RDS 的参数组。

2. 如果更改了数据库实例的参数组，请重新引导数据库实例。有关执行该操作的更多信息，请参阅重启中的数据库实例。
3. 创建数据库实例的一个或多个只读副本。有关执行该操作的更多信息，请参阅创建只读副本。

Amazon RDS 尝试使用 MASTER_AUTO_POSITION 在 MySQL 数据库实例和只读副本之间配置基于 GTID 的复制。如果尝试失败，Amazon RDS 将使用日志文件位置与只读副本之间进行复制。有关 MASTER_AUTO_POSITION 的更多信息，请参阅 MySQL 文档中的 GTID 自动定位。

为 RDS for MySQL 的现有只读副本启用基于 GTID 的复制

对于具有不使用基于 GTID 的复制的只读副本的现有 MySQL 数据库实例，您可以在数据库实例和只读副本之间配置基于 GTID 的复制。

为现有的只读副本启用基于 GTID 的复制

1. 如果数据库实例或任何只读副本使用了低于 8.0.26 的 RDS for MySQL 8.0.x 版本，请将数据库实例或只读副本升级为 8.0.26 或更高的 MySQL 8.0 版本。所有 RDS for MySQL 8.4 版本和 5.7 版本均支持基于 GTID 的复制。

   有关更多信息，请参阅 升级 RDS for MySQL 数据库引擎。
2. （可选）重置 GTID 参数并测试数据库实例和只读副本的行为：

   a. 确保与数据库实例和每个只读副本关联的参数组将 enforce_gtid_consistency 参数设置为 WARN。

      有关使用参数组设置配置参数的更多信息，请参阅 Amazon RDS 的参数组。
   b. 如果更改了数据库实例的参数组，请重新引导数据库实例。如果更改了只读副本的参数组，请重新引导只读副本。

      有关更多信息，请参阅“重启中的数据库实例”。
   c. 运行具有正常工作负载的数据库实例和只读副本并监控日志文件。

      如果看到有关与 GTID 不兼容的事务的警告，请调整您的应用程序，以使其仅使用与 GTID 兼容的功能。在继续执行下一步之前，请确保数据库实例未生成有关与 GTID 不兼容的事务的任何警告。
3. 为允许匿名事务的基于 GTID 的复制重置 GTID 参数，直到只读副本已处理所有这些事务。

   a. 确保与数据库实例和每个只读副本关联的参数组具有以下参数设置：

      - gtid_mode – ON_PERMISSIVE
      - enforce_gtid_consistency – ON

   b. 如果更改了数据库实例的参数组，请重新引导数据库实例。如果更改了只读副本的参数组，请重新引导只读副本。

4. 等待复制所有匿名事务。要检查是否复制了这些事务，请执行以下操作：

   a. 在源数据库实例上运行以下语句。

      MySQL 8.4

      ```
      SHOW BINARY LOG STATUS;
      ```

      MySQL 5.7 和 8.0

      ```
      SHOW MASTER STATUS;
      ```

      记下 File 和 Position 列中的值。

   b. 在每个只读副本上，使用上一步中的源实例上的文件和位置信息运行以下查询。

      ```
      SELECT MASTER_POS_WAIT('file', position);
      ```

      例如，如果文件名是 mysql-bin-changelog.000031 并且位置是 107，请运行以下语句。

      ```
      SELECT MASTER_POS_WAIT('mysql-bin-changelog.000031', 107);
      ```

      如果只读副本超过指定的位置，查询将立即返回。否则，该函数将等待一段时间。在查询返回所有只读副本的结果时，请继续执行下一步。

5. 仅重置基于 GTID 的复制的 GTID 参数。

   a. 确保与数据库实例和每个只读副本关联的参数组具有以下参数设置：

      - gtid_mode – ON
      - enforce_gtid_consistency – ON

   b. 重新引导数据库实例和每个只读副本。

6. 在每个只读副本上，运行以下过程。

   MySQL 8.4 及更高的主要版本

   ```
   CALL mysql.rds_set_source_auto_position(1);
   ```

   MySQL 8.0 及更低的主要版本

   ```
   CALL mysql.rds_set_master_auto_position(1);
   ```

为具有只读副本的 MySQL 数据库实例禁用基于 GTID 的复制

您可以为 一个具有只读副本的 MySQL 数据库实例。

为 具有只读副本的 MySQL 数据库实例禁用基于 GTID 的复制

1. 在每个只读副本上，运行以下过程：

   MySQL 8.4 及更高的主要版本

   ```
   CALL mysql.rds_set_source_auto_position(0);
   ```

   MySQL 8.0 及更低的主要版本

   ```
   CALL mysql.rds_set_master_auto_position(0);
   ```

2. 将 `gtid_mode` 重置为 `ON_PERMISSIVE`。

   a. 确保与 MySQL 数据库实例和每个只读副本关联的参数组将 `gtid_mode` 设置为 `ON_PERMISSIVE`。

      有关使用参数组设置配置参数的更多信息，请参阅 Amazon RDS 的参数组。

   b. 重新引导 MySQL 数据库实例和每个只读副本。有关重新引导的更多信息，请参阅重启中的数据库实例。

3. 将 `gtid_mode` 重置为 `OFF_PERMISSIVE`。

a. 确保与 MySQL 数据库实例和每个只读副本关联的参数组将 `gtid_mode` 设置为 `OFF_PERMISSIVE`。
b. 重新引导 MySQL 数据库实例和每个只读副本。

4. 等待在所有只读副本上应用所有 GTID 事务。要检查是否应用了这些事务，请按以下步骤操作：

   a. 在 MySQL 数据库实例上，运行以下命令：

      MySQL 8.4

      ```
      SHOW BINARY LOG STATUS
      ```

      MySQL 5.7 和 8.0

      ```
      SHOW MASTER STATUS
      ```

      您的输出应类似于以下输出。

      ```

      File                          Position
      ------------------------------------
      mysql-bin-changelog.000031      107
      ------------------------------------
      ```

      记下输出中的文件和位置。

   b. 在每个只读副本上，使用上一步中的源实例上的文件和位置信息运行以下查询：

      MySQL 8.4 和 MySQL 8.0.26 及更高的 MySQL 8.0 版本

      ```
      SELECT SOURCE_POS_WAIT('file', position);
      ```

      MySQL 5.7

      ```
      SELECT MASTER_POS_WAIT('file', position);
      ```

      例如，如果文件名是 `mysql-bin-changelog.000031` 并且位置是 `107`，请运行以下语句：

MySQL 8.4 和 MySQL 8.0.26 及更高的 MySQL 8.0 版本

```
SELECT SOURCE_POS_WAIT('mysql-bin-changelog.000031', 107);
```

MySQL 5.7

```
SELECT MASTER_POS_WAIT('mysql-bin-changelog.000031', 107);
```

5. 重置 GTID 参数以禁用基于 GTID 的复制。

   a. 确保与 MySQL 数据库实例和每个只读副本关联的参数组具有以下参数设置：

      - `gtid_mode` – `OFF`
      - `enforce_gtid_consistency` – `OFF`

   b. 重新引导 MySQL 数据库实例和每个只读副本。

配置与外部源实例之间的二进制日志文件位置复制

您可以使用二进制日志文件复制来设置 RDS for MySQL 或 MariaDB 数据库实例与 Amazon RDS 外部的 MySQL 或 MariaDB 实例之间的复制。

主题

- [开始前的准备工作]
- [配置与外部源实例之间的二进制日志文件位置复制]

开始前的准备工作

您可以使用复制的事务的二进制日志文件位置配置复制。

对 Amazon RDS 数据库实例启动复制功能所需的权限受到限制且对 Amazon RDS 主用户不可用。因此，请确保使用 Amazon RDS [mysql.rds_set_external_master（RDS for MariaDB 和 RDS for MySQL 主要版本 8.0 及更低版本）] 或 [mysql.rds_set_external_source（RDS for MySQL 主要版本 8.4 及更高版本）] 以及 [mysql.rds_start_replication] 命令来设置活动数据库和 Amazon RDS 数据库之间的复制。

要为 MySQL 或 MariaDB 数据库设置二进制日志记录格式，请更新 `binlog_format` 参数。如果您的数据库实例使用默认数据库实例参数组，请创建一个新的数据库参数组来修改 `binlog_format` 参数。在 MariaDB 和 MySQL 8.0 及更低版本中，`binlog_format` 默认设置为 `MIXED`。不过，如果您

需要特定的二进制日志 (binlog) 格式，也可以将 `binlog_format` 设置为 `ROW` 或 `STATEMENT`。重启您的数据库实例以使更改生效。在 MySQL 8.4 及更高版本中，`binlog_format` 默认设置为 `ROW`。

有关设置 `binlog_format` 参数的信息，请参阅为单可用区数据库配置 RDS for MySQL 二进制日志记录。有关不同 MySQL 复制类型的含义的信息，请参阅 MySQL 文档中的基于语句和基于行的复制的优点和缺点。

配置与外部源实例之间的二进制日志文件位置复制

对 Amazon RDS 设置外部源实例和副本时，请遵循以下准则：

- 监控作为副本的 Amazon RDS 数据库实例的失效转移事件。如果发生失效转移，则可能会在具有不同的网络地址的新主机上重新创建作为副本的数据库实例。有关如何监控失效转移事件的信息，请参阅使用 Amazon RDS 事件通知。
- 在您的源实例上维护二进制日志，直至您确认其已应用于副本。该维护确保在发生故障时可以还原源实例。
- 对 Amazon RDS 数据库实例启用自动备份。通过启用自动备份，可以确保在需要重新同步源实例和副本时能够将副本还原到特定时间点。有关备份和时间点还原的信息，请参阅备份、还原和导出数据。

配置与外部源实例之间的二进制日志文件复制

1. 将源 MySQL 或 MariaDB 实例设置为只读。

```
mysql> FLUSH TABLES WITH READ LOCK;
mysql> SET GLOBAL read_only = ON;
```

2. 对源 MySQL 或 MariaDB 实例运行 `SHOW MASTER STATUS` 命令以确定二进制日志位置。

   您将收到类似于以下示例的输出。

```
File                        Position
------------------------------------
 mysql-bin-changelog.000031      107
------------------------------------
```

3. 使用 `mysqldump` 将数据库从外部实例复制到 Amazon RDS 数据库实例。对于非常大的数据库，您可能需要使用将数据导入 Amazon RDS for MySQL 数据库实例并减少停机时间中的过程。

   对于 Linux、macOS 或 Unix：

```
mysqldump --databases database_name \
    --single-transaction \
    --compress \
    --order-by-primary \
    -u local_user \
    -plocal_password | mysql \
        --host=hostname \
        --port=3306 \
        -u RDS_user_name \
        -pRDS_password
```

对于 Windows：

```
mysqldump --databases database_name ^
    --single-transaction ^
    --compress ^
    --order-by-primary ^
    -u local_user ^
    -plocal_password | mysql ^
        --host=hostname ^
        --port=3306 ^
        -u RDS_user_name ^
        -pRDS_password
```

Note

确保 -p 选项和输入的密码之间没有空格。

要指定主机名、用户名、端口和密码以连接到 Amazon RDS 数据库实例，请在 `--host` 命令中使用 `--user (-u)`、`--port`、`-p` 和 `mysql` 选项。主机名是 Amazon RDS 数据库实例端点中的域名服务 (DNS) 名称，例如 `myinstance.123456789012.us-east-1.rds.amazonaws.com`。您可以在 Amazon Web Services Management Console上的实例详细信息中找到端点值。

4. 再次将源 MySQL 或 MariaDB 实例设置为可写。

```
mysql> SET GLOBAL read_only = OFF;
mysql> UNLOCK TABLES;
```

有关生成备份以用于复制的更多信息，请参阅 MySQL 文档。

5. 在 Amazon Web Services Management Console 中，将托管外部数据库的服务器的 IP 地址添加到 Amazon RDS 数据库实例的 Virtual Private Cloud (VPC) 安全组中。有关修改 VPC 安全组的更多信息，请参阅 Amazon Virtual Private Cloud 用户指南 中的您的 VPC 的安全组。

   在满足以下条件时，IP 地址可能会发生更改：

   - 您正在使用公有 IP 地址在外部源实例和数据库实例之间进行通信。
   - 外部源实例已停止并重新启动。

   如果满足这些条件，请在添加 IP 地址之前先对其进行验证。

   您可能还需要配置本地网络，以允许来自 Amazon RDS 数据库实例的 IP 地址的连接。通过执行此操作，您的本地网络可以与外部 MySQL 或 MariaDB 实例进行通信。要查找 Amazon RDS 数据库实例的 IP 地址，请使用 `host` 命令。

```
host db_instance_endpoint
```

   主机名是 Amazon RDS 数据库实例端点中的 DNS 名称。

6. 通过使用所选的客户端，连接到外部实例并创建要用于复制的用户。请仅将该账户用于复制，并将其限制为您的域以提高安全性。示例如下：

```
CREATE USER 'repl_user'@'mydomain.com' IDENTIFIED BY 'password';
```

> **Note**
> 作为安全最佳实践，请指定除此处所示提示以外的密码。

7. 对于外部 实例，向复制用户授予 `REPLICATION CLIENT` 和 `REPLICATION SLAVE` 权限。例如，要为您的域的“`REPLICATION CLIENT`”用户授予对所有数据库的 `REPLICATION SLAVE` 和 `repl_user` 权限，请发出以下命令。

```
GRANT REPLICATION CLIENT, REPLICATION SLAVE ON *.* TO 'repl_user'@'mydomain.com';
```

8. 将 Amazon RDS 数据库实例设置为副本。为此，请先以主用户身份连接到 Amazon RDS 数据库实例。然后使用 mysql.rds_set_external_source（RDS for MySQL 主要版本 8.4 及更高版本）或 mysql.rds_set_external_master（RDS for MariaDB 和 RDS for MySQL 主要版本 8.0 及更低版

本） 命令，将外部 MySQL 或 MariaDB 数据库标识为源实例。使用在步骤 2 中确定的主日志文件名和主日志位置。下面是一些命令示例。

MySQL 8.4

```
CALL mysql.rds_set_external_source ('mysourceserver.mydomain.com', 3306,
 'repl_user', 'password', 'mysql-bin-changelog.000031', 107, 1);
```

MariaDB 以及 MySQL 8.0 和 5.7

```
CALL mysql.rds_set_external_master ('mymasterserver.mydomain.com', 3306,
 'repl_user', 'password', 'mysql-bin-changelog.000031', 107, 1);
```

> **Note**
>
> 在 RDS for MySQL 上，可以选择通过运行 mysql.rds_set_external_source_with_delay（RDS for MySQL 主要版本 8.4 及更高版本） 或 mysql.rds_set_external_master_with_delay（RDS for MariaDB 和 RDS for MySQL 主要版本 8.0 及更低版本） 存储过程来使用延迟复制。在 RDS for MySQL 中，使用延迟复制的一个原因是，使用 mysql.rds_start_replication_until 存储过程打开灾难恢复。目前，RDS for MariaDB 支持延迟复制，但不支持 `mysql.rds_start_replication_until` 过程。

9. 在 Amazon RDS 数据库实例上，发出 mysql.rds_start_replication 命令以启动复制。

```
CALL mysql.rds_start_replication;
```

为 Amazon RDS for MySQL 配置多源复制

通过多源复制，您可以将 Amazon RDS for MySQL 数据库实例设置为一个副本，该副本接收来自多个 RDS for MySQL 源数据库实例的二进制日志事件。运行以下引擎版本的 RDS for MySQL 数据库实例支持多源复制：

- 所有 MySQL 8.4 版本
- 8.0.35 及更高的次要版本
- 5.7.44 及更高的次要版本

有关 MySQL 多源复制的信息，请参阅 MySQL 文档中的 [MySQL Multi-Source Replication]。MySQL 文档包含有关此功能的详细信息，而本主题则介绍如何在 RDS for MySQL 数据库实例上配置和管理多源复制通道。

多源复制的使用案例

以下情况很适合在 RDS for MySQL 上使用多源复制：

- 需要将不同数据库实例上的多个分片合并或组合成单个分片的应用程序。
- 需要根据从多个来源整合的数据生成报告的应用程序。
- 要求为分布在多个 RDS for MySQL 数据库实例中的数据创建长期合并备份。

多源复制的先决条件

在配置多源复制之前，请完成以下先决条件。

- 确保每个源 RDS for MySQL 数据库实例都启用了自动备份。启用自动备份可启用二进制日志记录。要了解如何启用自动备份，请参阅the section called "启用自动备份"。
- 为避免复制错误，建议您阻止对源数据库实例的写入操作。为此，您可以在附加到 RDS for MySQL 源数据库实例的自定义参数组中将 `read-only` 参数设置为 `ON`。您可以使用 Amazon Web Services Management Console或 Amazon CLI 来创建新的自定义参数组或修改现有的自定义参数组。有关更多信息，请参阅the section called "创建数据库参数组"和the section called "修改数据库参数组中的参数"。
- 对于每个源数据库实例，将实例的 IP 地址添加到多源数据库实例的 Amazon Virtual Private Cloud（VPC）安全组中。要确定源数据库实例的 IP 地址，可以运行命令 `dig` *`RDS Endpoint`*。在与目标多源数据库实例相同的 Amazon EC2 实例中运行命令。
- 对于每个源数据库实例，使用客户端连接到数据库实例，并创建一个具有复制所需权限的数据库用户，如下例所示。

```
CREATE USER 'repl_user' IDENTIFIED BY 'password';
GRANT REPLICATION CLIENT, REPLICATION SLAVE ON *.* TO 'repl_user';
```

> **Note**
>
> 从 MySQL 8.4 开始，`REPLICATION SLAVE` 权限已弃用并替换为 `REPLICATION REPLICA`。对于 8.4 及更高版本，请改用以下语法：

```
CREATE USER 'repl_user' IDENTIFIED BY 'password';
GRANT REPLICATION CLIENT, REPLICATION REPLICA ON *.* TO 'repl_user';
```

在 RDS for MySQL 数据库实例上配置多源复制通道

配置多源复制通道与配置单源复制类似。对于多源复制，首先在源实例上开启二进制日志记录。然后，将数据从源导入到多源副本。然后，您可以使用二进制日志坐标或使用 GTID 自动定位从每个源开始复制。

要将 RDS for MySQL 数据库实例配置为两个或更多 RDS for MySQL 数据库实例的多源副本，请执行以下步骤。

主题

步骤 1：将数据从源数据库实例导入到多源副本

在每个源数据库实例上执行以下步骤。

在将数据从源导入到多源副本之前，请通过运行 `SHOW MASTER STATUS` 命令来确定当前的二进制日志文件和位置。请记下这些详细信息，以便在下一个步骤中使用。在此示例输出中，文件为 `mysql-bin-changelog.000031`，位置为 `107`。

> **Note**
>
> 从 MySQL 8.4 开始，`SHOW MASTER STATUS` 命令已弃用并替换为 `SHOW BINARY LOG STATUS`。对于 8.4 及更高版本，请改用 `SHOW BINARY LOG STATUS`。

```
File                        Position
-----------------------------------
mysql-bin-changelog.000031      107
-----------------------------------
```

现在，使用 `mysqldump` 将数据库从源数据库实例复制到多源副本，如下例所示。

```
mysqldump --databases database_name \
 --single-transaction \
 --compress \
 --order-by-primary \
 -u RDS_user_name \
 -p RDS_password \
 --host=RDS Endpoint | mysql \
 --host=RDS Endpoint \
 --port=3306 \
 -u RDS_user_name \
-p RDS_password
```

复制数据库后，可以在源数据库实例上将只读参数设置为 OFF。

步骤 2：开始从源数据库实例复制到多源副本

对于每个源数据库实例，使用管理用户凭证连接到实例，然后运行以下两个存储过程。这些存储过程在通道上配置复制并开始复制。此示例会使用上一步的示例输出中的二进制日志文件名和位置。

```
CALL mysql.rds_set_external_source_for_channel('mysourcehost.example.com', 3306,
 'repl_user', 'password', 'mysql-bin-changelog.000031', 107, 1, 'channel_1');
CALL mysql.rds_start_replication_for_channel('channel_1');
```

有关使用这些存储过程和其它存储过程来设置和管理复制通道的更多信息，请参阅the section called “管理多源复制”。

将筛选条件与多源复制结合使用

您可以使用复制筛选条件来指定在多源副本中使用哪些数据库和表进行复制。复制筛选条件可以将数据库和表包含在复制之中或排除在复制之外。有关复制筛选条件的更多信息，请参阅the section called “配置复制筛选条件”。

使用多源复制，您可以全局或在通道级配置复制筛选条件。通道级筛选仅适用于运行 8.0 或 8.4 版本的受支持数据库实例。以下各示例介绍如何全局或在通道级配置筛选条件。

请注意多源复制中筛选的以下要求和行为：

- 通道名称前后需要使用反引号（``）。
- 如果您在参数组中更改复制筛选条件，则会对所有带有更新的通道重启多源副本的 sql_thread 以动态应用更改。如果更新涉及全局筛选条件，则所有处于运行状态的复制通道都将重启。

- 所有全局筛选条件都在任何通道特定的筛选条件之前应用。
- 如果在全局和通道级应用筛选条件，则仅应用通道级筛选条件。例如，如果筛选条件为 `replicate_ignore_db="db1,`channel_22`:db2"`，则 `replicate_ignore_db`（设置为 db1）将应用于除 channel_22 之外的所有通道，并且 channel_22 仅忽略来自 db2 的更改。

示例 1：设置全局筛选条件

在以下示例中，将 `temp_data` 数据库排除在每个通道的复制之外。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-parameter-group \
--db-parameter-group-name myparametergroup \
--parameters "ParameterName=replicate-ignore-
db,ParameterValue='temp_data',ApplyMethod=immediate"
```

示例 2：设置通道级筛选条件

在以下示例中，来自 `sample22` 数据库的更改仅包含在通道 `channel_22` 中。同样，来自 `sample99` 数据库的更改仅包含在通道 `channel_99` 中。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-parameter-group \
--db-parameter-group-name myparametergroup \
--parameters "ParameterName=replicate-do-db,ParameterValue='\`channel_22\`:sample22,
\`channel_99\`:sample99',ApplyMethod=immediate"
```

监控多源复制通道

您可以使用以下方法监控多源副本中的各个通道：

- 要监控所有通道或特定通道的状态，请连接到多源副本并运行 `SHOW REPLICA STATUS` 或 `SHOW REPLICA STATUS FOR CHANNEL 'channel_name'` 命令。有关更多信息，请参阅 MySQL 文档中的[检查复制状态]。
- 要在启动、停止或移除复制通道时接收通知，请使用 RDS 事件通知。有关更多信息，请参阅[the section called “使用 Amazon RDS 事件通知”]。
- 要监控特定通道的滞后，请检查它的 `ReplicationChannelLag` 指标。此指标的数据点的期间为 60 秒（1分钟），可使用 15 天。要查找通道的复制通道滞后，请使用实例标识符和复制通道名称。

要在此滞后超过特定阈值时收到通知，您可以设置 CloudWatch 警报。有关更多信息，请参阅the section called “使用 CloudWatch 监控 RDS”。

多源复制的注意事项和最佳实践

在 RDS for MySQL 上使用多源复制之前，请查看以下注意事项和最佳实践：

- 确保配置为多源副本的数据库实例具有足够的资源，例如吞吐量、内存、CPU 和 IOPS，以处理来自多个源实例的工作负载。
- 定期监控多源副本的资源利用率，调整存储或实例配置，以便在不造成资源紧张的情况下处理工作负载。
- 通过将系统变量 `replica_parallel_workers` 设置为大于 `0` 的值，可以在多源副本上配置多线程复制。在这种情况下，分配给每个通道的线程数就是此变量的值，再加上一个用于管理应用程序线程的协调器线程。
- 适当配置复制筛选条件以避免冲突。要将整个数据库复制到副本上的另一个数据库，可以使用 `--replicate-rewrite-db` 选项。例如，您可以将数据库 A 中的所有表复制到副本实例上的数据库 B。当所有源实例都使用相同的架构命名约定时，这种方法会很有用。有关 `--replicate-rewrite-db` 选项的信息，请参阅 MySQL 文档中的 Replica Server Options and Variables。
- 为避免复制错误，请避免写入副本。建议您在多源副本上启用 `read_only` 参数以阻止写入操作。这样做有助于消除由于写入操作冲突而导致的复制问题。
- 要提高在多源副本上执行的读取操作（例如排序和高负载联接）的性能，请考虑使用 RDS 优化型读取功能。此功能有助于处理依赖大型临时表或排序文件的查询。有关更多信息，请参阅the section called “使用 RDS 优化读取提高查询性能”。
- 为了显著减少复制滞后并提高多源副本的性能，请考虑启用优化型写入功能。有关更多信息，请参阅the section called “使用适用于 MySQL 的 RDS 优化型写入功能提高写入性能”。
- 一次对一个通道执行管理操作（例如更改配置），并避免从多个连接对多个通道执行更改。这些做法可能会导致复制操作发生冲突。例如，通过多个连接同时执行 `rds_skip_repl_error_for_channel` 和 `rds_start_replication_for_channel` 过程可能会导致跳过与预期不同的通道上的事件。
- 您可以在多源复制实例上启用备份，并将数据从该实例导出到 Amazon S3 存储桶，以便长期存储。但是，还要在各个源实例上配置具有适当保留期的备份，这一点很重要。有关将快照数据导出到 Amazon S3 的信息，请参阅the section called “将数据库快照数据导出到 Amazon S3”。
- 要在多源副本上分配读取工作负载，您可以从多源副本创建只读副本。您可以根据应用程序的要求将这些只读副本放在不同的 Amazon Web Services 区域中。有关只读副本的更多信息，请参阅 the section called “MySQL 只读副本”。

RDS for MySQL 上多源复制的限制

以下限制适用于 RDS for MySQL 上的多源复制：

- 目前，RDS for MySQL 支持为多源副本配置最多 15 个通道。
- 只读副本实例不能配置为多源副本。
- 要在运行引擎版本 5.7 的 RDS for MySQL 上配置多源复制，必须在副本实例上启用 Performance Schema。对于运行 8.0 或 8.4 版引擎的 RDS for MySQL，可选择是否启用 Performance Schema。
- 对于运行 5.7 版引擎的 RDS for MySQL，复制筛选条件适用于所有复制通道。对于运行 8.0 或 8.4 版引擎的 RDS for MySQL，可以配置适用于所有复制通道或单个通道的筛选条件。
- 还原 RDS 快照或执行时间点恢复（PITR）并不会还原多源副本通道配置。
- 创建多源副本的只读副本时，它只会复制多源实例中的数据。它不会还原任何通道配置。
- MySQL 不支持为每个通道设置不同数量的并行工作线程。每个通道根据 `replica_parallel_workers` 值获得相同数量的并行工作线程。

如果您的多源复制目标为多可用区数据库集群，则以下其它限制适用：

- 在对源 RDS for MySQL 实例进行任何写入之前，必须为该实例配置通道。
- 每个源 RDS for MySQL 实例都必须启用基于 GTID 的复制。
- 数据库集群上的失效转移事件会移除多源复制配置。还原该配置需要重复配置步骤。

为 RDS for MySQL 配置主动-主动集群

Amazon RDS 中的主动-主动集群是一种数据库配置，其中多个节点主动处理读取和写入操作，同时在实例之间分配工作负载来提高可用性和可扩展性。集群中的每个节点都经过同步以保持数据一致性，从而在节点出现故障时实现高可用性和更快的失效转移

您可以使用 MySQL 组复制插件为 RDS for MySQL 设置主动-主动集群。运行以下引擎版本的 RDS for MySQL 数据库实例支持组复制插件：

- 所有 MySQL 8.4 版本
- MySQL 8.0.35 及更高的次要版本

有关 MySQL 组复制的信息，请参阅 MySQL 文档中的 Group Replication。MySQL 文档包含有关此功能的详细信息，而本主题则介绍如何在 RDS for MySQL 数据库实例上配置和管理插件。

> **Note**
>
> 为简洁起见，本主题中所有提及的“主动-主动”集群均指使用 MySQL 组复制插件的主动-主动集群。

主动-主动集群的用例

以下情况很适合使用主动-主动集群：

- 需要集群中所有数据库实例才能支持写入操作的应用程序。组复制插件可使主动-主动集群中每个数据库实例的数据保持一致。有关这一点的工作原理的更多信息，请参阅 MySQL 文档中的 Group Replication。
- 需要数据库持续可用性的应用程序。对于主动-主动集群，数据将保留在集群中的所有数据库实例上。如果一个数据库实例出现故障，应用程序可以将流量重新路由到集群中的另一个数据库实例。
- 出于负载均衡的目的，可能需要在集群中的不同数据库实例之间拆分读取和写入操作的应用程序。使用主动-主动集群，您的应用程序可以将读取流量发送到特定的数据库实例，而将写入流量发送到其它数据库实例。您也可以随时切换要向哪些数据库实例发送读取或写入。

主题

主动-主动集群的限制和注意事项

Amazon RDS 中的主动-主动集群通过将工作负载分配到多个实例来增强可用性和可扩展性。但是，在使用此架构时，需要记住一些重要的限制和注意事项。

以下各节概述了诸如复制延迟、冲突解决、资源分配和失效转移行为等关键因素。了解这些注意事项有助于确保主动-主动集群部署实现理想的性能和可靠性。

主题

RDS for MySQL 主动-主动集群的限制

以下限制适用于 RDS for MySQL 的主动-主动集群。

- 对于主动-主动集群中的数据库实例，主用户名不能为 `rdsgrprepladmin`。此用户名保留用于组复制连接。
- 对于在主动-主动集群中具有只读副本的数据库实例，除 `Replicating` 之外的长时间复制状态可能会导致日志文件超出存储限制。有关只读副本的状态的信息，请参阅监控只读复制。
- 主动-主动集群中的数据库实例不支持蓝绿部署。有关更多信息，请参阅 使用 Amazon RDS 蓝绿部署进行数据库更新。
- 主动-主动集群中的数据库实例不支持 Kerberos 身份验证。有关更多信息，请参阅 对 Amazon RDS for MySQL 进行 Kerberos 身份验证。

- 无法将多可用区数据库集群中的数据库实例添加到主动-主动集群。但是，可以将多可用区数据库实例部署中的数据库实例添加到主动-主动集群。有关更多信息，请参阅 配置和管理 Amazon RDS 的多可用区部署。
- 由于组复制插件拒绝写入，因此不会在主动-主动集群中复制没有主键的表。
- 不会在主动-主动集群中复制非 InnoDB 表。
- 主动-主动集群不支持集群中不同数据库实例上的并发 DML 和 DDL 语句。
- 您无法将主动-主动集群配置为使用单主模式作为该组的复制模式。对于此配置，我们建议改用多可用区数据库集群。有关更多信息，请参阅 Amazon RDS 的多可用区数据库集群部署。
- 主动-主动集群中的数据库实例不支持多源复制。
- 跨区域主动-主动集群无法对组复制连接强制执行证书颁发机构（CA）验证。

RDS for MySQL 主动-主动集群的注意事项和最佳实践

在使用 RDS for MySQL 主动-主动集群之前，请查看以下注意事项和最佳实践：

- 主动-主动集群的数据库实例不能超过 9 个。
- 使用组复制插件，您可以控制主动-主动集群的事务一致性保证。有关更多信息，请参阅 MySQL 文档中的 Transaction Consistency Guarantees。
- 当不同的数据库实例更新主动-主动集群中的同一行时，可能会发生冲突。有关冲突和冲突解决的信息，请参阅 MySQL 文档中的 Group Replication。
- 为了实现容错，请在您的主动-主动集群中至少包含三个数据库实例。可以配置只具有一个或两个数据库实例的主动-主动集群，但该集群不具备容错能力。有关容错的信息，请参阅 MySQL 文档中的 Fault-tolerance。
- 当数据库实例加入现有的主动-主动集群并且运行的引擎版本与集群中的最低引擎版本相同时，该数据库实例将以读写模式加入。
- 当数据库实例加入现有的主动-主动集群并且运行的引擎版本高于集群中的最低引擎版本时，该数据库实例必须保持只读模式。
- 如果您通过在数据库参数组中将数据库实例的 `rds.group_replication_enabled` 参数设置为 `1` 来为它启用组复制，但复制尚未启动或未能启动，则该数据库实例将置于超级只读模式以防止数据不一致。有关超级只读模式的信息，请参阅 MySQL 文档。
- 您可以升级主动-主动集群中的数据库实例，但在主动-主动集群中的所有其它数据库实例都升级到相同的引擎版本或更高的引擎版本之前，该数据库实例是只读的。升级数据库实例时，升级完成后，数据库实例会自动加入同一个主动-主动集群。为避免数据库实例意外切换到只读模式，请对其禁用自动次要版本升级。有关升级 MySQL 数据库实例的信息，请参阅升级 RDS for MySQL 数据库引擎。

- 您可以将多可用区数据库实例部署中的数据库实例添加到现有的主动-主动集群中。您还可以将主动-主动集群中的单可用区数据库实例转换为多可用区数据库实例部署。如果多可用区部署中的主数据库实例出现故障，则该主实例会失效转移到备用实例。失效转移完成后，新的主数据库实例会自动加入同一个集群。有关多可用区数据库实例部署的更多信息，请参阅Amazon RDS 的多可用区数据库实例部署。
- 我们建议主动-主动集群中的数据库实例为其维护时段使用不同的时间范围。这种做法可以避免集群中的多个数据库实例同时脱机来进行维护。有关更多信息，请参阅 Amazon RDS 维护时段。
- 主动-主动集群可以使用 SSL 在数据库实例之间建立连接。要配置 SSL 连接，请设置 group_replication_recovery_use_ssl 和 group_replication_ssl_mode 参数。对于主动-主动集群中的所有数据库实例，这些参数的值必须匹配。

  目前，主动-主动集群不支持对 Amazon Web Services 区域之间的连接进行证书颁发机构（CA）验证。因此，对于跨区域集群，必须将 group_replication_ssl_mode 参数设置为 `DISABLED`（默认）或 `REQUIRED`。
- RDS for MySQL 主动-主动集群在多主模式下运行。group_replication_enforce_update_everywhere_checks 的默认值为 `ON`，参数是静态的。当此参数设置为 `ON` 时，应用程序无法插入到具有级联外键约束的表中。
- RDS for MySQL 主动-主动集群使用 MySQL 通信堆栈来保护连接安全，而不是 XCOM。有关更多信息，请参阅 MySQL 文档中的 Communication Stack for Connection Security Management。
- 当数据库参数组与主动-主动集群中的数据库实例关联时，我们建议仅将此数据库参数组与集群中的其它数据库实例相关联。
- 主动-主动集群仅支持 RDS for MySQL 数据库实例。这些数据库实例必须运行受支持的数据库引擎版本。
- 当主动-主动集群中的数据库实例出现意外故障时，RDS 会自动开始恢复该数据库实例。如果数据库实例无法恢复，我们建议将其替换为新的数据库实例，方法是使用集群中运行状况良好的数据库实例执行时间点恢复。有关说明，请参阅 使用时间点恢复将数据库实例添加到主动-主动集群。
- 您可以删除主动-主动集群中的数据库实例，而不会影响集群中的其它数据库实例。有关删除数据库实例的信息，请参阅删除数据库实例。
- 当数据库实例无意中离开主动-主动集群时，`group_replication_exit_state_action` 参数会更改为 `OFFLINE_MODE`。在此状态下，无法访问数据库实例，您必须重启数据库实例才能使其恢复在线状态并重新加入集群。可以通过修改自定义参数组中的 `group_replication_exit_state_action` 参数来更改此行为。通过将参数设置为 `READ_ONLY`，当数据库实例无意中离开集群时，它会进入超级只读状态而不是离线状态。

跨 VPC 主动-主动集群的准备工作

可以在多个 VPC 中配置包含 Amazon RDS for MySQL 数据库实例的主动-主动集群。VPC 可以位于相同的 Amazon Web Services 区域或不同的 Amazon Web Services 区域中。

> **Note**
>
> 在多个 Amazon Web Services 区域之间发送流量可能会产生额外费用。有关更多信息，请参阅常见架构的数据传输成本概览。

如果您要在单个 VPC 中配置主动-主动集群，可以跳过这些步骤并转入使用新的数据库实例设置主动-主动集群。

为在多个 VPC 中具有数据库实例的主动-主动集群做准备

1. 确保 CIDR 块中的 IPv4 地址范围满足以下要求：

   - VPC 的 CIDR 块中的 IPv4 地址范围不能重叠。
   - CIDR 块中的所有 IPv4 地址范围必须小于 `128.0.0.0/subnet_mask` 或高于 128.0.0.0/*subnet_mask*。

   以下范围说明了这些要求：

   - 支持在一个 VPC 中使用 `10.1.0.0/16`，而在另一个 VPC 中使用 `10.2.0.0/16`。
   - 支持在一个 VPC 中使用 `172.1.0.0/16`，而在另一个 VPC 中使用 `172.2.0.0/16`。
   - 不 支持在一个 VPC 中使用 `10.1.0.0/16`，而在另一个 VPC 中使用 `10.1.0.0/16`，因为范围重叠。
   - 不 支持在一个 VPC 中使用 `10.1.0.0/16`，而在另一个 VPC 中使用 `172.1.0.0/16`，因为一个低于 `128.0.0.0/subnet_mask`，另一个高于 `128.0.0.0/subnet_mask`。

   有关 CIDR 块的信息，请参阅《Amazon VPC 用户指南》中的 VPC CIDR 块。

2. 在每个 VPC 中，请确保 DNS 解析和 DNS 主机名都已启用。

   有关说明，请参阅《Amazon VPC 用户指南》中的查看和更新 VPC 的 DNS 属性。

3. 配置 VPC，以便您可以通过以下方式之一在它们之间路由流量：

- 在 VPC 之间创建 VPC 对等连接。

  有关说明，请参阅《Amazon VPC Peering Guide》中的 Create a VPC peering connection。在每个 VPC 中，确保您的安全组都有引用对等 VPC 中的安全组的入站规则。此操作将允许流量流入和流出与对等的 VPC 中的已引用安全组关联的实例。有关说明，请参阅《Amazon VPC 对等连接指南》中的更新安全组以引用对等安全组。
- 在 VPC 之间创建一个中转网关。

  有关说明，请参阅《Amazon VPC Transit Gateway》中的开始使用中转网关。在每个 VPC 中，确保您的安全组都有允许来自其它 VPC 的流量的入站规则，例如指定另一个 VPC 的 CIDR 的入站规则。此操作允许流量流入和流出与主动-主动集群中的已引用安全组关联的实例。有关更多信息，请参阅《Amazon VPC 用户指南》中的使用安全组控制到 Amazon 资源的流量。

主动-主动集群所需的参数设置

在 Amazon RDS for MySQL 中为主动-主动集群配置参数，对于保持一致的性能和运行稳定性至关重要。此表详细说明了控制复制、冲突解决和工作负载分配的关键参数。正确的配置可确保节点之间的高效同步，更大限度地减少复制滞后，并优化分布式或高流量环境中的资源利用率。

参数	描述	所需的设置
`binlog_format`	设置二进制日志记录格式。RDS for MySQL 8.0 及更低版本的默认值为 `MIXED`。RDS for MySQL 8.4 版的默认值为 `ROW`。有关更多信息，请参阅 MySQL 文档。	`ROW`
`enforce_gtid_consistency`	强制语句执行的 GTID 一致性。RDS for MySQL 的默认值为 `OFF`。有关更多信息，请参阅 MySQL 文档。	`ON`
`group_replication_group_name`	将组复制名称设置为 UUID。UUID 格式为 `11111111-2222-3333-4444-555555555555`	MySQL UUID

参数	描述	所需的设置
	。您可以通过连接到 MySQL 数据库实例并运行 SELECT UUID() 生成 MySQL UUID。对于主动-主动集群中的所有数据库实例，该值必须相同。有关更多信息，请参阅 MySQL 文档。	
gtid-mode	控制基于 GTID 的日志记录。RDS for MySQL 的默认值为 OFF_PERMISSIVE 。有关更多信息，请参阅 MySQL 文档。	ON
rds.custom_dns_resolution	指定是否允许从 VPC 中的 Amazon DNS 服务器进行 DNS 解析。当使用 rds.group_replication_enabled 参数启用组复制时，必须启用 DNS 解析。如果使用 rds.group_replication_enabled 参数禁用组复制，则无法启用 DNS 解析。有关更多信息，请参阅《Amazon VPC 用户指南》中的 Amazon DNS 服务器。	1
rds.group_replication_enabled	指定是否为数据库实例启用组复制。必须在主动-主动集群中的数据库实例上启用组复制。	1

参数	描述	所需的设置
replica_preserve_commit_order （RDS for MySQL 8.4 及更高版本）或 slave_preserve_commit_order （RDS for MySQL 8.0 版本）	控制在副本上提交事务的顺序。RDS for MySQL 的默认值为 ON。有关更多信息，请参阅 MySQL 文档。	ON

将现有数据库实例转换为主动-主动集群

要迁移到主动-主动集群的数据库实例的数据库引擎版本必须为以下版本之一：

- 所有 MySQL 8.4 版本
- MySQL 8.0.35 及更高的次要版本

如果您需要升级引擎版本，请参阅升级 RDS for MySQL 数据库引擎。

如果您要在多个 VPC 中设置具有数据库实例的主动-主动集群，请务必完成跨 VPC 主动-主动集群的准备工作中的先决条件。

完成以下步骤，以将现有数据库实例迁移到 RDS for MySQL 的主动-主动集群。

主题

步骤 1：在一个或多个自定义参数组中设置主动-主动集群参数

主动-主动集群中的 RDS for MySQL 数据库实例必须与一个自定义参数组相关联，该参数组对于所需参数具有正确的设置。有关参数和每个参数所需设置的信息，请参阅主动-主动集群所需的参数设置。

您可以在新的参数组或现有参数组中设置这些参数。但是，为避免意外影响不属于主动-主动集群的数据库实例，我们强烈建议您创建一个新的自定义参数组。主动-主动集群中的数据库实例可以与相同的数据库参数组或不同的数据库参数组相关联。

您可以使用 Amazon Web Services Management Console或 Amazon CLI 创建新的自定义参数组。有关更多信息，请参阅 在 Amazon RDS 中创建数据库参数组。以下示例运行 create-db-parameter-group Amazon CLI 命令来为 RDS for MySQL 8.0 创建名为 *myactivepg* 的自定义数据库参数组：

对于 Linux、macOS 或 Unix：

```
aws rds create-db-parameter-group \
  --db-parameter-group-name myactivepg \
  --db-parameter-group-family mysql8.0 \
  --description "Parameter group for active-active clusters"
```

对于 Windows：

```
aws rds create-db-parameter-group ^
  --db-parameter-group-name myactivepg ^
  --db-parameter-group-family mysql8.0 ^
  --description "Parameter group for active-active clusters"
```

还可以使用 Amazon Web Services Management Console或 Amazon CLI 设置自定义参数组中的参数。有关更多信息，请参阅 在 Amazon RDS 中修改数据库参数组中的参数。

以下示例运行 modify-db-parameter-group Amazon CLI 命令来为 RDS for MySQL 8.0 设置参数。要将此示例与 RDS for MySQL 8.4 结合使用，请将 `slave_preserve_commit_order` 更改为 `replica_preserve_commit_order`。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-parameter-group \
  --db-parameter-group-name myactivepg \
  --parameters
 "ParameterName='rds.group_replication_enabled',ParameterValue='1',ApplyMethod=pending-
reboot" \
```

```
 "ParameterName='rds.custom_dns_resolution',ParameterValue='1',ApplyMethod=pending-
reboot" \
 
 "ParameterName='enforce_gtid_consistency',ParameterValue='ON',ApplyMethod=pending-
reboot" \
               "ParameterName='gtid-mode',ParameterValue='ON',ApplyMethod=pending-
reboot" \
 
 "ParameterName='binlog_format',ParameterValue='ROW',ApplyMethod=immediate" \
 
 "ParameterName='slave_preserve_commit_order',ParameterValue='ON',ApplyMethod=immediate"
 \
 
 "ParameterName='group_replication_group_name',ParameterValue='11111111-2222-3333-4444-55555555
reboot"
```

对于 Windows：

```
aws rds modify-db-parameter-group ^
  --db-parameter-group-name myactivepg ^
  --parameters
 "ParameterName='rds.group_replication_enabled',ParameterValue='1',ApplyMethod=pending-
reboot" ^
 
 "ParameterName='rds.custom_dns_resolution',ParameterValue='1',ApplyMethod=pending-
reboot" ^
 
 "ParameterName='enforce_gtid_consistency',ParameterValue='ON',ApplyMethod=pending-
reboot" ^
               "ParameterName='gtid-mode',ParameterValue='ON',ApplyMethod=pending-
reboot" ^
 
 "ParameterName='binlog_format',ParameterValue='ROW',ApplyMethod=immediate" ^
 
 "ParameterName='slave_preserve_commit_order',ParameterValue='ON',ApplyMethod=immediate"
 ^
 
 "ParameterName='group_replication_group_name',ParameterValue='11111111-2222-3333-4444-55555555
reboot"
```

步骤 2：将数据库实例与设置了所需组复制参数的数据库参数组相关联

将数据库实例与您在上一步中创建或修改的参数组相关联。有关说明，请参阅 在 Amazon RDS 中将数据库参数组与数据库实例关联。

重启数据库实例以使新的参数设置生效。有关说明，请参阅 重启中的数据库实例。

步骤 3：创建主动-主动集群

在与数据库实例关联的数据库参数组中，将 `group_replication_group_seeds` 参数设置为要转换的数据库实例的端点。

可以使用 Amazon Web Services Management Console或 Amazon CLI 来设置参数。设置此参数后，您无需重启数据库实例。有关设置参数的更多信息，请参阅 在 Amazon RDS 中修改数据库参数组中的参数。

以下示例运行 modify-db-parameter-group Amazon CLI 命令来设置参数：

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-parameter-group \
  --db-parameter-group-name myactivepg \
  --parameters
 "ParameterName='group_replication_group_seeds',ParameterValue='myactivedb1.123456789012.us-east-1.rds.amazonaws.com:3306',ApplyMethod=immediate"
```

对于 Windows：

```
aws rds modify-db-parameter-group ^
  --db-parameter-group-name myactivepg ^
  --parameters
 "ParameterName='group_replication_group_seeds',ParameterValue='myactivedb1.123456789012.us-east-1.rds.amazonaws.com:3306',ApplyMethod=immediate"
```

步骤 4：为主动-主动集群创建其它 RDS for MySQL 数据库实例

要为主动-主动集群创建其它数据库实例，请对要转换的数据库实例执行时间点恢复。有关说明，请参阅 使用时间点恢复将数据库实例添加到主动-主动集群。

一个主动-主动集群最多可以有九个数据库实例。在数据库实例上执行时间点恢复，直到集群拥有所需的数据库实例数量。执行恢复点恢复时，请确保将您要添加的数据库实例与将

rds.group_replication_enabled 设置为 1 的数据库参数组相关联。否则，组复制将无法在新添加的数据库实例上启动。

步骤 5：在要转换的数据库实例上初始化组

初始化组并开始复制：

1. 在 SQL 客户端中连接到您要转换的该数据库实例。有关连接到 RDS for MySQL 数据库实例的更多信息，请参阅连接到 MySQL 数据库实例。
2. 在 SQL 客户端中，运行以下存储过程并将 *group_replication_user_password* 替换为 rdsgrprepladmin 用户的密码。在主动-主动集群中，rdsgrprepladmin 用户保留用于组复制连接。在主动-主动集群中的所有数据库实例上，该用户的密码必须相同。

   ```
   call mysql.rds_set_configuration('binlog retention hours', 168); -- 7 days binlog
   call mysql.rds_group_replication_create_user('group_replication_user_password');
   call
    mysql.rds_group_replication_set_recovery_channel('group_replication_user_password');
   call mysql.rds_group_replication_start(1);
   ```

   此示例将 binlog retention hours 值设置为 168，这意味着二进制日志文件在数据库实例上保留七天。您可以根据要求调整该值。

   此示例在 mysql.rds_group_replication_start 存储过程中指定 1，以使用当前数据库实例初始化新组。

   有关此示例中调用的存储过程的更多信息，请参阅管理主动-主动集群。

步骤 6：在主动-主动集群中的其它数据库实例上开始复制

对于主动-主动集群中的每个数据库实例，使用 SQL 客户端连接到该实例，然后运行以下存储过程。将 *group_replication_user_password* 替换为 rdsgrprepladmin 用户的密码。

```
call mysql.rds_set_configuration('binlog retention hours', 168); -- 7 days binlog
call mysql.rds_group_replication_create_user('group_replication_user_password');
call
 mysql.rds_group_replication_set_recovery_channel('group_replication_user_password');
call mysql.rds_group_replication_start(0);
```

此示例将 binlog retention hours 值设置为 168，这意味着二进制日志文件在每个数据库实例上保留七天。您可以根据要求调整该值。

此示例在 mysql.rds_group_replication_start 存储过程中指定 0，以将当前数据库实例加入现有组。

> **Tip**
>
> 确保在主动-主动集群中的所有其它数据库实例上运行这些存储过程。

步骤 7：（建议）检查主动-主动集群的状态

要确保集群的每个成员都配置正确，请通过连接到主动-主动集群中的数据库实例并运行以下 SQL 命令来检查集群的状态：

```
SELECT * FROM performance_schema.replication_group_members;
```

输出应针对每个数据库实例的 MEMBER_STATE 显示 ONLINE，如以下示例输出所示：

```
+---------------------------+--------------------------------------
+---------------+-------------+-------------+-------------+---------------
+---------------------------+
| CHANNEL_NAME              | MEMBER_ID                            | MEMBER_HOST    |
 MEMBER_PORT | MEMBER_STATE | MEMBER_ROLE | MEMBER_VERSION | MEMBER_COMMUNICATION_STACK
 |
+---------------------------+--------------------------------------
+---------------+-------------+-------------+-------------+---------------
+---------------------------+
| group_replication_applier | 9854d4a2-5d7f-11ee-b8ec-0ec88c43c251 | ip-10-15-3-137 |
     3306 | ONLINE       | PRIMARY     | 8.0.35         | MySQL                      |
| group_replication_applier | 9e2e9c28-5d7f-11ee-8039-0e5d58f05fef | ip-10-15-3-225 |
     3306 | ONLINE       | PRIMARY     | 8.0.35         | MySQL                      |
| group_replication_applier | a6ba332d-5d7f-11ee-a025-0a5c6971197d | ip-10-15-1-83  |
     3306 | ONLINE       | PRIMARY     | 8.0.35         | MySQL                      |
+---------------------------+--------------------------------------
+---------------+-------------+-------------+-------------+---------------
+---------------------------+
3 rows in set (0.00 sec)
```

有关可能的 MEMBER_STATE 值的信息，请参阅 MySQL 文档中的 Group Replication Server States。

使用新的数据库实例设置主动-主动集群

完成以下步骤，使用新的 Amazon RDS for MySQL 数据库实例设置主动-主动集群。

如果您要在多个 VPC 中设置具有数据库实例的主动-主动集群，请务必完成跨 VPC 主动-主动集群的准备工作中的先决条件。

主题

步骤 1：在一个或多个自定义参数组中设置主动-主动集群参数

主动-主动集群中的 RDS for MySQL 数据库实例必须与一个自定义参数组相关联，该参数组对于所需参数具有正确的设置。有关参数和每个参数所需设置的信息，请参阅主动-主动集群所需的参数设置。

您可以在新的参数组或现有参数组中设置这些参数。但是，为避免意外影响不属于主动-主动集群的数据库实例，我们强烈建议您创建一个新的自定义参数组。主动-主动集群中的数据库实例可以与相同的数据库参数组或不同的数据库参数组相关联。

您可以使用 Amazon Web Services Management Console或 Amazon CLI 创建新的自定义参数组。有关更多信息，请参阅 在 Amazon RDS 中创建数据库参数组。以下示例运行 create-db-parameter-group Amazon CLI 命令来为 RDS for MySQL 8.0 创建名为 *myactivepg* 的自定义数据库参数组：

对于 Linux、macOS 或 Unix：

```
aws rds create-db-parameter-group \
  --db-parameter-group-name myactivepg \
  --db-parameter-group-family mysql8.0 \
  --description "Parameter group for active-active clusters"
```

对于 Windows：

```
aws rds create-db-parameter-group ^
  --db-parameter-group-name myactivepg ^
  --db-parameter-group-family mysql8.0 ^
  --description "Parameter group for active-active clusters"
```

还可以使用 Amazon Web Services Management Console或 Amazon CLI 设置自定义参数组中的参数。有关更多信息，请参阅 在 Amazon RDS 中修改数据库参数组中的参数。

以下示例运行 modify-db-parameter-group Amazon CLI 命令来为 RDS for MySQL 8.0 设置参数。要将此示例与 RDS for MySQL 8.4 结合使用，请将 `slave_preserve_commit_order` 更改为 `replica_preserve_commit_order`。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-parameter-group \
  --db-parameter-group-name myactivepg \
  --parameters
 "ParameterName='rds.group_replication_enabled',ParameterValue='1',ApplyMethod=pending-
reboot" \

 "ParameterName='rds.custom_dns_resolution',ParameterValue='1',ApplyMethod=pending-
reboot" \

 "ParameterName='enforce_gtid_consistency',ParameterValue='ON',ApplyMethod=pending-
reboot" \
               "ParameterName='gtid-mode',ParameterValue='ON',ApplyMethod=pending-
reboot" \

 "ParameterName='binlog_format',ParameterValue='ROW',ApplyMethod=immediate" \

 "ParameterName='slave_preserve_commit_order',ParameterValue='ON',ApplyMethod=immediate"
 \

 "ParameterName='group_replication_group_name',ParameterValue='11111111-2222-3333-4444-55555555
reboot"
```

对于 Windows：

```
aws rds modify-db-parameter-group ^
  --db-parameter-group-name myactivepg ^
```

```
  --parameters
 "ParameterName='rds.group_replication_enabled',ParameterValue='1',ApplyMethod=pending-
reboot" ^

 "ParameterName='rds.custom_dns_resolution',ParameterValue='1',ApplyMethod=pending-
reboot" ^

 "ParameterName='enforce_gtid_consistency',ParameterValue='ON',ApplyMethod=pending-
reboot" ^
               "ParameterName='gtid-mode',ParameterValue='ON',ApplyMethod=pending-
reboot" ^

 "ParameterName='binlog_format',ParameterValue='ROW',ApplyMethod=immediate" ^

 "ParameterName='slave_preserve_commit_order',ParameterValue='ON',ApplyMethod=immediate"
 ^

 "ParameterName='group_replication_group_name',ParameterValue='11111111-2222-3333-4444-55555555
reboot"
```

步骤 2：为主动-主动集群创建新的 RDS for MySQL 数据库实例

以下 RDS for MySQL 数据库实例版本支持主动-主动集群。

- 所有 MySQL 8.4 版本
- MySQL version 8.0.35 及更高的次要版本

您可为集群创建最多九个新的数据库实例。

您可以使用 Amazon Web Services Management Console或 Amazon CLI 来创建新的数据库实例。有关创建数据库实例的更多信息，请参阅 创建 Amazon RDS 数据库实例。创建数据库实例时，将其与您在上一步中创建或修改的数据库参数组相关联。

步骤 3：指定主动-主动集群中的数据库实例

在与每个数据库实例关联的数据库参数组中，将 `group_replication_group_seeds` 参数设置为要包含在集群中的数据库实例的端点。

可以使用 Amazon Web Services Management Console或 Amazon CLI 来设置参数。设置此参数后，您无需重启数据库实例。有关设置参数的更多信息，请参阅 在 Amazon RDS 中修改数据库参数组中的参数。

以下示例运行 modify-db-parameter-group Amazon CLI 命令来设置参数：

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-parameter-group \
  --db-parameter-group-name myactivepg \
  --parameters
 "ParameterName='group_replication_group_seeds',ParameterValue='myactivedb1.123456789012.us-
east-1.rds.amazonaws.com:3306,myactivedb2.123456789012.us-
east-1.rds.amazonaws.com:3306,myactivedb3.123456789012.us-
east-1.rds.amazonaws.com:3306',ApplyMethod=immediate"
```

对于 Windows：

```
aws rds modify-db-parameter-group ^
  --db-parameter-group-name myactivepg ^
  --parameters
 "ParameterName='group_replication_group_seeds',ParameterValue='myactivedb1.123456789012.us-
east-1.rds.amazonaws.com:3306,myactivedb2.123456789012.us-
east-1.rds.amazonaws.com:3306,myactivedb3.123456789012.us-
east-1.rds.amazonaws.com:3306',ApplyMethod=immediate"
```

> **Tip**
>
> 确保在每个与主动-主动集群中的数据库实例关联的数据库参数组中设置 `group_replication_group_seeds` 参数。

步骤 4：在数据库实例上初始化组并开始复制

您可以选择任何新数据库来初始化组并开始复制。为此，请完成以下步骤：

1. 在主动-主动集群中选择一个数据库实例，然后在 SQL 客户端中连接到该数据库实例。有关连接到 RDS for MySQL 数据库实例的更多信息，请参阅连接到 MySQL 数据库实例。
2. 在 SQL 客户端中，运行以下存储过程并将 *group_replication_user_password* 替换为 `rdsgrprepladmin` 用户的密码。在主动-主动集群中，`rdsgrprepladmin` 用户保留用于组复制连接。在主动-主动集群中的所有数据库实例上，该用户的密码必须相同。

```
call mysql.rds_set_configuration('binlog retention hours', 168); -- 7 days binlog
call mysql.rds_group_replication_create_user('group_replication_user_password');
```

```
call
 mysql.rds_group_replication_set_recovery_channel('group_replication_user_password');
call mysql.rds_group_replication_start(1);
```

此示例将 `binlog retention hours` 值设置为 168，这意味着二进制日志文件在数据库实例上保留七天。您可以根据要求调整该值。

此示例在 `mysql.rds_group_replication_start` 存储过程中指定 1，以使用当前数据库实例初始化新组。

有关此示例中调用的存储过程的更多信息，请参阅管理主动-主动集群。

步骤 5：在主动-主动集群中的其它数据库实例上开始复制

对于主动-主动集群中的每个数据库实例，使用 SQL 客户端连接到该实例，然后运行以下存储过程。将 *group_replication_user_password* 替换为 `rdsgrprepladmin` 用户的密码。

```
call mysql.rds_set_configuration('binlog retention hours', 168); -- 7 days binlog
call mysql.rds_group_replication_create_user('group_replication_user_password');
call
 mysql.rds_group_replication_set_recovery_channel('group_replication_user_password');
call mysql.rds_group_replication_start(0);
```

此示例将 `binlog retention hours` 值设置为 168，这意味着二进制日志文件在每个数据库实例上保留七天。您可以根据要求调整该值。

此示例在 `mysql.rds_group_replication_start` 存储过程中指定 `0`，以将当前数据库实例加入现有组。

> **Tip**
>
> 确保在主动-主动集群中的所有其它数据库实例上运行这些存储过程。

步骤 6：（建议）检查主动-主动集群的状态

要确保集群的每个成员都配置正确，请通过连接到主动-主动集群中的数据库实例并运行以下 SQL 命令来检查集群的状态：

```
SELECT * FROM performance_schema.replication_group_members;
```

输出应针对每个数据库实例的 MEMBER_STATE 显示 ONLINE，如以下示例输出所示：

```
+---------------------------+-------------------------------------- 
+----------------+-------------+--------------+-------------+----------------
+----------------------------+
| CHANNEL_NAME              | MEMBER_ID                            | MEMBER_HOST    |
 MEMBER_PORT | MEMBER_STATE | MEMBER_ROLE | MEMBER_VERSION | MEMBER_COMMUNICATION_STACK
 |
+---------------------------+-------------------------------------- 
+----------------+-------------+--------------+-------------+----------------
+----------------------------+
| group_replication_applier | 9854d4a2-5d7f-11ee-b8ec-0ec88c43c251 | ip-10-15-3-137 |
      3306 | ONLINE       | PRIMARY     | 8.0.35           | MySQL                      |
| group_replication_applier | 9e2e9c28-5d7f-11ee-8039-0e5d58f05fef | ip-10-15-3-225 |
      3306 | ONLINE       | PRIMARY     | 8.0.35           | MySQL                      |
| group_replication_applier | a6ba332d-5d7f-11ee-a025-0a5c6971197d | ip-10-15-1-83  |
      3306 | ONLINE       | PRIMARY     | 8.0.35           | MySQL                      |
+---------------------------+-------------------------------------- 
+----------------+-------------+--------------+-------------+----------------
+----------------------------+
3 rows in set (0.00 sec)
```

有关可能的 MEMBER_STATE 值的信息，请参阅 MySQL 文档中的 Group Replication Server States。

步骤 7：（可选）将数据导入到主动-主动集群中的数据库实例

您可以将数据从 MySQL 数据库导入到主动-主动集群中的数据库实例。导入数据后，组复制会将其复制到集群中的其它数据库实例。

有关导入用户的信息，请参阅将数据导入 Amazon RDS for MySQL 数据库实例并减少停机时间。

将数据库实例添加到主动-主动集群

可以通过还原数据库快照或将数据库实例还原到某个时间点，将数据库实例添加到 Amazon RDS for MySQL 主动-主动集群。一个主动-主动集群最多可以包含九个数据库实例。

将数据库实例恢复到某个时间点时，它通常包含比从数据库快照还原的数据库实例较新的事务。当数据库实例具有较新的事务时，启动复制时需要应用的事务会较少。因此，使用时间点恢复将数据库实例添加到集群通常比从数据库快照还原更快。

主题

- 使用时间点恢复将数据库实例添加到主动-主动集群

- 使用数据库快照将数据库实例添加到主动-主动集群

使用时间点恢复将数据库实例添加到主动-主动集群

您可以通过对集群中的数据库实例执行时间点恢复，将数据库实例添加到主动-主动集群中。

有关在不同的 Amazon Web Services 区域将数据库实例恢复到某个时间点的信息，请参阅将自动备份复制到其他 Amazon Web Services 区域。

使用时间点恢复将数据库实例添加到主动-主动集群

1. 通过对主动-主动集群中的数据库实例执行时间点恢复，创建新的数据库实例。

   您可以对集群中的任何数据库实例执行时间点恢复以创建新的数据库实例。有关说明，请参阅 将 Amazon RDS 的数据库实例还原到指定时间。

   > **Important**
   >
   > 在时间点恢复期间，将新的数据库实例与设置了主动-主动集群参数的数据库参数组相关联。否则，组复制将无法在新的数据库实例上启动。有关参数和每个参数所需设置的信息，请参阅主动-主动集群所需的参数设置。

   > **Tip**
   >
   > 如果您在开始时间点恢复之前拍摄数据库实例的快照，则可以减少在新数据库实例上应用事务所需的时间量。

2. 将数据库实例添加到与主动-主动集群中的一个数据库实例相关联的每个数据库参数组中的 `group_replication_group_seeds` 参数，包括与新数据库实例关联的数据库参数组。

   有关设置参数的更多信息，请参阅 在 Amazon RDS 中修改数据库参数组中的参数。

3. 在 SQL 客户端中，连接到新的数据库实例，然后调用 mysql.rds_group_replication_set_recovery_channel 存储过程。将 *group_replication_user_password* 替换为 `rdsgrprepladmin` 用户的密码。

   ```
   call
    mysql.rds_group_replication_set_recovery_channel('group_replication_user_password');
   ```

4. 使用 SQL 客户端，调用 mysql.rds_group_replication_start 存储过程以开始复制：

```
call mysql.rds_group_replication_start(0);
```

使用数据库快照将数据库实例添加到主动-主动集群

通过在集群中创建数据库实例的数据库快照，然后还原数据库快照，可以将数据库实例添加到主动-主动集群。

有关将快照复制到不同 Amazon Web Services 区域的信息，请参阅the section called “跨区域复制”。

使用数据库快照将数据库实例添加到主动-主动集群

1. 创建主动-主动集群中数据库实例的数据库快照。

   您可以创建集群中任何数据库实例的数据库快照。有关说明，请参阅 为 Amazon RDS 的单可用区数据库实例创建数据库快照。
2. 从数据库快照还原数据库实例。

   在快照还原操作期间，将新的数据库实例与设置了主动-主动集群参数的数据库参数组相关联。有关参数和每个参数所需设置的信息，请参阅主动-主动集群所需的参数设置。

   有关从数据库快照还原数据库实例的信息，请参阅还原到数据库实例。
3. 将数据库实例添加到与主动-主动集群中的一个数据库实例相关联的每个数据库参数组中的 `group_replication_group_seeds` 参数，包括与新数据库实例关联的数据库参数组。

   有关设置参数的更多信息，请参阅 在 Amazon RDS 中修改数据库参数组中的参数。
4. 在 SQL 客户端中，连接到新的数据库实例，然后调用 mysql.rds_group_replication_set_recovery_channel 存储过程。将 *`group_replication_user_password`* 替换为 `rdsgrprepladmin` 用户的密码。

```
call
 mysql.rds_group_replication_set_recovery_channel('group_replication_user_password');
```

5. 使用 SQL 客户端，调用 mysql.rds_group_replication_start 存储过程以开始复制：

```
call mysql.rds_group_replication_start(0);
```

监控主动-主动集群

在 Amazon RDS for MySQL 中监控主动-主动集群，对于跟踪性能、复制完整性和节点同步至关重要。您可以通过连接到集群中的数据库实例并运行以下 SQL 命令来监控您的主动-主动集群：

```
SELECT * FROM performance_schema.replication_group_members;
```

输出应针对每个数据库实例的 MEMBER_STATE 显示 ONLINE，如以下示例输出所示：

```
+---------------------------+--------------------------------------
+----------------+-------------+--------------+-------------+----------------
+----------------------------+
| CHANNEL_NAME              | MEMBER_ID                            | MEMBER_HOST    |
 MEMBER_PORT | MEMBER_STATE | MEMBER_ROLE | MEMBER_VERSION | MEMBER_COMMUNICATION_STACK
 |
+---------------------------+--------------------------------------
+----------------+-------------+--------------+-------------+----------------
+----------------------------+
| group_replication_applier | 9854d4a2-5d7f-11ee-b8ec-0ec88c43c251 | ip-10-15-3-137 |
     3306 | ONLINE       | PRIMARY     | 8.0.35         | MySQL                      |
| group_replication_applier | 9e2e9c28-5d7f-11ee-8039-0e5d58f05fef | ip-10-15-3-225 |
     3306 | ONLINE       | PRIMARY     | 8.0.35         | MySQL                      |
| group_replication_applier | a6ba332d-5d7f-11ee-a025-0a5c6971197d | ip-10-15-1-83  |
     3306 | ONLINE       | PRIMARY     | 8.0.35         | MySQL                      |
+---------------------------+--------------------------------------
+----------------+-------------+--------------+-------------+----------------
+----------------------------+
3 rows in set (0.00 sec)
```

有关可能的 MEMBER_STATE 值的信息，请参阅 MySQL 文档中的 Group Replication Server States。

在主动-主动集群中的数据库实例上停止组复制

您可以在主动-主动集群中的数据库实例上停止组复制。停止组复制后，数据库实例将处于超级只读模式，直到重启复制或从主动-主动集群中移除该数据库实例。有关超级只读模式的信息，请参阅 MySQL 文档。

暂时停止主动-主动集群的组复制

1. 使用 SQL 客户端连接到主动-主动集群中的数据库实例。

   有关连接到 RDS for MySQL 数据库实例的更多信息，请参阅连接到 MySQL 数据库实例。

2. 在 SQL 客户端中，调用 mysql.rds_group_replication_stop 存储过程：

```
call mysql.rds_group_replication_stop();
```

重命名主动-主动集群中的数据库实例

您可以更改主动-主动集群中数据库实例的名称。要重命名主动-主动集群中的多个数据库实例，请一次重命名一个数据库实例。因此，先重命名一个数据库实例并将其重新加入集群，然后重命名下一个数据库库实例。

重命名主动-主动集群中的数据库实例

1. 在 SQL 客户端中连接到数据库实例，然后调用 mysql.rds_group_replication_stop 存储过程：

```
call mysql.rds_group_replication_stop();
```

2. 按照重命名数据库实例中的说明重命名数据库实例。
3. 在每个与主动-主动集群中的数据库实例关联的数据库参数组中修改 `group_replication_group_seeds` 参数。

   在参数设置中，将旧的数据库实例端点替换为新的数据库实例端点。有关设置参数的更多信息，请参阅 在 Amazon RDS 中修改数据库参数组中的参数。
4. 在 SQL 客户端中连接到数据库实例，然后调用 mysql.rds_group_replication_start 存储过程：

```
call mysql.rds_group_replication_start(0);
```

从主动-主动集群中移除数据库实例

当您从主动-主动集群中移除数据库实例时，它会恢复为独立的数据库实例。

从主动-主动集群中移除数据库实例

1. 在 SQL 客户端中连接到数据库实例，然后调用 mysql.rds_group_replication_stop 存储过程：

```
call mysql.rds_group_replication_stop();
```

2. 修改将保留在主动-主动集群中的数据库实例的 `group_replication_group_seeds` 参数。

在 group_replication_group_seeds 参数中，删除要从主动-主动集群中移除的数据库实例。有关设置参数的更多信息，请参阅 在 Amazon RDS 中修改数据库参数组中的参数。

3. 修改要从主动-主动集群中移除的数据库实例的参数，使其不再是集群的一部分。

   您可以将数据库实例与其它参数组关联，也可以修改与数据库实例关联的数据库参数组中的参数。要修改的参数包括 group_replication_group_name、rds.group_replication_enabled 和 group_replication_group_seeds。有关主动-主动集群参数的更多信息，请参阅主动-主动集群所需的参数设置。

   如果您修改数据库参数组中的参数，请确保该数据库参数组未与主动-主动集群中的其它数据库实例相关联。

4. 重启您从主动-主动集群中移除的数据库实例，以使新的参数设置生效。

   有关说明，请参阅 重启中的数据库实例。

使用复制从 MySQL 数据库实例中导出数据

可以使用复制功能，将数据从 RDS for MySQL 数据库实例导出到在 Amazon RDS 外部运行的 MySQL 实例。在这种情况下，MySQL 数据库实例是源 MySQL 数据库实例，而在 Amazon RDS 外部运行的 MySQL 实例则是外部 MySQL 数据库。

外部 MySQL 数据库可以在数据中心本地运行，也可以在 Amazon EC2 实例上运行。外部 MySQL 数据库必须运行与源 MySQL 数据库实例相同的版本或更高版本。

只有在从源 MySQL 数据库实例导出数据库期间，才支持复制到外部 MySQL 数据库。导出数据后，应终止复制，而应用程序可开始访问该外部 MySQL 实例。

下面列表显示了需要采取的步骤。每个步骤在后续的几节中有详述。

1. 准备外部 MySQL 数据库实例。
2. 准备用于复制的源 MySQL 数据库实例。
3. 使用 mysqldump 实用程序将数据库从源 MySQL 数据库实例传输到外部 MySQL 数据库。
4. 开始复制到外部 MySQL 数据库。
5. 导出完毕后，停止复制。

准备外部 MySQL 数据库

执行以下步骤来准备外部 MySQL 数据库。

准备外部 MySQL 数据库

1. 安装外部 MySQL 数据库。
2. 以主用户身份连接到外部 MySQL 数据库。然后创建支持访问数据库的管理员、应用程序和服务所需的用户。
3. 按照 MySQL 文档中的说明以副本的形式准备外部 MySQL 数据库。有关更多信息，请参阅 MySQL 文档中的 Setting the Replica Configuration。
4. 配置一条传出规则，使外部 MySQL 数据库在导出过程中作为只读副本运行。该传出规则将允许外部 MySQL 数据库在复制期间连接到源 MySQL 数据库实例。指定传出规则以允许通过传输控制协议 (TCP) 连接到源 MySQL 数据库实例的端口和 IP 地址。

   为您的环境指定适当的传出规则：

- 如果外部 MySQL 数据库在基于 Amazon VPC 服务的 Virtual Private Cloud (VPC) 中的 Amazon EC2 实例中运行，请在 VPC 安全组中指定传出规则。有关更多信息，请参阅 使用安全组控制访问权限。
- 如果外部 MySQL 数据库安装在本地，请在防火墙中指定传出规则。

5. 如果外部 MySQL 数据库在 VPC 中运行，则除了安全组传出规则外，还应配置 VPC 访问控制列表 (ACL) 规则：

   - 配置允许将 TCP 流量从源 MySQL 数据库实例的 IP 地址传入到端口 1024–65535 的 ACL 传入规则。
   - 配置允许将 TCP 流量传出到源 MySQL 数据库实例的端口 和 IP 地址的 ACL 传出规则。

   有关 Amazon VPC 网络 ACL 的更多信息，请参阅《Amazon VPC 用户指南》中的网络 ACL。
6. （可选）将 `max_allowed_packet` 参数设置为最大大小，以避免复制错误。建议使用此设置。

准备源 MySQL 数据库实例

执行以下步骤以复制源的形式准备源 MySQL 数据库实例。

准备源 MySQL 数据库实例

1. 确保设置复制时您的客户端计算机有足够的磁盘空间来保存二进制日志。
2. 连接到源 MySQL 数据库实例，并按照 MySQL 文档中 Creating a User for Replication 的说明创建复制账户。
3. 在运行源 MySQL 数据库实例的系统上配置传入规则，以允许外部 MySQL 数据库在复制期间连接。指定一条传入规则，以允许从外部 MySQL 数据库的 IP 地址通过 TCP 连接到源 MySQL 数据库实例使用的端口。
4. 指定传出规则：

   - 如果源 MySQL 数据库实例在 VPC 中运行，请在 VPC 安全组中指定传入规则。有关更多信息，请参阅 使用安全组控制访问权限。
5. 如果源 MySQL 数据库实例在 VPC 中运行，则除了安全组传入规则外，还要配置 VPC ACL 规则。

   - 配置 ACL 传入规则，以允许从外部 MySQL 数据库的 IP 地址通过 TCP 连接到 Amazon RDS 实例使用的端口。

- 配置 ACL 出口规则，以允许从端口 1024-65535 通过 TCP 连接到外部 MySQL 数据库的 IP 地址。

有关 Amazon VPC 网络 ACL 的更多信息，请参阅《Amazon VPC 用户指南》中的网络 ACL。

6. 确保设置的备份保留期足够长，以便在导出期间不会清除任何二进制日志。如果在导出完成前清除了所有日志，您必须从头重新启动复制进程。有关设置备份保留期的更多信息，请参阅 备份简介。
7. 使用 `mysql.rds_set_configuration` 存储过程将二进制日志文件保留期设置得足够长，以便在导出期间不会清除二进制日志文件。有关更多信息，请参阅 访问 MySQL 二进制日志。
8. 从源 MySQL 数据库实例创建 Amazon RDS 只读副本，以进一步确保不会清除源 MySQL 数据库实例的二进制日志文件。有关更多信息，请参阅 创建只读副本。
9. 在创建 Amazon RDS 只读副本后，调用 `mysql.rds_stop_replication` 存储过程以停止复制过程。源 MySQL 数据库实例不会再清除其二进制日志文件，以便这些文件可用于复制过程。
10. （可选）将 `max_allowed_packet` 和 `slave_max_allowed_packet` 参数都设置为最大大小，以避免复制错误。两个参数的最大大小为 1 GB。建议对两个参数都使用此设置。有关设置参数的信息，请参阅 在 Amazon RDS 中修改数据库参数组中的参数。

复制数据库

执行以下步骤以复制数据库。

复制数据库

1. 连接到源 MySQL 数据库实例的 RDS 只读副本，然后运行 MySQL `SHOW REPLICA STATUS\G` 语句。请记录下以下值：
   - `Master_Host`
   - `Master_Port`
   - `Master_Log_File`
   - `Exec_Master_Log_Pos`

> **Note**
>
> 以前的 MySQL 版本使用的是 SHOW SLAVE STATUS，而不是 SHOW REPLICA STATUS。如果您使用的 MySQL 版本低于 8.0.23，那么请使用 SHOW SLAVE STATUS。

2. 使用 mysqldump 实用程序创建快照，此过程会将数据从 Amazon RDS 复制到本地客户端计算机。确保您的客户端计算机有足够的空间容纳要复制的数据库中的 mysqldump 文件。为大量数据库完成此过程可能需要数小时的时间。按照 MySQL 文档中 [Creating a Data Snapshot Using mysqldump](#) 的说明进行操作。

   以下示例在客户端上运行 mysqldump，并将转储写入文件。

   对于 Linux、macOS 或 Unix：

```
mysqldump -h source_MySQL_DB_instance_endpoint \
    -u user \
    -ppassword \
    --port=3306 \
    --single-transaction \
    --routines \
    --triggers \
    --databases  database database2 > path/rds-dump.sql
```

   对于 Windows：

```
mysqldump -h source_MySQL_DB_instance_endpoint ^
    -u user ^
    -ppassword ^
    --port=3306 ^
    --single-transaction ^
    --routines ^
    --triggers ^
    --databases  database database2 > path\rds-dump.sql
```

   您可以将备份文件加载到外部 MySQL 数据库中。有关更多信息，请参阅 MySQL 文档中的[重新加载 SQL 格式备份](#)。您可以运行另一个实用程序以将数据加载到外部 MySQL 数据库中。

完成导出

执行以下步骤以完成导出。

完成导出

1. 使用 MySQL `CHANGE MASTER` 语句配置外部 MySQL 数据库。指定授予 `REPLICATION SLAVE` 权限的用户的 ID 和密码。指定您对 RDS 只读副本运行的 MySQL `SHOW REPLICA STATUS\G` 语句所返回的 `Master_Host`、`Master_Port`、`Relay_Master_Log_File` 和 `Exec_Master_Log_Pos` 值。有关更多信息，请参阅 MySQL 文档中的 [CHANGE MASTER TO Statement]。

   > **Note**
   >
   > 以前的 MySQL 版本使用的是 `SHOW SLAVE STATUS`，而不是 `SHOW REPLICA STATUS`。如果您使用的 MySQL 版本低于 8.0.23，那么请使用 `SHOW SLAVE STATUS`。

2. 使用 MySQL `START REPLICA` 命令启动从源 MySQL 数据库实例到外部 MySQL 数据库的复制。

   执行此操作将启动从源 MySQL 数据库实例进行复制的过程，并会导出在停止从 Amazon RDS 只读副本进行复制后发生的所有源更改。

   > **Note**
   >
   > 以前的 MySQL 版本使用的是 `START SLAVE`，而不是 `START REPLICA`。如果您使用的 MySQL 版本低于 8.0.23，那么请使用 `START SLAVE`。

3. 对外部 MySQL 数据库运行 MySQL `SHOW REPLICA STATUS\G` 命令，以验证它是否作为只读副本运行。有关解释结果的更多信息，请参阅 MySQL 文档中的 [SHOW SLAVE | REPLICA STATUS Statement]。

4. 在外部 MySQL 数据库上的复制跟上源 MySQL 数据库实例后，使用 MySQL `STOP REPLICA` 命令停止从源 MySQL 数据库实例的复制。

   > **Note**
   >
   > 以前的 MySQL 版本使用的是 `STOP SLAVE`，而不是 `STOP REPLICA`。如果您使用的 MySQL 版本低于 8.0.23，那么请使用 `STOP SLAVE`。

5. 在 Amazon RDS 只读副本上，调用 `mysql.rds_start_replication` 存储过程。这样，Amazon RDS 就能开始从源 MySQL 数据库实例清除二进制日志文件。

适用于 MySQL 数据库实例的选项

在下文中，您可以找到适用于运行 MySQL 数据库引擎的 Amazon RDS 实例的选项或其他功能的说明。要启用这些选项，您可以将它们添加到自定义选项组中，然后将选项组关联到数据库实例。有关使用选项组的详细信息，请参阅使用选项组。

Amazon RDS 支持 MySQL 的以下选项：

选项	选项 ID	引擎版本
MySQL 的 MariaDB 审计插件支持	`MARIADB_AUDIT_PLUGIN`	所有 MySQL 8.4 版本 MySQL 8.0.28 及更高的 8.0 版本 所有 MySQL 5.7 版本
MySQL memcached 支持	`MEMCACHED`	所有 MySQL 5.7 和 8.0 版本

MySQL 的 MariaDB 审计插件支持

Amazon RDS 基于开源 MariaDB 审计插件为 MySQL 数据库实例提供审计插件。有关更多信息，请参阅 MySQL Server GitHub 存储库的审计插件。

> **Note**
> MySQL 的审计插件基于 MariaDB 审计插件。在本文中，我们将其称为 MariaDB 审计插件。

MariaDB 审计插件记录数据库活动，包括用户登录数据库和针对数据库运行的查询等。数据库活动记录存储在日志文件中。

审计插件选项设置

Amazon RDS 支持 MariaDB 审计插件选项的以下设置。

选项设置	有效值	默认值	描述
SERVER_AUDIT_FILE_PATH	/rdsdbdata/log/audit/	/rdsdbdata/log/audit/	日志文件的位置。日志文件包含 SERVER_AUDIT_EVENTS 中指定的活动记录。有关更多信息，请参阅“查看和列出数据库日志文件”和“MySQL 数据库日志文件”。
SERVER_AUDIT_FILE_ROTATE_SIZE	1–1000000000	1000000	达到该字节大小时，会导致文件轮换。有关更多信息，请参阅“RDS for MySQL 数据库日志概览”。
SERVER_AUDIT_FILE_ROTATIONS	0–100	9	当 server_audit_output_type=file 时要保存的日志轮换数量。如果设置为 0，则日志文件从不轮换。有关更多信息，请参阅RDS for MySQL 数据库日志概览 和下载数据库日志文件。
SERVER_AUDIT_EVENTS	CONNECT, QUERY, QUERY_DDL,	CONNECT, QUERY	要在日志中记录的活动类型。安装 MariaDB 审核插件可自行登录。

选项设置	有效值	默认值	描述
	QUERY_DML ，QUERY_DML_NO_SELECT ，QUERY_DCL		• CONNECT：记录成功和失败的数据库连接以及数据库断开连接。 • QUERY：记录针对数据库运行的所有查询文本。 • QUERY_DDL ：类似于 QUERY 事件，不过仅返回数据定义语言 (DDL) 查询（CREATE、ALTER 等）。 • QUERY_DML ：类似于 QUERY 事件，不过仅返回数据操作语言 (DML) 查询（INSERT、UPDATE 等，以及 SELECT）。 • QUERY_DML_NO_SELECT ：与 QUERY_DML 事件类似，但不会记录 SELECT 查询。 • QUERY_DCL ：类似于 QUERY 事件，不过仅返回数据控制语言 (DCL) 查询（GRANT、REVOKE 等）。 MySQL 不支持 TABLE。
SERVER_AUDIT_INCL_USERS	多个逗号分隔值	无	仅包括指定用户的活动。默认情况下，会记录所有用户的活动。SERVER_AUDIT_INCL_USERS 和 SERVER_AUDIT_EXCL_USERS 是相互排斥的。如果将值添加到 SERVER_AUDIT_INCL_USERS ，请确保未向 SERVER_AUDIT_EXCL_USERS 中添加任何值。

选项设置	有效值	默认值	描述
SERVER_AUDIT_EXCL_USERS	多个逗号分隔值	无	排除指定用户的活动。默认情况下，会记录所有用户的活动。SERVER_AUDIT_INCL_USERS 和 SERVER_AUDIT_EXCL_USERS 是相互排斥的。如果将值添加到 SERVER_AUDIT_EXCL_USERS ，请确保未向 SERVER_AUDIT_INCL_USERS 中添加任何值。 rdsadmin 用户会每秒查询一次数据库，以检查数据库的运行状况。根据您的其他设置，此活动可能导致您的日志文件大小非常快速地增长。如果您不需要记录此活动，请将 rdsadmin 用户添加到 SERVER_AUDIT_EXCL_USERS 列表。 **Note** CONNECT会始终为所有用户记录 活动，即使是该选项设置指定的用户。
SERVER_AUDIT_LOGGING	ON	ON	日志记录处于活动状态。唯一有效值为 ON。Amazon RDS 不支持停用日志记录。如果您要停用日志记录，请删除 MariaDB 审核插件。有关更多信息，请参阅“删除 MariaDB 审核插件”。
SERVER_AUDIT_QUERY_LOG_LIMIT	0–2147483647	1024	记录中查询字符串的长度限制。

添加 MariaDB 审核插件

向数据库实例添加 MariaDB 审核插件的一般过程如下：

- 创建新的选项组，或者复制或修改现有选项组
- 向选项组添加选项
- 将选项组与数据库实例相关联

添加 MariaDB 审核插件后，您无需重新启动数据库实例。一旦选项组处于活动状态，审核便会立即开始。

> **Important**
> 将 MariaDB 审计插件添加到数据库实例可能会导致中断。我们建议在维护时段或数据库工作负载较低时添加 MariaDB 审计插件。

添加 MariaDB 审核插件

1. 确定您想要使用的选项组。您可以创建新的选项组，或使用现有选项组。如果您想使用现有选项组，请跳到下一步。否则，请创建自定义数据库选项组。为引擎选择 mysql，并为主引擎版本选择 5.7、8.0 或 8.4。有关更多信息，请参阅 创建选项组。
2. 将 MARIADB_AUDIT_PLUGIN 选项添加到选项组中，然后配置选项设置。有关添加选项的更多信息，请参阅 将选项添加到选项组。有关各项设置的更多信息，请参阅审计插件选项设置。
3. 将选项组应用到新的或现有的数据库实例。
   - 对于新数据库实例，您可以在启动实例时应用选项组。有关更多信息，请参阅“创建 Amazon RDS 数据库实例”。
   - 对于现有数据库实例，您可以通过修改实例并附加新的选项组来应用选项组。有关更多信息，请参阅 修改 Amazon RDS 数据库实例。

审计日志格式

日志文件表示为 UTF-8 格式的逗号分隔变量 (CSV) 文件。

> **Tip**
> 日志文件条目不按先后顺序排列。要对条目进行排序，请使用时间戳值。要查看最新事件，您可能需要查看所有日志文件。为了更加灵活地对日志数据进行排序和搜索，请启用设置将审计日志上传到 CloudWatch，再使用 CloudWatch 界面进行查看。

若要查看包含更多字段类型和 JSON 格式输出的审计数据，还可以使用数据库活动流功能。有关更多信息，请参阅 使用数据库活动流监控 Amazon RDS。

审核日志文件的各行按照指定顺序包含以下逗号分隔的信息：

字段	描述
timestamp	所记录的事件的 `YYYYMMDD` 后跟 `HH:MI:SS`（24 小时制）。
serverhost	记录了其事件的实例的名称。
username	已连接用户的用户名。
host	用户发起连接时所在的主机。
connectionid	所记录操作的连接 ID 号。
queryid	查询 ID 号，可用于查找关系表事件和相关查询。对于 `TABLE` 事件，添加多行。
operation	记录的操作类型。可能值为：`CONNECT`、`QUERY`、`READ`、`WRITE`、`CREATE`、`ALTER`、`RENAME` 和 `DROP`。
database	活动数据库，由 `USE` 命令设置。
object	对于 `QUERY` 事件，此值指示数据库执行的查询。对于 `TABLE` 事件，它指示表名。
retcode	所记录操作的返回代码。
connection_type	与服务器的连接的安全状态。可能的值有： • `0`：未定义 • `1`：TCP/IP • `2`：套接字 • `3`：命名管道 • `4`：SSL/TLS • `5`：共享内存

查看和下载 MariaDB 审计插件日志

启用 MariaDB 审计插件后，您将以访问其他任何基于文本的日志文件的相同方式来访问日志文件中的结果。审计日志文件位于 `/rdsdbdata/log/audit/`。有关在控制台中查看日志文件的信息，请参阅 查看和列出数据库日志文件。有关下载日志文件的信息，请参阅 下载数据库日志文件。

修改 MariaDB 审计插件设置

启用 MariaDB 审计插件后，您可以修改各项设置。有关如何修改选项设置的更多信息，请参阅 修改选项设置。有关各项设置的更多信息，请参阅审计插件选项设置。

删除 MariaDB 审核插件

Amazon RDS 不支持关闭 MariaDB 审核插件中的日志记录。不过，您可以从数据库实例删除插件。在删除 MariaDB 审核插件时，将自动重新启动数据库实例以停止审核。

要从数据库实例中删除 MariaDB 审核插件，请执行以下操作之一：

- 从 MariaDB 审核插件选项所属的选项组中删除 MariaDB 审核插件选项。此更改会影响使用该选项组的所有数据库实例。有关更多信息，请参阅"从选项组中删除选项"
- 修改数据库实例，并指定不包含插件的其他选项组。此更改会影响单个数据库实例。您可以指定默认 (空) 选项组，或指定其他自定义选项组。有关更多信息，请参阅"修改 Amazon RDS 数据库实例"。

MySQL memcached 支持

Amazon RDS 支持对 InnoDB 表使用 memcached 接口，该接口已在 MySQL 5.6 中引入。通过 memcached API，应用程序可以按照 NoSQL 密钥值数据存储的类似使用方式来使用 InnoDB 表。

> **Note**
>
> Memcached 接口在 MySQL 8.4 上不再可用。当您将数据库实例升级到 MySQL 8.4 时，必须在现有选项组中禁用 memcached。

memcached 接口是基于键的简单缓存。应用程序使用 memcached 插入、操作和检索缓存中的“键-值”数据对。MySQL 5.6 引入了一个插件来实现一个守护进程服务，该服务通过 memcached 协议公开 InnoDB 表中的数据。有关 MySQL memcached 插件的更多信息，请参阅 InnoDB 与 memcached 的集成。

实现对 RDS for MySQL 数据库实例的 memcached 支持

1. 确定用于控制对 memcached 接口的访问的安全组。如果已在使用 SQL 接口的应用程序集与访问 memcached 接口的应用程序集相同，则可使用 SQL 接口所用的现有 VPC 安全组。如果访问 memcached 接口的系列应用程序与之不同，则请定义新的 VPC 或数据库安全组。有关管理安全组的更多信息，请参阅 使用安全组控制访问权限。
2. 创建自定义数据库选项组，引擎类型选择 MySQL 并选择版本。有关创建选项组的更多信息，请参阅 创建选项组。
3. 将 MEMCACHED 选项添加到该选项组。指定 memcached 接口将使用的端口以及用于控制对接口的访问的安全组。有关添加选项的更多信息，请参阅 将选项添加到选项组。
4. 如有必要，修改选项设置以配置 memcached 参数。有关如何修改选项设置的更多信息，请参阅 修改选项设置。
5. 将选项组应用于实例。在应用选项组时，Amazon RDS 会为该实例启用 memcached 支持：
    - 您可以通过在启动实例时指定自定义选项组来启用新实例的 memcached 支持。有关启动 MySQL 实例的更多信息，请参阅 创建 Amazon RDS 数据库实例。
    - 您可以通过在修改实例时指定自定义选项组来启用现有实例的 memcached 支持。有关修改 数据库实例的更多信息，请参阅 修改 Amazon RDS 数据库实例。
6. 指定 MySQL 表中哪些列可以通过 memcached 接口进行访问。memcached 插件会在名为 containers 的专用数据库中创建名为 innodb_memcache 的目录表。在 containers 表中插

入行，使之映射通过 memcached 访问的 InnoDB 表。在 InnoDB 表中指定用于存储 memcached 键值的列，并指定一个或多个用于存储与该键关联的数据值的列。您还指定一个 memcached 应用程序用于引用列集的名称。有关在 containers 表中插入行的详细信息，请参阅 InnoDB memcached 插件内部信息。有关映射 InnoDB 表并通过 memcached 访问该表的示例，请参阅为 InnoDB memcached 插件编写应用程序。

7. 如果访问 memcached 接口的应用程序与使用 SQL 接口的应用程序位于不同的计算机或 EC2 实例上，请将这些计算机的连接信息添加到与 MySQL 实例关联的 VPC 安全组。有关管理安全组的更多信息，请参阅 使用安全组控制访问权限。

您通过修改该实例并指定 MySQL 版本的默认选项组来关闭实例的 memcached 支持。有关修改 数据库实例的更多信息，请参阅 修改 Amazon RDS 数据库实例。

MySQL memcached 安全注意事项

memcached 协议不支持用户身份验证。有关 MySQL memcached 安全注意事项的更多信息，请参阅 MySQL 文档中的 InnoDB memcached 插件的安全注意事项。

您可以采取下列操作来帮助提高 memcached 接口的安全性：

- 在向选项组添加 MEMCACHED 选项时，指定一个与默认 11211 不同的端口。
- 确保将 memcached 接口与用于限制对已知的可信客户端地址和 EC2 实例的访问的 VPC 安全组关联。有关管理安全组的更多信息，请参阅 使用安全组控制访问权限。

MySQL memcached 连接信息

要访问 memcached 接口，应用程序必须同时指定 Amazon RDS 实例的 DNS 名称和 memcached 端口号。例如，如果实例的 DNS 名称为 my-cache-instance.cg034hpkmmjt.region.rds.amazonaws.com，并且内存缓存接口使用端口 11212，PHP 中指定的连接信息将为：

```
<?php

$cache = new Memcache;
$cache->connect('my-cache-instance.cg034hpkmmjt.region.rds.amazonaws.com',11212);
?>
```

要查找 MySQL DB 实例的 DNS 名称和 memcached 端口，请执行以下操作：

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在 Amazon Web Services Management Console中的右上角，选择包含该数据库实例的地区。
3. 在导航窗格中，选择 Databases (数据库)。
4. 选择 MySQL 数据库实例名称以显示其详细信息。
5. 在连接部分，记下终端节点字段的值。DNS 名称与终端节点相同。另请注意，连接部分中的端口未用于访问 memcached 接口。
6. 在详细信息部分，记下选项组字段中列出的名称。
7. 在导航窗格中，选择选项组。
8. 选择 MySQL 数据库实例所用的选项组名称以显示该选项组的详细信息。在选项部分，记下 MEMCACHED 选项的端口设置值。

MySQL memcached 选项设置

Amazon RDS 将 MySQL memcached 参数公开为 Amazon RDS MEMCACHED 选项中的选项设置。

MySQL memcached 参数

- DAEMON_MEMCACHED_R_BATCH_SIZE – 一个整数，用于指定在执行 COMMIT 操作启动新的事务前应执行多少个 memcached 读取操作（获取）。允许的值是 1 到 4294967295；默认值为 1。此选项在重新启动实例后生效。
- DAEMON_MEMCACHED_W_BATCH_SIZE – 一个整数，用于指定在执行 COMMIT 操作启动新的事务前应执行多少个 memcached 写入操作，如添加、设置或者 incr。允许的值是 1 到 4294967295；默认值为 1。此选项在重新启动实例后生效。
- INNODB_API_BK_COMMIT_INTERVAL – 一个整数，用于指定多长时间自动提交一次使用 InnoDB memcached 接口的闲置连接。允许的值是 1 到 1073741824；默认值为 5。该选项会立即生效，无需重新启动实例。
- INNODB_API_DISABLE_ROWLOCK – 一个布尔值，该值在使用 InnoDB memcached 接口时禁用 (1 (true)) 或启用 (0 (false)) 行锁。默认为 0 (false)。此选项在重新启动实例后生效。
- INNODB_API_ENABLE_MDL –一个布尔值，设置为 0(false) 会锁定 InnoDB memcached 插件使用的表，以便 DDL 无法通过 SQL 接口删除或者更改该表。默认为 0 (false)。此选项在重新启动实例后生效。

- INNODB_API_TRX_LEVEL – 一个整数，用于指定 memcached 接口处理的查询的事务隔离级别。容许值是 0 到 3。默认值为 0。此选项在重新启动实例后生效。

Amazon RDS 配置以下 MySQL memcached 参数，这些参数无法修改：DAEMON_MEMCACHED_LIB_NAME、DAEMON_MEMCACHED_LIB_PATH 和 INNODB_API_ENABLE_BINLOG。在 Amazon RDS 中，也以单个 daemon_memcached_options 选项设置是形式提供 MySQL 管理员使用 MEMCACHED 设置的参数。

MySQL daemon_memcached_options 参数

- BINDING_PROTOCOL – 一个字符串，用于指定要使用的绑定协议。允许的值是 auto、ascii 或 binary。默认值为 auto，这意味着服务器与客户端自动协商协议。此选项在重新启动实例后生效。
- BACKLOG_QUEUE_LIMIT 一个整数，用于指定可等待 处理的网络连接的数目。memcached增大此限值会减少无法连接到 memcached 实例的客户端所收到的错误，但不会提高服务器性能。允许的值是 1 到 2048；默认值为 1024。此选项在重新启动实例后生效。
- CAS_DISABLED – 一个布尔值，该值启用 (1 (true)) 或禁用 (0 (false)) 比较和交换 (CAS)，它可使每个项大小减少 8 个字节。默认为 0 (false)。此选项在重新启动实例后生效。
- CHUNK_SIZE – 一个整数，用于指定为最小项的键、值和标志所分配的最小区块大小（计算单位：字节）。容许值是 1 到 48。默认为 48，且可以使用较低的值显著地提高内存效率。此选项在重新启动实例后生效。
- CHUNK_SIZE_GROWTH_FACTOR – 一个浮点数，用于控制新区块的大小。新区块的大小是前一区块时间的大小 CHUNK_SIZE_GROWTH_FACTOR。允许的值是 1 到 2；默认值为 1.25。此选项在重新启动实例后生效。
- ERROR_ON_MEMORY_EXHAUSTED – 一个布尔值，当该值设置为 1 (true) 时，将指定在没有更多内存来存储项时 memcached 将返回错误，而不会返回逐出项。如果设置为 0(false)，当没有更多内存时，memcached 将移出项目。默认为 0 (false)。此选项在重新启动实例后生效。
- MAX_SIMULTANEOUS_CONNECTIONS – 一个整数，用于指定最大并发连接数。将这个值设置为低于 10 的任何数值可防止 MySQL 启动。允许的值是 10 到 1024；默认值为 1024。此选项在重新启动实例后生效。
- VERBOSITY – 一个字符串，用于指定 memcached 服务在 MySQL 错误日志中记录的信息级别。默认值为 v。此选项在重新启动实例后生效。允许的许值为：
  - v – 记录运行主事件循环时出现的错误和警告。
  - vv – 除了 v 选项记录的信息之外，还会记录每个客户端命令和响应。
  - vvv – 除了 vv 选项记录的信息之外，还会记录内部状态转换。

Amazon RDS 配置这些 MySQL DAEMON_MEMCACHED_OPTIONS 参数，且无法进行修改：DAEMON_PROCESS、LARGE_MEMORY_PAGES、MAXIMUM_CORE_FILE_LIMIT、MAX_ITEM_SIZE、LO
和 USER。

MySQL 的参数

默认情况下，MySQL 数据库实例使用特定于 MySQL 数据库的数据库参数组。此参数组包含 MySQL 数据库引擎的参数。有关使用参数组和设置参数的信息，请参阅Amazon RDS 的参数组。

RDS for MySQL 参数设置为您已选择的存储引擎的默认值。有关 MySQL 参数的更多信息，请参阅 MySQL 文档。有关 MySQL 存储引擎的更多信息，请参阅 支持的 RDS for MySQL 存储引擎。

您可以使用 RDS 控制台或 Amazon CLI 查看特定 RDS for MySQL 版本的可用参数。有关在 RDS 控制台中查看 MySQL 参数组中参数的信息，请参阅 在 Amazon RDS 中查看数据库参数组的参数值。

使用 Amazon CLI，您可以通过运行 `describe-engine-default-parameters` 命令查看 RDS for MySQL 版本的参数。为 `--db-parameter-group-family` 选项指定下列值之一：

- `mysql8.4`
- `mysql8.0`
- `mysql5.7`

例如，要查看 RDS for MySQL 版本 8.0 的参数，请运行以下命令。

```
aws rds describe-engine-default-parameters --db-parameter-group-family mysql8.0
```

输出看上去类似于以下内容。

```
{
    "EngineDefaults": {
        "Parameters": [
            {
                "ParameterName": "activate_all_roles_on_login",
                "ParameterValue": "0",
                "Description": "Automatically set all granted roles as active after the
 user has authenticated successfully.",
                "Source": "engine-default",
                "ApplyType": "dynamic",
                "DataType": "boolean",
                "AllowedValues": "0,1",
                "IsModifiable": true
            },
            {
                "ParameterName": "allow-suspicious-udfs",
```

```
                "Description": "Controls whether user-defined functions that have only
 an xxx symbol for the main function can be loaded",
                "Source": "engine-default",
                "ApplyType": "static",
                "DataType": "boolean",
                "AllowedValues": "0,1",
                "IsModifiable": false
            },
            {
                "ParameterName": "auto_generate_certs",
                "Description": "Controls whether the server autogenerates SSL key and
 certificate files in the data directory, if they do not already exist.",
                "Source": "engine-default",
                "ApplyType": "static",
                "DataType": "boolean",
                "AllowedValues": "0,1",
                "IsModifiable": false
            },
        ...
```

要仅列出 RDS for MySQL 版本 8.0 的可修改参数，请运行以下命令。

对于 Linux、macOS 或 Unix：

```
aws rds describe-engine-default-parameters --db-parameter-group-family mysql8.0 \
   --query 'EngineDefaults.Parameters[?IsModifiable==`true`]'
```

对于 Windows：

```
aws rds describe-engine-default-parameters --db-parameter-group-family mysql8.0 ^
   --query "EngineDefaults.Parameters[?IsModifiable==`true`]"
```

MySQL 数据库实例的数据库管理员常见任务

您可以在下面的内容中找到一些常见数据管理员任务的 Amazon RDS 特定实施描述，这些任务适用于运行 MySQL 数据库引擎的数据库实例。为了提供托管服务体验，Amazon RDS 未提供数据库实例的 Shell 访问权限。并且它还限制了对需要高级权限的某些系统程序和表的访问。

有关在 Amazon RDS 上使用 MySQL 日志文件的信息，请参阅 MySQL 数据库日志文件。

了解预定义用户

Amazon RDS 使用新的 RDS for MySQL 数据库实例自动创建若干预定义用户。无法更改预定义用户及其权限。无法删除、重命名或修改这些预定义用户的权限。此类尝试会导致错误。

- rdsadmin – 具有 `superuser` 权限的管理员将对独立的 PostgreSQL 数据库执行许多管理任务，此用户专为处理这些管理任务而创建。此用户供 RDS for MySQL 在内部执行许多管理任务。
- rdsrepladmin - Amazon RDS 在内部使用的用户，用于支持 RDS for MySQL 数据库实例和集群上的复制活动。

有关其它常见 DBA 任务的信息，请参阅以下主题。

主题

- RDS for MySQL 的基于角色的权限模型
- 适用于 RDS for MySQL 的动态权限
- 结束 RDS for MySQL 的会话或查询
- 跳过 RDS for MySQL 的当前复制错误
- 使用 InnoDB 表空间改善 RDS for MySQL 的崩溃恢复时间
- 管理 RDS for MySQL 的全局状态历史记录
- 在 MySQL 8.4 中配置缓冲池大小和重做日志容量

RDS for MySQL 的基于角色的权限模型

从 RDS for MySQL 版本 8.0.36 开始，您无法直接修改 `mysql` 数据库中的表。特别是，您不能通过对 `grant` 表执行数据操作语言（DML）操作来创建数据库用户。相反，您可以使用 MySQL 账户管理语句（例如 `CREATE USER`、`GRANT`、和 `REVOKE`）向用户授予基于角色的权限。您也无法创建其他类型的对象，例如 `mysql` 数据库中的存储过程。您仍然可以查询 `mysql` 表。如果您使用二进制日志复制，则直接对源数据库实例上的 `mysql` 表进行的更改不会复制到目标集群中。

在某些情况下，您的应用程序可能会使用快捷方式通过插入到 mysql 表来创建用户或其他对象。如果是这样，请更改应用程序代码以使用相应的语句，例如 CREATE USER。

要在从外部 MySQL 数据库迁移期间导出数据库用户的元数据，请使用以下方法之一：

- 将 MySQL Shell 的实例转储实用程序与筛选条件一起使用，来排除用户、角色和授权。以下示例显示了要使用的命令语法。确保 outputUrl 为空。

  ```
  mysqlsh user@host -- util.dumpInstance(outputUrl,{excludeSchemas:['mysql'],users:
   true})
  ```

  有关更多信息，请参阅《MySQL 参考手册》中的 Instance Dump Utility, Schema Dump Utility, and Table Dump Utility。
- 使用 mysqlpump 客户端实用程序。此示例包括除 mysql 系统数据库中的表之外的所有表。它还包括用于重现迁移数据库中的所有 MySQL 用户的 CREATE USER 和 GRANT 语句。

  ```
  mysqlpump --exclude-databases=mysql --users
  ```

  MySQL 8.4 中不再提供 mysqlpump 客户端实用工具。请改用mysqldump。

为了简化对许多用户或应用程序的权限管理，您可以使用 CREATE ROLE 语句来创建具有一组权限的角色。然后，您可以使用 GRANT 和 SET ROLE 语句以及 current_role 函数将角色分配给用户或应用程序、切换当前角色以及检查哪些角色有效。有关 MySQL 8.0 中基于角色的权限系统的更多信息，请参阅 MySQL 参考手册中的使用角色。

> **Important**
>
> 我们强烈建议不要直接在应用程序中使用主用户。请遵守使用数据库用户的最佳实践，按照您的应用程序所需的最少权限创建用户。

从版本 8.0.36 开始，RDS for MySQL 包括一个具有以下所有权限的特殊角色。该角色命名为 rds_superuser_role。每个数据库实例的主管理用户已经获得了此角色。rds_superuser_role 角色包括所有数据库对象的以下权限：

- ALTER
- APPLICATION_PASSWORD_ADMIN

- `ALTER ROUTINE`
- `CREATE`
- `CREATE ROLE`
- `CREATE ROUTINE`
- `CREATE TEMPORARY TABLES`
- `CREATE USER`
- `CREATE VIEW`
- `DELETE`
- `DROP`
- `DROP ROLE`
- `EVENT`
- `EXECUTE`
- `INDEX`
- `INSERT`
- `LOCK TABLES`
- `PROCESS`
- `REFERENCES`
- `RELOAD`
- `REPLICATION CLIENT`
- `REPLICATION SLAVE`
- `ROLE_ADMIN`
- `SET_USER_ID`
- `SELECT`
- `SHOW DATABASES`
- `SHOW VIEW`
- `TRIGGER`
- `UPDATE`
- `XA_RECOVER_ADMIN`

角色定义还包括 WITH GRANT OPTION，以便管理用户可以将该角色授予其他用户。特别是，管理员必须授予以 MySQL 集群作为目标执行二进制日志复制所需的任何权限。

> **Tip**
>
> 要查看权限的完整详细信息，请使用以下语句。
>
> ```
> SHOW GRANTS FOR rds_superuser_role@'%';
> ```

当您使用 RDS for MySQL 版本 8.0.36 和更高版本中的角色授予访问权限时，还可以通过使用 SET ROLE *role_name* 或 SET ROLE ALL 语句来激活角色。下面的示例演示如何操作。将适当的角色名称替换为 CUSTOM_ROLE。

```
# Grant role to user
mysql> GRANT CUSTOM_ROLE TO 'user'@'domain-or-ip-address'

# Check the current roles for your user. In this case, the CUSTOM_ROLE role has not
 been activated.
# Only the rds_superuser_role is currently in effect.
mysql> SELECT CURRENT_ROLE();
+--------------------------+
| CURRENT_ROLE()           |
+--------------------------+
| `rds_superuser_role`@`%` |
+--------------------------+
1 row in set (0.00 sec)

# Activate all roles associated with this user using SET ROLE.
# You can activate specific roles or all roles.
# In this case, the user only has 2 roles, so we specify ALL.
mysql> SET ROLE ALL;
Query OK, 0 rows affected (0.00 sec)

# Verify role is now active
mysql> SELECT CURRENT_ROLE();
+-----------------------------------------------------+
| CURRENT_ROLE()                                      |
+-----------------------------------------------------+
| `CUSTOM_ROLE`@`%`,`rds_superuser_role`@`%` |
+-----------------------------------------------------+
```

适用于 RDS for MySQL 的动态权限

动态权限是可以使用 GRANT 语句显式授予的 MySQL 权限。根据 RDS for MySQL 版本，RDS 支持您仅授予特定的动态权限。RDS 不支持使用其中某些权限，因为这些权限可能会干扰特定的数据库操作，例如复制和备份。

下表显示了可以针对不同的 MySQL 版本授予其中哪些权限。如果要从低于 8.0.36 的 MySQL 版本升级到 8.0.36 或更高版本，则当不再支持授予特定权限时，可能需要更新应用程序代码。

特权	MySQL 8.0.35 及更低版本	MySQL 8.0.36 及更高的次要版本	MySQL 8.4.3 及更高版本
ALLOW_NONEXISTENT_DEFINER	不可用	不可用	不允许
APPLICATION_PASSWORD_ADMIN	已允许	已允许	允许
AUDIT_ABORT_EXEMPT	允许	不允许	不允许
AUDIT_ADMIN	不允许	不允许	不允许
AUTHENTICATION_POLICY_ADMIN	允许	不允许	不允许
BACKUP_ADMIN	允许	不允许	不允许
BINLOG_ADMIN	允许	不允许	不允许
BINLOG_ENCRYPTION_ADMIN	不允许	不允许	不允许
CLONE_ADMIN	不允许	不允许	不允许
CONNECTION_ADMIN	允许	不允许	不允许

特权	MySQL 8.0.35 及更低版本	MySQL 8.0.36 及更高的次要版本	MySQL 8.4.3 及更高版本
ENCRYPTION_KEY_ADMIN	不允许	不允许	不允许
FIREWALL_ADMIN	不允许	不允许	不允许
FIREWALL_EXEMPT	允许	不允许	不允许
FIREWALL_USER	不允许	不允许	不允许
FLUSH_OPTIMIZER_COSTS	已允许	已允许	允许
FLUSH_PRIVILEGES	不可用	不可用	允许
FLUSH_STATUS	已允许	已允许	允许
FLUSH_TABLES	已允许	已允许	允许
FLUSH_USER_RESOURCES	已允许	已允许	允许
GROUP_REPLICATION_ADMIN	不允许	不允许	不允许
GROUP_REPLICATION_STREAM	不允许	不允许	不允许
INNODB_REDO_LOG_ARCHIVE	不允许	不允许	不允许
INNODB_REDO_LOG_ENABLE	不允许	不允许	不允许
MASKING_DICTIONARIES_ADMIN	不允许	不允许	不允许

特权	MySQL 8.0.35 及更低版本	MySQL 8.0.36 及更高的次要版本	MySQL 8.4.3 及更高版本
NDB_STORED_USER	不允许	不允许	不允许
OPTIMIZE_LOCAL_TABLE	不可用	不可用	不允许
PASSWORDLESS_USER_ADMIN	不允许	不允许	不允许
PERSIST_RO_VARIABLES_ADMIN	不允许	不允许	不允许
REPLICATION_APPLIER	允许	不允许	不允许
REPLICATION_SLAVE_ADMIN	不允许	不允许	不允许
RESOURCE_GROUP_ADMIN	允许	不允许	不允许
RESOURCE_GROUP_USER	允许	不允许	不允许
ROLE_ADMIN	已允许	已允许	允许
SENSITIVE_VARIABLES_OBSERVER	已允许	已允许	允许
SERVICE_CONNECTION_ADMIN	允许	不允许	不允许
SESSION_VARIABLES_ADMIN	已允许	已允许	允许

特权	MySQL 8.0.35 及更低版本	MySQL 8.0.36 及更高的次要版本	MySQL 8.4.3 及更高版本
SET_ANY_DEFINER	不可用	不可用	允许
SET_USER_ID	已允许	允许	不可用
SHOW_ROUTINE	已允许	已允许	允许
SKIP_QUERY_REWRITE	不允许	不允许	不允许
SYSTEM_USER	不允许	不允许	不允许
SYSTEM_VARIABLES_ADMIN	不允许	不允许	不允许
TABLE_ENCRYPTION_ADMIN	不允许	不允许	不允许
TELEMETRY_LOG_ADMIN	允许	不允许	不允许
TP_CONNECTION_ADMIN	不允许	不允许	不允许
TRANSACTION_GTID_TAG	不可用	不可用	不允许
VERSION_TOKEN_ADMIN	不允许	不允许	不允许
XA_RECOVER_ADMIN	已允许	已允许	允许

结束 RDS for MySQL 的会话或查询

您可使用 `rds_kill` 和 `rds_kill_query` 命令结束数据库实例上的用户会话或查询。首先连接到 MySQL 数据库实例，然后发出正确的命令，如下所示。有关更多信息，请参阅“连接到 MySQL 数据库实例”。

```
CALL mysql.rds_kill(thread-ID)
CALL mysql.rds_kill_query(thread-ID)
```

例如，要结束线程 99 上运行的会话，可以键入以下内容：

```
CALL mysql.rds_kill(99);
```

要结束线程 99 上运行的查询，可以键入以下内容：

```
CALL mysql.rds_kill_query(99);
```

跳过 RDS for MySQL 的当前复制错误

如果错误导致只读副本停止响应但不影响数据的完整性，您可以跳过只读副本错误。

> **Note**
>
> 请先验证是否可以安全地跳过错误。在 MySQL 实用工具中，连接到只读副本并运行以下 MySQL 命令。
>
> ```
> SHOW REPLICA STATUS\G
> ```
>
> 有关返回值的信息，请参阅 MySQL 文档。
> 以前的 MySQL 版本使用的是 `SHOW SLAVE STATUS` 而不是 `SHOW REPLICA STATUS`。如果您使用的 MySQL 版本低于 8.0.23，那么请使用 `SHOW SLAVE STATUS`。

您可以通过以下方式跳过只读副本上的错误。

主题

- 设置 slave_skip_errors 参数

调用 mysql.rds_skip_repl_error 过程

Amazon RDS 提供了一个存储过程，您可以调用该过程以跳过只读副本上的错误。首先连接到您的只读副本，然后发出正确的命令，如下所示。有关更多信息，请参阅 连接到 MySQL 数据库实例。

要跳过错误，请发出以下命令。

```
CALL mysql.rds_skip_repl_error;
```

如果您在未遇到复制错误的源数据库实例或只读副本上运行此命令，则此命令不会产生任何影响。

有关更多信息 (如支持 `mysql.rds_skip_repl_error` 的 MySQL 版本)，请参阅 mysql.rds_skip_repl_error。

> **Important**
>
> 如果您尝试调用 `mysql.rds_skip_repl_error` 并遇到以下错误：`ERROR 1305 (42000): PROCEDURE mysql.rds_skip_repl_error does not exist`，请将您的 MySQL 数据库实例升级到最新次要版本或是 mysql.rds_skip_repl_error 中列出的最低次要版本之一。

设置 slave_skip_errors 参数

要跳过一个或多个错误，您可以设置在只读副本上设置 `slave_skip_errors` 静态参数。您可以将此参数设置为跳过一个或多个特定的复制错误代码。目前，您只能为适用于 MySQL 5.7 的 RDS 数据库实例设置此参数。更改此参数的设置后，确保重启数据库实例，以使新设置生效。有关设置此参数的信息，请参阅 MySQL 文档。

我们建议在单独的数据库参数组中设置此参数。您只能将此数据库参数组与需要跳过错误的只读副本关联。遵循此最佳实践可以减少对其他数据库实例和只读副本的潜在影响。

> **Important**
>
> 为此参数设置非默认值可能会导致复制不一致。只有在用尽其他选项来解决问题并且确定对只读副本数据的潜在影响时，才将此参数设置为非默认值。

使用 InnoDB 表空间改善 RDS for MySQL 的崩溃恢复时间

MySQL 中的所有表均由表定义、数据和索引组成。MySQL 存储引擎 InnoDB 将表数据和索引存储在表空间中。InnoDB 创建全局共享的表空间，其中包括数据字典和其他相关元数据，并且它可能包含表数据和索引。InnoDB 还可以为每张表创建单独的表空间和分区。这些单独的表空间存储在后缀名为 .ibd 的文件中，并且每个表空间的头部包含一个唯一识别编号。

`innodb_file_per_table` 提供一个属于 MySQL 参数组的参数，名称为 Amazon RDS。这些参数控制 InnoDB 是否（通过将参数值设置为 0）向共享表空间或（通过将参数值设置为 1）向单个表空间添加新的表数据和索引。Amazon RDS 将 `innodb_file_per_table` 参数的默认值设置为 1，以便删除单独的 InnoDB 表并回收数据库实例的表所占用的存储空间。在大多数使用案例中，推荐将 `innodb_file_per_table` 参数设置为 1。

当表特别多时 (例如，当使用标准 (磁性) 或通用 SSD 存储时，有超过 1000 张表，或当使用预置的 IOPS 存储时，有超过 10000 张表)，应将 `innodb_file_per_table` 参数设置为 0。当将该参数设置为 0 时，不会创建独立表空间，这样可以节省数据库崩溃恢复所需的时间。

MySQL 在崩溃恢复循环期间处理每个元数据文件 (包括表空间)。当有多个表空间时，与 MySQL 用于处理数以千计的表空间文件的时间相比，MySQL 用于处理共享表空间中的元数据信息的时间可以忽略不计。因为表空间编号存储在每个文件的头部，读取所有表空间文件的总时间可能需要长达数小时。例如，在崩溃恢复循环期间，标准存储空间上的一百万个 InnoDB 表空间可能需要 5 到 8 个小时的处理时间。在一些案例中，InnoDB 可能会在一个崩溃恢复循环后确定它需要额外清理，因此，它会开始另一个崩溃恢复循环，这会使恢复时间延长。请记住，崩溃恢复循环也会引起回滚事务、修复损坏页面和除表空间信息处理以外的其他操作。

因为 `innodb_file_per_table` 参数存储在参数组中，所以可通过编辑数据库实例所使用的参数组来更改该参数值，无需重新启动该数据库实例。更改该设置后，例如从 1 (创建独立的表) 改为 0 (使用共享的表空间)，新的 InnoDB 表将添加到共享的表空间，而现有的表继续使用独立的表空间。要将 InnoDB 表移动到共享的表空间，必须使用 `ALTER TABLE` 命令。

将多个表空间迁移到共享的表空间

您可将 InnoDB 表的元数据从其自己的表空间移至共享的表空间，这将按照 `innodb_file_per_table` 参数设置重建表的元数据。首先连接到 MySQL 数据库实例，然后发出正确的命令，如下所示。有关更多信息，请参阅“[连接到 MySQL 数据库实例](#)”。

```
ALTER TABLE table_name ENGINE = InnoDB, ALGORITHM=COPY;
```

例如，以下查询为不在共享表空间中的每个 InnoDB 表返回 `ALTER TABLE` 语句。

对于 MySQL 5.7 数据库实例：

```
SELECT CONCAT('ALTER TABLE `',
REPLACE(LEFT(NAME , INSTR((NAME), '/') - 1), '`', '``'), '`.`',
REPLACE(SUBSTR(NAME FROM INSTR(NAME, '/') + 1), '`', '``'), '` ENGINE=InnoDB,
 ALGORITHM=COPY;') AS Query
FROM INFORMATION_SCHEMA.INNODB_SYS_TABLES
WHERE SPACE <> 0 AND LEFT(NAME, INSTR((NAME), '/') - 1) NOT IN ('mysql','');
```

对于 MySQL 8.4 和 8.0 数据库实例：

```
SELECT CONCAT('ALTER TABLE `',
REPLACE(LEFT(NAME , INSTR((NAME), '/') - 1), '`', '``'), '`.`',
REPLACE(SUBSTR(NAME FROM INSTR(NAME, '/') + 1), '`', '``'), '` ENGINE=InnoDB,
 ALGORITHM=COPY;') AS Query
FROM INFORMATION_SCHEMA.INNODB_TABLES
WHERE SPACE <> 0 AND LEFT(NAME, INSTR((NAME), '/') - 1) NOT IN ('mysql','');
```

重建一个 MySQL 表以便将该表的元数据移动到共享的表空间，这需要额外的临时存储空间以重建该表，因此，该数据库实例必须有可用的存储空间。在重建期间，该表会被锁定，无法访问查询。对于不经常访问的小型表或普通表，这可能不是问题。对于在大量并发环境中经常访问的大型表或普通表，您可以在只读副本上重建表。

您可以创建只读副本，并将表元数据迁移到只读副本上的共享表空间。虽然 ALTER TABLE 语句会阻止访问只读副本，但源数据库实例不会受到影响。尽管只读副本会在表重建过程中滞后，但源数据库实例将继续生成其二进制日志。由于重建需要额外的存储空间，并且重放日志文件可能会变得很大，因此，您应该创建只读副本，并向其分配大于源数据库实例的存储空间。

要创建只读副本和重建 InnoDB 表以便使用共享表空间，请遵循以下步骤：

1. 确保在源数据库实例上启用备份保留，以便启用二进制日志记录。
2. 使用 Amazon Web Services Management Console或 Amazon CLI 创建源数据库实例的只读副本。由于创建只读副本涉及的流程与崩溃恢复一样多，因此，如果存在大量 InnoDB 表空间，创建过程可能会需要一些时间。在只读副本上分配的存储空间应大于源数据库实例上当前使用的空间。
3. 创建只读副本后，使用参数设置 `read_only = 0` 和 `innodb_file_per_table = 0` 创建参数组，然后将该参数组与只读副本关联。
4. 针对您要在副本中迁移的所有表发出以下 SQL 语句。

```
ALTER TABLE name ENGINE = InnoDB
```

5. 在只读副本中完成所有 `ALTER TABLE` 语句后，验证只读副本是否连接至源数据库实例，并且两个实例是否同步。
6. 使用控制台或 CLI 将只读副本提升为实例。确保用于新的独立数据库实例的参数组已将 `innodb_file_per_table` 参数设置为 0。更改新的独立数据库实例的名称，并将所有应用程序指向新的独立数据库实例。

管理 RDS for MySQL 的全局状态历史记录

> **Tip**
>
> 要分析数据库性能，您还可以在 Amazon RDS 上使用性能详情。有关更多信息，请参阅 在 Amazon RDS 上使用性能详情监控数据库负载。

MySQL 保存了很多状态变量，用于提供与操作相关的信息。状态变量值可以帮助您检测数据库实例上的锁定或内存问题。这些状态变量的值会从上次启动数据库实例时开始累积。可以使用 `FLUSH STATUS` 命令将大多数状态变量重置为 0。

Amazon RDS 提供了一套程序，这些程序会随着时间为这些状态变量的值创建快照，并将它们及上次创建快照后所做的任何更改写入一个表中，从而可以随时间监控这些值。此基础设施名为 Global Status History (GoSH)，安装在从版本 5.5.23 开始的所有 MySQL 数据库实例上。默认情况下 GoSH 处于禁用状态。

要启用 GoSH，请首先在数据库参数组中通过将参数 `event_scheduler` 设置为 `ON`，启用事件调度器。对于运行 MySQL 5.7 的 MySQL 数据库实例，还要将参数 `show_compatibility_56` 设置为 `1`。有关创建和修改数据库参数组的信息，请参阅 Amazon RDS 的参数组。有关启用此参数的副作用的信息，请参阅《MySQL 5.7 参考手册》中的 show_compatibility_56。

然后可以使用下表中的程序启用和配置 GoSH。首先连接到 MySQL 数据库实例，然后发出正确的命令，如下所示。有关更多信息，请参阅 连接到 MySQL 数据库实例。对于每个过程，运行以下命令并替换 *`procedure-name`*：

```
CALL procedure-name;
```

下表列出了可用于上一个命令中 *`procedure-name`* 的所有过程。

程序	描述
mysql.rds_enable_gsh_collector	启用 GoSH，按照 rds_set_gsh_collector 指定的间隔创建默认快照。
mysql.rds_set_gsh_collector	指定快照之间的间隔，以分钟为单位。默认值是 5。
mysql.rds_disable_gsh_collector	禁用快照。
mysql.rds_collect_global_status_history	根据需求创建快照。
mysql.rds_enable_gsh_rotation	按照 mysql.rds_global_status_history 指定的间隔，使 mysql.rds_global_status_history_old 表的内容可交替到 rds_set_gsh_rotation 。
mysql.rds_set_gsh_rotation	指定表交替之间的间隔，以天为单位。默认值是 7。
mysql.rds_disable_gsh_rotation	禁用表交替。
mysql.rds_rotate_global_status_history	根据需求将 mysql.rds_global_status_history 表的内容交替到 mysql.rds_global_status_history_old 。

GoSH 运行时，您可以查询 GoSH 要写入的表。例如，要查询 Innodb 缓冲池的命中率，您可以发送以下查询请求：

```
select a.collection_end, a.collection_start, (( a.variable_Delta-b.variable_delta)/
a.variable_delta)*100 as "HitRatio"
    from mysql.rds_global_status_history as a join mysql.rds_global_status_history as b
 on a.collection_end = b.collection_end
    where a. variable_name = 'Innodb_buffer_pool_read_requests' and b.variable_name =
 'Innodb_buffer_pool_reads'
```

在 MySQL 8.4 中配置缓冲池大小和重做日志容量

在 MySQL 8.4 中，Amazon RDS 默认启用 innodb_dedicated_server 参数。使用 innodb_dedicated_server 参数，数据库引擎可计算 innodb_buffer_pool_size 和 innodb_redo_log_capacity 参数。有关如何计算这些参数的信息，请参阅 MySQL 文档中的 [Configuring InnoDB Buffer Pool Size](#) 和 [Redo Log](#)。

启用 innodb_dedicated_server 后，将根据数据库实例类内存计算 innodb_buffer_pool_size 参数。下表显示了检测到的服务器内存和相应的缓冲池大小。

检测到的服务器内存	缓冲池大小
< 1 GB	默认值为 128 MB
1 GB 到 4 GB	######### * 0.5
> 4 GB	######### * 0.75

innodb_redo_log_capacity 参数会随实例类自动扩展到（vCPU 数量 / 2）GB，最大可达 16 GB。较大的实例类具有更大的重做日志容量，这样可以提高写入密集型工作负载的性能和弹性。

从 MySQL 8.0 升级到 MySQL 8.4 之前，请务必增加存储空间，以适应升级完成后可能出现的重做日志大小的增加。有关更多信息，请参阅 [增加数据库实例存储容量](#)。

如果您不希望 innodb_dedicated_server 参数计算 innodb_buffer_pool_size 和 innodb_redo_log_capacity 参数的值，则可以通过在自定义参数组中为这些值设置特定值来覆盖这些值。或者，您可以禁用 innodb_dedicated_server 参数，并在自定义参数组中设置 innodb_buffer_pool_size 和 innodb_redo_log_capacity 参数的值。有关更多信息，请参阅 [原定设置和自定义参数组](#)。

如果您通过将 innodb_dedicated_server 参数设置为 0 来禁用该参数，并且未为 innodb_buffer_pool_size 和 innodb_redo_log_capacity 参数设置值，则 Amazon RDS 会将后两个参数分别设置为 128 MB 和 100 MB。这些默认值会导致在较大的实例类上性能不佳。

MySQL 数据库实例的本地时区

默认情况下，MySQL 数据库实例的时区是协调世界时 (UTC)。您可以将数据库实例的时区设置为您的应用程序的本地时区。

要设置数据库实例的本地时区，请将数据库实例参数组中的 `time_zone` 参数设置为本节后面列出的受支持值之一。在设置参数组的 `time_zone` 参数时，所有使用该参数组的数据库实例和只读副本都会改用新的本地时区。有关如何设置参数组中的参数的信息，请参阅[Amazon RDS 的参数组](#)。

设置本地时区之后，所有新数据库连接都会反映更改。如果在更改本地时区时打开了任何数据库连接，则到关闭连接再打开新连接之后才会看到本地时区更新。

您可以为数据库实例及其一个或多个只读副本设置不同的本地时区。为此，请对数据库实例和副本使用不同的参数组，并将每个参数组中的 `time_zone` 参数设置为不同的本地时区。

如果要跨 Amazon Web Services 区域复制，则源数据库实例和只读副本使用不同的参数组（参数组对于 Amazon Web Services 区域是唯一的）。要对每个实例使用相同的本地时区，您必须在实例和只读副本的参数组中设置 `time_zone` 参数。

从数据库快照还原数据库实例时，本地时区设置为 UTC。还原完成之后，可以将时区更新为本地时区。如果将数据库实例还原到某个时间点，则还原的数据库实例的本地时区是来自还原的数据库实例的参数组的时区设置。

互联网编号分配机构（IANA）每年多次在 https://www.iana.org/time-zones 上发布新的时区。每当 RDS 发布 MySQL 的新次要维护版本时，RDS 都会附带发布时的最新时区数据。当您使用最新的 RDS for MySQL 版本时，您会获得来自 RDS 的最新时区数据。为确保您的数据库实例具有最新的时区数据，我们建议升级到更高的数据库引擎版本。或者，您可以手动修改 MariaDB 数据库实例中的时区表。为此，可以使用 SQL 命令或者在 SQL 客户端中运行 mysql_tzinfo_to_sql 工具。手动更新时区数据后，重启数据库实例以使更改生效。RDS 不会修改或重置正在运行的数据库实例的时区数据。只有在执行数据库引擎版本升级时，才会安装新的时区数据。

您可以将本地时区设置为以下值之一。

区	时区
非洲	非洲/开罗、非洲/卡萨布兰卡、非洲/哈拉雷、非洲/蒙罗维亚、非洲/内罗毕、非洲/的黎波里、非洲/温得和克
美洲	美洲/阿拉瓜尼亚、美洲/亚松森、美洲/波哥大、美洲/布宜诺斯艾利斯、美洲/加拉加斯、美洲/奇瓦瓦、美洲/库亚巴、美洲/丹佛、美洲/福塔莱萨、美洲/危地马

区	时区
	拉、美洲/哈利法克斯、美洲/玛瑙斯、美洲/马塔莫罗斯、美洲/蒙特雷、美洲/蒙得维的亚、美洲/凤凰城、美洲/圣地亚哥、美洲/蒂华纳
亚洲	亚洲/阿曼、亚洲/阿什哈巴德、亚洲/巴格达、亚洲/巴库、亚洲/曼谷、亚洲/贝鲁特、亚洲/加尔各答、亚洲/大马士革、亚洲/达卡、亚洲/伊尔库茨克、亚洲/耶路撒冷、亚洲/喀布尔、亚洲/卡拉奇、亚洲/加德满都、亚洲/克拉斯诺雅茨克、亚洲/马加丹、亚洲/马斯喀特、亚洲/新西伯利亚、亚洲/利雅得、亚洲/首尔、亚洲/上海、亚洲/新加坡、亚洲/台北、亚洲/德黑兰、亚洲/东京、亚洲/乌兰巴托、亚洲/符拉迪沃斯托克、亚洲/雅库茨克、亚洲/埃里温
大西洋	大西洋州/亚速尔群岛
澳大利亚	澳大利亚/阿德莱德、澳大利亚/布里斯班、澳大利亚/达尔文、澳大利亚/霍巴特、澳大利亚/珀斯、澳大利亚/悉尼
巴西	巴西/迪诺罗尼亚、巴西/东部
加拿大	加拿大/纽芬兰、加拿大/萨斯喀彻温省、加拿大/育空地区
欧洲	欧洲/阿姆斯特丹、欧洲/雅典、欧洲/都柏林、欧洲/赫尔辛基、欧洲/伊斯坦布尔、欧洲/加里宁格勒、欧洲/莫斯科、欧洲/巴黎、欧洲/布拉格、欧洲/萨拉热窝
太平洋	太平洋/奥克兰、太平洋/斐济、太平洋/关岛、太平洋/檀香山、太平洋/萨摩亚群岛
美国	美国/阿拉斯加、美国/中部、美国/印第安纳东部、美国/东部、美国/太平洋地区
UTC	UTC

Amazon RDS for MySQL 的已知问题和限制

下面给出了有关使用 Amazon RDS for MySQL 的已知问题和限制。

主题

InnoDB 保留字

`InnoDB` 是 RDS for MySQL 的保留字。您不能将此名称用于 MySQL 数据库。

Amazon RDS for MySQL 存储已满行为

当 MySQL 数据库实例的存储空间已满时，可能存在元数据不一致、字典不匹配和孤立表。为防止出现这些问题，Amazon RDS 自动停止达到 `storage-full` 状态的数据库实例。

在以下情况下，MySQL 数据库实例达到 `storage-full` 状态：

- 数据库实例的存储空间少于 20000 MiB，可用存储空间达到 200 MiB 或更少。
- 数据库实例的存储空间多于 102400 MiB，可用存储空间达到 1024 MiB 或更少。

- 数据库实例的存储空间介于 20000 MiB 到 102400 MiB 之间，可用存储空间不到 1%。

在由于数据库实例达到 `storage-full` 状态而 Amazon RDS 将其自动停止后，您仍然可以对其进行修改。要重新启动数据库实例，请至少完成以下操作之一：

- 修改数据库实例以启用存储自动扩缩。

  有关存储自动扩展的更多信息，请参阅 使用 Amazon RDS 存储自动扩展功能自动管理容量。
- 修改数据库实例以增加其存储容量。

  有关增加存储容量的更多信息，请参阅 增加数据库实例存储容量。

进行这些更改之一后，数据库实例将自动重新启动。有关修改 数据库实例的信息，请参阅修改 Amazon RDS 数据库实例。

不一致的 InnoDB 缓冲池大小

对于 MySQL 5.7，管理 InnoDB 缓冲池的方式当前存在错误。MySQL 5.7 可能将 `innodb_buffer_pool_size` 参数的值调整为较大的值，这会导致 InnoDB 缓冲池增长得过大并占用过多内存。此效果会导致 MySQL 数据库引擎停止运行或阻止该引擎启动。可用内存较少的数据库实例类更易出现此问题。

要解决此问题，请将 `innodb_buffer_pool_size` 参数的值设置为 `innodb_buffer_pool_instances` 参数值和 `innodb_buffer_pool_chunk_size` 参数值的积的倍数。例如，您可以将 `innodb_buffer_pool_size` 参数值设置为 `innodb_buffer_pool_instances` 参数值和 `innodb_buffer_pool_chunk_size` 参数值的积的 8 倍，如以下示例所示。

```
innodb_buffer_pool_chunk_size = 536870912
innodb_buffer_pool_instances = 4
innodb_buffer_pool_size = (536870912 * 4) * 8 = 17179869184
```

有关此 MySQL 5.7 错误的详细信息，请参阅 MySQL 文档中的 https://bugs.mysql.com/bug.php?id=79379。

索引合并优化返回不正确的结果

使用索引合并优化功能的查询可能因 MySQL 5.5.37 中引入的 MySQL 查询优化程序中的错误而返回不正确的结果。当您针对一个带多条索引的表发出查询时，此优化器将基于多条索引扫描一系列行，但

无法正确地合并这些结果。有关查询优化程序错误的详细信息，请参阅 MySQL 错误数据库中的 http://bugs.mysql.com/bug.php?id=72745 和 http://bugs.mysql.com/bug.php?id=68194。

例如，假设发出针对一个带两条索引的表的查询，其中搜索参数会引用索引列。

```
SELECT * FROM table1
WHERE indexed_col1 = 'value1' AND indexed_col2 = 'value2';
```

在此情况下，搜索引擎将搜索这两条索引。由于出现错误，导致合并结果不正确。

要解决此问题，您可以执行下列操作之一：

- 在 MySQL 数据库实例的数据库参数组中将 `optimizer_switch` 参数设置为 `index_merge=off`。有关设置数据库参数组参数的信息，请参阅Amazon RDS 的参数组。
- 将 MySQL 数据库实例升级到 MySQL 5.7 或 8.0 版。有关更多信息，请参阅 升级 RDS for MySQL 数据库引擎。
- 如果无法升级实例或更改 `optimizer_switch` 参数，您可以通过显式标识查询的索引来纠正错误，例如：

```
SELECT * FROM table1
USE INDEX covering_index
WHERE indexed_col1 = 'value1' AND indexed_col2 = 'value2';
```

有关更多信息，请参阅 MySQL 文档中的索引合并优化。

Amazon RDS 数据库实例的 MySQL 参数异常错误

某些 MySQL 参数在与 Amazon RDS 数据库实例一起使用时需要考虑一些特殊的注意事项。

lower_case_table_names

因为 Amazon RDS 使用区分大小写的文件系统，所以不支持将 `lower_case_table_names` 服务器参数的值设置为 2（名称存储为给定名称，但按小写字母进行比较）。以下是 Amazon RDS for MySQL 数据库实例支持的值：

- 所有 RDS for MySQL 版本均支持 0（名称存储为给定名称，而比较区分大小写）。
- RDS for MySQL 5.7、8.0.28 及更高的 8.0 和 8.4 版本支持 1（以小写字母形式存储名称，并且比较不区分大小写）。

创建数据库实例前，请在自定义数据库参数组中设置 `lower_case_table_names` 参数。然后，您可以在创建数据库实例时指定自定义数据库参数组。

当参数组与版本低于 8.0 的 MySQL 数据库实例相关联时，建议您避免在参数组中修改 `lower_case_table_names` 参数。更改它可能导致时间点恢复备份与只读副本数据库实例不一致。

当参数组与 8.0 或 8.4 版 MySQL 数据库实例相关联时，您将无法在参数组中修改 `lower_case_table_names` 参数。

只读副本应总是使用与源数据库实例相同的 `lower_case_table_names` 参数值。

long_query_time

您可以将 `long_query_time` 参数设置为浮点值，这样，您可以将慢速查询以微秒分辨率记录到 MySQL 慢速查询日志中。您可以设置一个值（如 0.1 秒，也就是 100 毫秒），在调试时间不到一秒的慢事务时提供帮助。

Amazon RDS 中的 MySQL 文件大小限制

对于 MySQL 8.0 及更高版本的数据库实例，最大文件大小为 16 TiB。使用 file-per-table 表空间时，最大文件大小将 InnoDB 表的大小限制为 16 TiB。默认情况下，为 MySQL 数据库实例设置 InnoDB file-per-table 表空间 (表各自位于自己的表空间中)。有关更多信息，请参阅 MySQL 文档中的 InnoDB limits。

> **Note**
>
> 某些现有的数据库实例具有下限。例如，在 2014 年 4 月之前创建的 MySQL 数据库实例的文件和表大小限制为 2 TB。不管创建数据库实例的时间如何，此 2 TB 文件大小限制也适用于从 2014 年 4 月之前创建的数据库快照创建的数据库实例或只读副本。

根据您的应用程序，使用 InnoDB file-per-table 表空间有一些优缺点。要确定适用于您的应用程序的最佳方法，请参阅 MySQL 文档中的 File-Per-Table 表空间。

建议不要允许表增长到最大文件大小。一般而言，更好的方法是将数据分区到较小的表中，这可以提高性能并缩短恢复时间。

可用于将大型表划分为多个较小表的一个选项是分区。分区 可基于您指定的规则将大型表的各个部分分布到单独的文件中。例如，如果您按日期存储事务，则可以创建分区规则，以便使用分区将较旧的事

务分布到单独的文件中。随后您可以定期存档无需随时可供应用程序使用的历史事务数据。有关更多信息，请参阅 MySQL 文档中的分区。

由于没有单个系统表或视图可以提供所有表和 InnoDB 系统表空间的大小，因此必须查询多个表才能确定表空间的大小。

确定 InnoDB 系统表空间和数据字典表空间的大小

- 使用以下 SQL 命令可确定任何表空间是否过大以及是否适合进行分区。

> **Note**
>
> 数据字典表空间特定于 MySQL 8.0 及更高版本。

```
select FILE_NAME,TABLESPACE_NAME, ROUND(((TOTAL_EXTENTS*EXTENT_SIZE)
/1024/1024/1024), 2)  as "File Size (GB)" from information_schema.FILES
where tablespace_name in ('mysql','innodb_system');
```

确定 InnoDB 系统表空间之外的 InnoDB 用户表的大小（针对 MySQL 5.7 版本）

- 使用以下 SQL 命令可确定任何表是否太大以及是否适合进行分区。

```
SELECT SPACE,NAME,ROUND((ALLOCATED_SIZE/1024/1024/1024), 2)
as "Tablespace Size (GB)"
FROM information_schema.INNODB_SYS_TABLESPACES ORDER BY 3 DESC;
```

确定 InnoDB 系统表空间之外的 InnoDB 用户表的大小（对于 MySQL 8.0 及更高版本）

- 使用以下 SQL 命令可确定任何表是否太大以及是否适合进行分区。

```
SELECT SPACE,NAME,ROUND((ALLOCATED_SIZE/1024/1024/1024), 2)
as "Tablespace Size (GB)"
FROM information_schema.INNODB_TABLESPACES ORDER BY 3 DESC;
```

确定非 InnoDB 用户表的大小

- 使用以下 SQL 命令来确定是否有任何非 InnoDB 用户表过大。

```
SELECT TABLE_SCHEMA, TABLE_NAME, round(((DATA_LENGTH + INDEX_LENGTH+DATA_FREE)
/ 1024 / 1024/ 1024), 2) As "Approximate size (GB)" FROM information_schema.TABLES
WHERE TABLE_SCHEMA NOT IN ('mysql', 'information_schema', 'performance_schema')
and ENGINE<>'InnoDB';
```

启用 InnoDB file-per-table 表空间

- 请在数据库实例的参数组中将 innodb_file_per_table 参数设置为 `1`。

禁用 InnoDB file-per-table 表空间

- 请在数据库实例的参数组中将 innodb_file_per_table 参数设置为 `0`。

有关更新参数组的信息，请参阅Amazon RDS 的参数组。

启用或禁用 InnoDB file-per-table 表空间后，可以发出 `ALTER TABLE` 命令将一个表从全局表空间移至它自己的表空间，或从它自己的表空间移至全局表空间，如以下示例所示：

```
ALTER TABLE table_name TABLESPACE=innodb_file_per_table;
```

不支持 MySQL 密钥环插件

目前，Amazon RDS for MySQL 不支持 MySQL `keyring_aws` Amazon Web Services 密钥环插件。

自定义端口

Amazon RDS 禁止连接到 MySQL 引擎的自定义端口 33060。为您的 MySQL 引擎选择不同的端口。

MySQL 存储过程限制

在以下 RDS for MySQL 版本上，mysql.rds_kill 和 mysql.rds_kill_query 存储过程无法终止由用户名多于 16 个字符的 MySQL 用户所拥有的会话或查询：

- 8.0.32 及更低的 8 版本
- 5.7.41 及更低的 5.7 版本

与外部源实例之间的基于 GTID 的复制

Amazon RDS 支持基于全局事务标识符（GTID），从外部 MySQL 实例复制到需要在配置过程中设置 GTID_PURGED 的 Amazon RDS for MySQL 数据库实例。但是，只有 RDS for MySQL 8.0.37 及更高版本才支持此功能。

MySQL 默认身份验证插件

RDS for MySQL 8.0.34 及更高的 8.0 版本使用 `mysql_native_password` 插件。您将无法更改 `default_authentication_plugin` 设置。

RDS for MySQL 8.4 及更高版本使用 `caching_sha2_password` 插件作为默认身份验证插件。可以更改 MySQL 8.4 的默认身份验证插件。`mysql_native_password` 插件仍然适用于 MySQL 8.4，但对该插件的支持将在 MySQL 8.4 中终止。要更改默认身份验证插件，请创建自定义参数组并修改 `authentication_policy` 参数的值。有关更多信息，请参阅 the section called “原定设置和自定义参数组”。

覆盖 innodb_buffer_pool_size

对于微型或小型数据库实例类，`innodb_buffer_pool_size` 参数的默认值可能与运行以下命令时返回的值不同：

```
mysql> SELECT @@innodb_buffer_pool_size;
```

当 Amazon RDS 在管理数据库实例类时需要覆盖默认值时，可能会出现这种差异。如有必要，您可以覆盖默认值并将其设置为数据库实例类支持的值。要确定有效值，请将内存使用量和数据库实例上的可用内存总量相加。有关更多信息，请参阅 Amazon RDS 实例类型。

如果您的数据库实例只有 4 GB 的内存，则无法将 `innodb_buffer_pool_size` 设置为 8 GB，但可以将其设置为 3 GB，具体取决于您为其他参数分配的内存量。

如果您输入的值太大，Amazon RDS 会将该值降低到以下限制范围内：

- 微型数据库实例类：256 MB
- db.t4g.micro 数据库实例类：128 MB

从 MySQL 5.7 升级到 MySQL 8.4

您无法从 MySQL 5.7 直接升级到 MySQL 8.4。必须首先从 MySQL 5.7 升级到 MySQL 8.0，然后从 MySQL 8.0 升级到 MySQL 8.4。有关更多信息，请参阅 RDS for MySQL 的主要版本升级。

InnoDB 页面压缩

InnoDB 页面压缩不适用于文件系统块大小为 16k 的 Amazon RDS 数据库实例，因为文件系统块大小必须小于 InnoDB 页面大小。从 2024 年 2 月开始，所有新创建的数据库实例的文件系统块大小均为 16k，这会增加吞吐量并减少页面刷新期间的 IOPS 消耗。

RDS for MySQL 存储过程参考

这些主题介绍对于运行 MySQL 数据库引擎的 Amazon RDS 实例可用的系统存储过程。主用户必须运行这些过程。

主题

收集和维护全局状态历史记录

Amazon RDS 提供了一组过程，这些过程可以随时间推移对状态变量的值创建快照，并将它们及上次创建快照后所做的任何更改写入一个表中。该基础设施称为全局状态历史记录。有关更多信息，请参阅管理 RDS for MySQL 的全局状态历史记录。

以下存储过程管理全局状态历史记录的收集和维护方式。

主题

- mysql.rds_collect_global_status_history
- mysql.rds_disable_gsh_collector
- mysql.rds_disable_gsh_rotation
- mysql.rds_enable_gsh_collector
- mysql.rds_enable_gsh_rotation
- mysql.rds_rotate_global_status_history
- mysql.rds_set_gsh_collector
- mysql.rds_set_gsh_rotation

mysql.rds_collect_global_status_history

按需为全局状态历史记录创建快照。

语法

```
CALL mysql.rds_collect_global_status_history;
```

mysql.rds_disable_gsh_collector

关闭全局状态历史记录创建的快照。

语法

```
CALL mysql.rds_disable_gsh_collector;
```

mysql.rds_disable_gsh_rotation

关闭 `mysql.global_status_history` 表的交替。

语法

```
CALL mysql.rds_disable_gsh_rotation;
```

mysql.rds_enable_gsh_collector

开启全局状态历史记录，以 `rds_set_gsh_collector` 指定的间隔创建原定设置快照。

语法

```
CALL mysql.rds_enable_gsh_collector;
```

mysql.rds_enable_gsh_rotation

按照 `rds_set_gsh_rotation` 指定的间隔，开启将 `mysql.global_status_history` 表的内容交替到 `mysql.global_status_history_old`

语法

```
CALL mysql.rds_enable_gsh_rotation;
```

mysql.rds_rotate_global_status_history

根据需求将 `mysql.global_status_history` 表的内容交替到 `mysql.global_status_history_old`。

语法

```
CALL mysql.rds_rotate_global_status_history;
```

mysql.rds_set_gsh_collector

指定全局状态历史记录创建的快照之间的间隔，以分钟为单位。

语法

```
CALL mysql.rds_set_gsh_collector(intervalPeriod);
```

参数

intervalPeriod

快照之间的间隔，以分钟为单位。默认值为 5。

mysql.rds_set_gsh_rotation

指定 `mysql.global_status_history` 表轮换之间的间隔，以天为单位。

语法

```
CALL mysql.rds_set_gsh_rotation(intervalPeriod);
```

参数

intervalPeriod

表轮换之间的间隔，以天为单位。默认值为 7。

配置、启动和停止二进制日志（binlog）复制

以下存储过程控制事务如何从外部数据库复制到 RDS for MySQL，或从 RDS for MySQL 复制到外部数据库。

在使用这些存储过程管理复制时，如果其复制用户配置了 `caching_sha2_password`，则必须通过指定 `SOURCE_SSL=1` 来配置 TLS。`caching_sha2_password` 是 RDS for MySQL 8.4 的默认身份验证插件。有关更多信息，请参阅使用 SSL/TLS 进行加密。

有关配置、使用和管理只读副本的信息，请参阅 the section called “MySQL 只读副本”。

主题

mysql.rds_next_master_log（RDS for MariaDB 和 RDS for MySQL 主要版本 8.0 及更低版本）

将源数据库实例日志位置更改为源数据库实例上的下一个二进制日志的开始位置。只有在只读副本上收到复制 I/O 错误 1236 时，才能使用该过程。

语法

```
CALL mysql.rds_next_master_log(
curr_master_log
);
```

参数

curr_master_log

当前主日志文件的索引。例如，如果当前文件名为 mysql-bin-changelog.012345，则索引为 12345。若要确定当前主日志文件名，请运行 SHOW REPLICA STATUS 命令并查看 Master_Log_File 字段。

使用说明

主用户必须运行 mysql.rds_next_master_log 过程。

> **Warning**
>
> 仅在对作为复制源的多可用区数据库实例进行故障转移后复制失败，并且 mysql.rds_next_master_log 的 Last_IO_Errno 字段报告 I/O 错误 1236 时调用 SHOW REPLICA STATUS。
> 如果在发生失效转移事件之前，源实例中的事务未写入到磁盘上的二进制日志，调用 mysql.rds_next_master_log 可能会导致只读副本丢失数据。可以通过将源实例参数 sync_binlog 和 innodb_support_xa 设置为 1 来减少发生这种情况的几率，尽管这可能会降低性能。有关更多信息，请参阅排查 MySQL 只读副本问题。

示例

假设复制在 RDS for MySQL 只读副本上失败。对只读副本运行 SHOW REPLICA STATUS\G 会返回以下结果：

```
*************************** 1. row ***************************
             Replica_IO_State:
                  Source_Host: myhost.XXXXXXXXXXXXXXX.rr-rrrr-1.rds.amazonaws.com
                  Source_User: MasterUser
                  Source_Port: 3306
                Connect_Retry: 10
              Source_Log_File: mysql-bin-changelog.012345
          Read_Source_Log_Pos: 1219393
               Relay_Log_File: relaylog.012340
                Relay_Log_Pos: 30223388
        Relay_Source_Log_File: mysql-bin-changelog.012345
           Replica_IO_Running: No
          Replica_SQL_Running: Yes
              Replicate_Do_DB:
          Replicate_Ignore_DB:
           Replicate_Do_Table:
       Replicate_Ignore_Table:
      Replicate_Wild_Do_Table:
  Replicate_Wild_Ignore_Table:
                   Last_Errno: 0
                   Last_Error:
                 Skip_Counter: 0
          Exec_Source_Log_Pos: 30223232
              Relay_Log_Space: 5248928866
              Until_Condition: None
               Until_Log_File:
                Until_Log_Pos: 0
           Source_SSL_Allowed: No
           Source_SSL_CA_File:
           Source_SSL_CA_Path:
              Source_SSL_Cert:
            Source_SSL_Cipher:
               Source_SSL_Key:
        Seconds_Behind_Master: NULL
Source_SSL_Verify_Server_Cert: No
                Last_IO_Errno: 1236
                Last_IO_Error: Got fatal error 1236 from master when reading data from
 binary log: 'Client requested master to start replication from impossible position;
 the first event 'mysql-bin-changelog.013406' at 1219393, the last event read from
 '/rdsdbdata/log/binlog/mysql-bin-changelog.012345' at 4, the last byte read from '/
rdsdbdata/log/binlog/mysql-bin-changelog.012345' at 4.'
               Last_SQL_Errno: 0
               Last_SQL_Error:
```

```
  Replicate_Ignore_Server_Ids:
               Source_Server_Id: 67285976
```

Last_IO_Errno 字段显示该实例收到 I/O 错误 1236。Master_Log_File 字段显示文件名为 mysql-bin-changelog.012345，这意味着日志文件索引为 12345。要纠正该错误，您可以使用以下参数调用 mysql.rds_next_master_log：

```
CALL mysql.rds_next_master_log(12345);
```

mysql.rds_next_source_log（RDS for MySQL 主要版本 8.4 及更高版本）

将源数据库实例日志位置更改为源数据库实例上的下一个二进制日志的开始位置。只有在只读副本上收到复制 I/O 错误 1236 时，才能使用该过程。

语法

```
CALL mysql.rds_next_source_log(
curr_source_log
);
```

参数

curr_source_log

当前源日志文件的索引。例如，如果当前文件名为 mysql-bin-changelog.012345，则索引为 12345。要确定当前源日志文件名，请运行 SHOW REPLICA STATUS 命令并查看 Source_Log_File 字段。

使用说明

管理用户必须运行 mysql.rds_next_source_log 过程。

> **Warning**
>
> 仅在对作为复制源的多可用区数据库实例进行故障转移后复制失败，并且 mysql.rds_next_source_log 的 Last_IO_Errno 字段报告 I/O 错误 1236 时调用 SHOW REPLICA STATUS。
> 如果在发生失效转移事件之前，源实例中的事务未写入到磁盘上的二进制日志，调用 mysql.rds_next_source_log 可能会导致只读副本丢失数据。可以通过将源实例参数

sync_binlog 和 innodb_support_xa 设置为 1 来减少发生这种情况的几率，尽管这可能会降低性能。有关更多信息，请参阅排查 MySQL 只读副本问题。

示例

假设复制在 RDS for MySQL 只读副本上失败。对只读副本运行 SHOW REPLICA STATUS\G 会返回以下结果：

```
*************************** 1. row ***************************
             Replica_IO_State:
                  Source_Host: myhost.XXXXXXXXXXXXXXX.rr-rrrr-1.rds.amazonaws.com
                  Source_User: MasterUser
                  Source_Port: 3306
                Connect_Retry: 10
              Source_Log_File: mysql-bin-changelog.012345
          Read_Source_Log_Pos: 1219393
               Relay_Log_File: relaylog.012340
                Relay_Log_Pos: 30223388
        Relay_Source_Log_File: mysql-bin-changelog.012345
           Replica_IO_Running: No
          Replica_SQL_Running: Yes
              Replicate_Do_DB:
          Replicate_Ignore_DB:
           Replicate_Do_Table:
       Replicate_Ignore_Table:
      Replicate_Wild_Do_Table:
  Replicate_Wild_Ignore_Table:
                   Last_Errno: 0
                   Last_Error:
                 Skip_Counter: 0
          Exec_Source_Log_Pos: 30223232
              Relay_Log_Space: 5248928866
              Until_Condition: None
               Until_Log_File:
                Until_Log_Pos: 0
           Source_SSL_Allowed: No
           Source_SSL_CA_File:
           Source_SSL_CA_Path:
              Source_SSL_Cert:
            Source_SSL_Cipher:
               Source_SSL_Key:
        Seconds_Behind_Source: NULL
```

```
Source_SSL_Verify_Server_Cert: No
                Last_IO_Errno: 1236
                Last_IO_Error: Got fatal error 1236 from source when reading data from
 binary log: 'Client requested source to start replication from impossible position;
 the first event 'mysql-bin-changelog.013406' at 1219393, the last event read from
 '/rdsdbdata/log/binlog/mysql-bin-changelog.012345' at 4, the last byte read from '/
rdsdbdata/log/binlog/mysql-bin-changelog.012345' at 4.'
               Last_SQL_Errno: 0
               Last_SQL_Error:
  Replicate_Ignore_Server_Ids:
             Source_Server_Id: 67285976
```

`Last_IO_Errno` 字段显示该实例收到 I/O 错误 1236。`Source_Log_File` 字段显示文件名为 `mysql-bin-changelog.012345`，这意味着日志文件索引为 `12345`。要纠正该错误，您可以使用以下参数调用 `mysql.rds_next_source_log`：

```
CALL mysql.rds_next_source_log(12345);
```

mysql.rds_reset_external_master（RDS for MariaDB 和 RDS for MySQL 主要版本 8.0 及更低版本）

重新配置 RDS for MySQL 数据库实例，使其不再是在 Amazon RDS 之外运行的某个 MySQL 实例的只读副本。

> **Important**
>
> 要运行此过程，必须启用 `autocommit`。要启用它，请将 `autocommit` 参数设置为 `1`。有关修改参数的信息，请参阅 [在 Amazon RDS 中修改数据库参数组中的参数]。

语法

```
CALL mysql.rds_reset_external_master;
```

使用说明

主用户必须运行 `mysql.rds_reset_external_master` 过程。此过程必须运行于一个 MySQL 数据库实例上，后者要作为在 Amazon RDS 之外运行的 MySQL 实例的只读副本而被删除。

> **Note**
>
> 建议您尽可能使用只读副本管理两个 Amazon RDS 数据库实例之间的复制。在执行该操作时，建议您仅使用该存储过程以及其他与复制相关的存储过程。这些做法可以在 Amazon RDS 数据库实例之间实现更复杂的复制拓扑。我们提供这些存储过程主要是为了与在 Amazon RDS 外部运行的 MySQL 实例之间启用复制。有关管理 Amazon RDS 数据库实例之间的复制的信息，请参阅使用数据库实例只读副本。

想要了解更多有关使用复制从在 Amazon RDS 之外运行的 MySQL 实例导入数据的信息，请参阅 配置与外部源实例之间的二进制日志文件位置复制。

mysql.rds_reset_external_source（RDS for MySQL 主要版本 8.4 及更高版本）

重新配置 RDS for MySQL 数据库实例，使其不再是在 Amazon RDS 之外运行的某个 MySQL 实例的只读副本。

> **Important**
>
> 要运行此过程，必须启用 `autocommit`。要启用它，请将 `autocommit` 参数设置为 `1`。有关修改参数的信息，请参阅 在 Amazon RDS 中修改数据库参数组中的参数。

语法

```
CALL mysql.rds_reset_external_source;
```

使用说明

管理用户必须运行 `mysql.rds_reset_external_source` 过程。此过程必须运行于一个 MySQL 数据库实例上，后者要作为在 Amazon RDS 之外运行的 MySQL 实例的只读副本而被删除。

> **Note**
>
> 建议您尽可能使用只读副本管理两个 Amazon RDS 数据库实例之间的复制。在执行该操作时，建议您仅使用该存储过程以及其他与复制相关的存储过程。这些做法可以在 Amazon RDS 数据库实例之间实现更复杂的复制拓扑。我们提供这些存储过程主要是为了与在 Amazon RDS 外部运行的 MySQL 实例之间启用复制。

有关管理 Amazon RDS 数据库实例之间的复制的信息，请参阅使用数据库实例只读副本。想要了解更多有关使用复制从在 Amazon RDS 之外运行的 MySQL 实例导入数据的信息，请参阅配置与外部源实例之间的二进制日志文件位置复制。

mysql.rds_set_external_master（RDS for MariaDB 和 RDS for MySQL 主要版本 8.0 及更低版本）

将 RDS for MySQL 数据库实例配置为在 Amazon RDS 之外运行的 MySQL 实例的只读副本。

> **Important**
>
> 要运行此过程，必须启用 `autocommit`。要启用它，请将 `autocommit` 参数设置为 1。有关修改参数的信息，请参阅 在 Amazon RDS 中修改数据库参数组中的参数。

> **Note**
>
> 您可以使用 mysql.rds_set_external_master_with_delay（RDS for MariaDB 和 RDS for MySQL 主要版本 8.0 及更低版本） 存储过程配置外部源数据库实例和延迟复制。

语法

```
CALL mysql.rds_set_external_master (
  host_name
  , host_port
  , replication_user_name
  , replication_user_password
  , mysql_binary_log_file_name
  , mysql_binary_log_file_location
  , ssl_encryption
);
```

参数

host_name

在 Amazon RDS 之外运行以变为源数据库实例的 MySQL 实例的主机名或 IP 地址。

host_port

在 Amazon RDS 之外运行的要配置为源数据库实例的 MySQL 实例使用的端口。如果网络配置包括转换端口号的安全 Shell (SSH) 端口复制，请指定由 SSH 公开的端口号。

replication_user_name

对在 Amazon RDS 外部运行的 MySQL 实例具有 `REPLICATION CLIENT` 和 `REPLICATION SLAVE` 权限的用户的 ID。建议您向专用于复制的账户提供外部实例。

replication_user_password

在 `replication_user_name` 中指定的用户 ID 的密码。

mysql_binary_log_file_name

源数据库实例上包含复制信息的二进制日志的名称。

mysql_binary_log_file_location

`mysql_binary_log_file_name` 二进制日志中复制将开始读取复制信息的位置。

您可以通过在源数据库实例上运行 `SHOW MASTER STATUS` 来确定二进制日志文件名和位置。

ssl_encryption

指定是否在复制连接中使用安全套接字层（SSL）加密的值。1 表示使用 SSL 加密，0 表示不使用加密。默认值为 0。

> **Note**
>
> 不支持 `MASTER_SSL_VERIFY_SERVER_CERT` 选项。此选项设置为 0，这意味着连接已加密，但未验证证书。

使用说明

主用户必须运行 `mysql.rds_set_external_master` 过程。必须在要配置为在 Amazon RDS 外部运行的 MySQL 实例的只读副本的 MySQL 数据库实例上运行该过程。

运行 `mysql.rds_set_external_master` 之前，您必须先将在 Amazon RDS 之外运行的 MySQL 实例配置为源数据库实例。要连接到在 Amazon RDS 之外运行的 MySQL 实例，您必须指定 `replication_user_name` 和 `replication_user_password` 值，这些值指示对 MySQL 的外部实例具有 `REPLICATION CLIENT` 和 `REPLICATION SLAVE` 权限的复制用户。

将 MySQL 的外部实例配置为源数据库实例

1. 通过使用所选的 MySQL 客户端，连接到 MySQL 的外部实例并创建要用于复制的用户账户。示例如下：

   MySQL 5.7

   ```
   CREATE USER 'repl_user'@'mydomain.com' IDENTIFIED BY 'password';
   ```

   MySQL 8.0

   ```
   CREATE USER 'repl_user'@'mydomain.com' IDENTIFIED WITH mysql_native_password BY
    'password';
   ```

   > **Note**
   >
   > 作为安全最佳实践，请指定除此处所示提示以外的密码。

2. 对于 MySQL 的外部实例，向复制用户授予 `REPLICATION CLIENT` 和 `REPLICATION SLAVE` 权限。以下示例向您所在域的“repl_user”用户授予对所有数据库的 `REPLICATION CLIENT` 和 `REPLICATION SLAVE` 权限。

   MySQL 5.7

   ```
   GRANT REPLICATION CLIENT, REPLICATION SLAVE ON *.* TO 'repl_user'@'mydomain.com'
    IDENTIFIED BY 'password';
   ```

   MySQL 8.0

   ```
   GRANT REPLICATION CLIENT, REPLICATION SLAVE ON *.* TO 'repl_user'@'mydomain.com';
   ```

要使用加密复制，请将源数据库实例配置为使用 SSL 连接。

> **Note**
>
> 建议您尽可能使用只读副本管理两个 Amazon RDS 数据库实例之间的复制。在执行该操作时，建议您仅使用该存储过程以及其他与复制相关的存储过程。这些做法可以在 Amazon RDS 数据库实例之间实现更复杂的复制拓扑。我们提供这些存储过程主要是为了与在 Amazon RDS

外部运行的 MySQL 实例之间启用复制。有关管理 Amazon RDS 数据库实例之间的复制的信息，请参阅使用数据库实例只读副本。

在调用 `mysql.rds_set_external_master` 将 Amazon RDS 数据库实例配置为只读副本后，可对该只读副本调用 mysql.rds_start_replication 开始复制过程。您可以调用 mysql.rds_reset_external_master（RDS for MariaDB 和 RDS for MySQL 主要版本 8.0 及更低版本）删除只读副本配置。

调用 `mysql.rds_set_external_master` 时，Amazon RDS 将时间、用户和 `set master` 的操作记录在 `mysql.rds_history` 和 `mysql.rds_replication_status` 表中。

示例

在 MySQL 数据库实例上运行时，下例将该数据库实例配置为在 Amazon RDS 之外运行的某个 MySQL 实例的只读副本。

```
call mysql.rds_set_external_master(
  'Externaldb.some.com',
  3306,
  'repl_user',
  'password',
  'mysql-bin-changelog.0777',
  120,
  1);
```

mysql.rds_set_external_source（RDS for MySQL 主要版本 8.4 及更高版本）

将 RDS for MySQL 数据库实例配置为在 Amazon RDS 之外运行的 MySQL 实例的只读副本。

Important

要运行此过程，必须启用 `autocommit`。要启用它，请将 `autocommit` 参数设置为 `1`。有关修改参数的信息，请参阅 在 Amazon RDS 中修改数据库参数组中的参数。

语法

```
CALL mysql.rds_set_external_source (
```

```
  host_name
  , host_port
  , replication_user_name
  , replication_user_password
  , mysql_binary_log_file_name
  , mysql_binary_log_file_location
  , ssl_encryption
);
```

参数

host_name

在 Amazon RDS 之外运行以变为源数据库实例的 MySQL 实例的主机名或 IP 地址。

host_port

在 Amazon RDS 之外运行的要配置为源数据库实例的 MySQL 实例使用的端口。如果网络配置包括转换端口号的安全 Shell (SSH) 端口复制，请指定由 SSH 公开的端口号。

replication_user_name

对在 Amazon RDS 外部运行的 MySQL 实例具有 `REPLICATION CLIENT` 和 `REPLICATION SLAVE` 权限的用户的 ID。建议您向专用于复制的账户提供外部实例。

replication_user_password

在 `replication_user_name` 中指定的用户 ID 的密码。

mysql_binary_log_file_name

源数据库实例上包含复制信息的二进制日志的名称。

mysql_binary_log_file_location

`mysql_binary_log_file_name` 二进制日志中复制将开始读取复制信息的位置。

您可以通过在源数据库实例上运行 `SHOW MASTER STATUS` 来确定二进制日志文件名和位置。

ssl_encryption

指定是否在复制连接中使用安全套接字层（SSL）加密的值。1 表示使用 SSL 加密，0 表示不使用加密。默认值是 0。

> **Note**
>
> 不支持 SOURCE_SSL_VERIFY_SERVER_CERT 选项。此选项设置为 0，这意味着连接已加密，但未验证证书。

使用说明

管理用户必须运行 mysql.rds_set_external_source 过程。该过程必须在以下 RDS for MySQL 数据库实例上运行：也即，该实例将配置为在 Amazon RDS 外部运行的 MySQL 实例的只读副本。

运行 mysql.rds_set_external_source 之前，必须先将在 Amazon RDS 之外运行的 MySQL 实例配置为源数据库实例。要连接到在 Amazon RDS 之外运行的 MySQL 实例，您必须指定 replication_user_name 和 replication_user_password 值，这些值指示对 MySQL 的外部实例具有 REPLICATION CLIENT 和 REPLICATION SLAVE 权限的复制用户。

将 MySQL 的外部实例配置为源数据库实例

1. 通过使用所选的 MySQL 客户端，连接到 MySQL 的外部实例并创建要用于复制的用户账户。示例如下：

   ```
   CREATE USER 'repl_user'@'mydomain.com' IDENTIFIED BY 'password';
   ```

   > **Note**
   >
   > 作为安全最佳实践，请指定除此处所示提示以外的密码。

2. 对于 MySQL 的外部实例，向复制用户授予 REPLICATION CLIENT 和 REPLICATION SLAVE 权限。以下示例向您所在域的“repl_user”用户授予对所有数据库的 REPLICATION CLIENT 和 REPLICATION SLAVE 权限。

   ```
   GRANT REPLICATION CLIENT, REPLICATION SLAVE ON *.* TO 'repl_user'@'mydomain.com';
   ```

要使用加密复制，请将源数据库实例配置为使用 SSL 连接。此外，使用 [mysql.rds_import_binlog_ssl_material](#) 过程，将证书颁发机构证书、客户端证书和客户端密钥导入到数据库实例或数据库集群中。

> **Note**
> 建议您尽可能使用只读副本管理两个 Amazon RDS 数据库实例之间的复制。在执行该操作时，建议您仅使用该存储过程以及其他与复制相关的存储过程。这些做法可以在 Amazon RDS 数据库实例之间实现更复杂的复制拓扑。我们提供这些存储过程主要是为了与在 Amazon RDS 外部运行的 MySQL 实例之间启用复制。有关管理 Amazon RDS 数据库实例之间的复制的信息，请参阅使用数据库实例只读副本。

在调用 `mysql.rds_set_external_source` 将 RDS for MySQL 数据库实例配置为只读副本后，可对该只读副本调用 mysql.rds_start_replication 以开始复制过程。您可以调用 mysql.rds_reset_external_source（RDS for MySQL 主要版本 8.4 及更高版本）来删除只读副本配置。

调用 `mysql.rds_set_external_source` 时，Amazon RDS 将时间、用户和 `set master` 的操作记录在 `mysql.rds_history` 和 `mysql.rds_replication_status` 表中。

示例

在 RDS for MySQL 数据库实例上运行时，以下示例将该数据库实例配置为在 Amazon RDS 外部运行的某个 MySQL 实例的只读副本。

```
call mysql.rds_set_external_source(
  'Externaldb.some.com',
  3306,
  'repl_user',
  'password',
  'mysql-bin-changelog.0777',
  120,
  1);
```

mysql.rds_set_external_master_with_auto_position（RDS for MySQL 主要版本 8.0 及更低版本）

将 RDS for MySQL 数据库实例配置为在 Amazon RDS 之外运行的 MySQL 实例的只读副本。该过程还会配置延迟复制和基于全局事务标识符 (GTID) 的复制。

> **Important**
>
> 要运行此过程，必须启用 `autocommit`。要启用它，请将 `autocommit` 参数设置为 `1`。有关修改参数的信息，请参阅 在 Amazon RDS 中修改数据库参数组中的参数。

语法

```
CALL mysql.rds_set_external_master_with_auto_position (
  host_name
  , host_port
  , replication_user_name
  , replication_user_password
  , ssl_encryption
  , delay
);
```

参数

host_name

在 Amazon RDS 之外运行以变为源数据库实例的 MySQL 实例的主机名或 IP 地址。

host_port

在 Amazon RDS 之外运行的要配置为源数据库实例的 MySQL 实例使用的端口。如果网络配置包括转换端口号的安全 Shell (SSH) 端口复制，请指定由 SSH 公开的端口号。

replication_user_name

对在 Amazon RDS 外部运行的 MySQL 实例具有 `REPLICATION CLIENT` 和 `REPLICATION SLAVE` 权限的用户的 ID。建议您向专用于复制的账户提供外部实例。

replication_user_password

在 `replication_user_name` 中指定的用户 ID 的密码。

ssl_encryption

指定是否在复制连接中使用安全套接字层（SSL）加密的值。1 表示使用 SSL 加密，0 表示不使用加密。默认值为 0。

> **Note**
>
> 不支持 MASTER_SSL_VERIFY_SERVER_CERT 选项。此选项设置为 0，这意味着连接已加密，但未验证证书。

delay

延迟从源数据库实例复制的最小秒数。

该参数的限制为一天（86400 秒）。

使用说明

主用户必须运行 mysql.rds_set_external_master_with_auto_position 过程。必须在要配置为在 Amazon RDS 外部运行的 MySQL 实例的只读副本的 MySQL 数据库实例上运行该过程。

所有 RDS for MySQL 5.7 版本和 RDS for MySQL 8.0.26 及更高的 8.0 版本均支持此过程。

运行 mysql.rds_set_external_master_with_auto_position 之前，您必须先将在 Amazon RDS 之外运行的 MySQL 实例配置为源数据库实例。要连接到在 Amazon RDS 外部运行的 MySQL 实例，您必须指定 replication_user_name 和 replication_user_password 值。这些值必须指示具有外部 MySQL 实例的 REPLICATION CLIENT 和 REPLICATION SLAVE 权限的复制用户。

将 MySQL 的外部实例配置为源数据库实例

1. 通过使用所选的 MySQL 客户端，连接到 MySQL 的外部实例并创建要用于复制的用户账户。以下是示例。

```
CREATE USER 'repl_user'@'mydomain.com' IDENTIFIED BY 'SomePassW0rd'
```

2. 对于 MySQL 的外部实例，向复制用户授予 REPLICATION CLIENT 和 REPLICATION SLAVE 权限。以下示例为您的域的 REPLICATION CLIENT 用户授予所有数据库的 REPLICATION SLAVE 和 'repl_user' 权限。

```
GRANT REPLICATION CLIENT, REPLICATION SLAVE ON *.* TO 'repl_user'@'mydomain.com'
IDENTIFIED BY 'SomePassW0rd'
```

有关更多信息，请参阅“配置与外部源实例之间的二进制日志文件位置复制”。

> **Note**
> 建议您尽可能使用只读副本管理两个 Amazon RDS 数据库实例之间的复制。在执行该操作时，建议您仅使用该存储过程以及其他与复制相关的存储过程。这些做法可以在 Amazon RDS 数据库实例之间实现更复杂的复制拓扑。我们提供这些存储过程主要是为了与在 Amazon RDS 外部运行的 MySQL 实例之间启用复制。有关管理 Amazon RDS 数据库实例之间的复制的信息，请参阅使用数据库实例只读副本。

在调用 `mysql.rds_set_external_master_with_auto_position` 之前，务必调用 the section called “mysql.rds_set_external_source_gtid_purged”，以便使用来自外部源的指定 GTID 范围设置 `gtid_purged` 系统变量。

在调用 `mysql.rds_set_external_master_with_auto_position` 将 Amazon RDS 数据库实例配置为只读副本后，可对该只读副本调用 mysql.rds_start_replication 开始复制过程。您可以调用 mysql.rds_reset_external_master（RDS for MariaDB 和 RDS for MySQL 主要版本 8.0 及更低版本）删除只读副本配置。

在调用 `mysql.rds_set_external_master_with_auto_position` 时，Amazon RDS 在 `set master` 和 `mysql.rds_history` 表中记录时间、用户和 `mysql.rds_replication_status` 操作。

对于灾难恢复，您可以将该过程与 mysql.rds_start_replication_until 或 mysql.rds_start_replication_until_gtid 存储过程一起使用。要将延迟只读副本的更改向前滚动到发生灾难之前的时间，您可以运行 `mysql.rds_set_external_master_with_auto_position` 过程。在 `mysql.rds_start_replication_until_gtid` 过程停止复制后，您可以通过使用将只读副本提升为独立的数据库实例中的说明将只读副本提升为新的主数据库实例。

要使用 `mysql.rds_rds_start_replication_until_gtid` 过程，必须启用基于 GTID 的复制。要跳过已知会导致灾难的特定基于 GTID 的事务，您可以使用 mysql.rds_skip_transaction_with_gtid 存储过程。有关使用基于 GTID 的复制的更多信息，请参阅使用基于 GTID 的复制。

示例

在 MySQL 数据库实例上运行时，下例将该数据库实例配置为在 Amazon RDS 之外运行的某个 MySQL 实例的只读副本。它在 MySQL 数据库实例上将最短复制延迟时间设置为一小时（3600 秒）。在 Amazon RDS 外部运行的 MySQL 源数据库实例的更改在至少一个小时内不会在 MySQL 数据库实例只读副本上应用。

```
call mysql.rds_set_external_master_with_auto_position(
```

```
  'Externaldb.some.com',
  3306,
  'repl_user',
  'SomePassW0rd',
  1,
  3600);
```

mysql.rds_set_external_source_with_auto_position（RDS for MySQL 主要版本 8.4 及更高版本）

将 RDS for MySQL 数据库实例配置为在 Amazon RDS 之外运行的 MySQL 实例的只读副本。该过程还会配置延迟复制和基于全局事务标识符 (GTID) 的复制。

> **Important**
>
> 要运行此过程，必须启用 autocommit。要启用它，请将 autocommit 参数设置为 1。有关修改参数的信息，请参阅 在 Amazon RDS 中修改数据库参数组中的参数。

语法

```
CALL mysql.rds_set_external_source_with_auto_position (
  host_name
  , host_port
  , replication_user_name
  , replication_user_password
  , ssl_encryption
  , delay
);
```

参数

host_name

在 Amazon RDS 之外运行以变为源数据库实例的 MySQL 实例的主机名或 IP 地址。

host_port

在 Amazon RDS 之外运行的要配置为源数据库实例的 MySQL 实例使用的端口。如果网络配置包括转换端口号的安全 Shell (SSH) 端口复制，请指定由 SSH 公开的端口号。

replication_user_name

对在 Amazon RDS 外部运行的 MySQL 实例具有 `REPLICATION CLIENT` 和 `REPLICATION SLAVE` 权限的用户的 ID。建议您向专用于复制的账户提供外部实例。

replication_user_password

在 `replication_user_name` 中指定的用户 ID 的密码。

ssl_encryption

指定是否在复制连接中使用安全套接字层（SSL）加密的值。1 表示使用 SSL 加密，0 表示不使用加密。默认值为 0。

> **Note**
>
> 不支持 `SOURCE_SSL_VERIFY_SERVER_CERT` 选项。此选项设置为 0，这意味着连接已加密，但未验证证书。

delay

延迟从源数据库实例复制的最小秒数。

该参数的限制为一天（86400 秒）。

使用说明

管理用户必须运行 `mysql.rds_set_external_source_with_auto_position` 过程。必须在要配置为在 Amazon RDS 外部运行的 MySQL 实例的只读副本的 MySQL 数据库实例上运行该过程。

运行 `mysql.rds_set_external_source_with_auto_position` 之前，您必须先将在 Amazon RDS 之外运行的 MySQL 实例配置为源数据库实例。要连接到在 Amazon RDS 外部运行的 MySQL 实例，您必须指定 `replication_user_name` 和 `replication_user_password` 值。这些值必须指示具有外部 MySQL 实例的 `REPLICATION CLIENT` 和 `REPLICATION SLAVE` 权限的复制用户。

将 MySQL 的外部实例配置为源数据库实例

1. 通过使用所选的 MySQL 客户端，连接到 MySQL 的外部实例并创建要用于复制的用户账户。以下是示例。

   ```
   CREATE USER 'repl_user'@'mydomain.com' IDENTIFIED BY 'SomePassW0rd'
   ```

2. 对于 MySQL 的外部实例，向复制用户授予 REPLICATION CLIENT 和 REPLICATION SLAVE 权限。以下示例为您的域的 REPLICATION CLIENT 用户授予所有数据库的 REPLICATION SLAVE 和 'repl_user' 权限。

```
GRANT REPLICATION CLIENT, REPLICATION SLAVE ON *.* TO 'repl_user'@'mydomain.com'
IDENTIFIED BY 'SomePassW0rd'
```

有关更多信息，请参阅"配置与外部源实例之间的二进制日志文件位置复制"。

> **Note**
>
> 建议您尽可能使用只读副本管理两个 Amazon RDS 数据库实例之间的复制。在执行该操作时，建议您仅使用该存储过程以及其他与复制相关的存储过程。这些做法可以在 Amazon RDS 数据库实例之间实现更复杂的复制拓扑。我们提供这些存储过程主要是为了与在 Amazon RDS 外部运行的 MySQL 实例之间启用复制。有关管理 Amazon RDS 数据库实例之间的复制的信息，请参阅使用数据库实例只读副本。

在调用 mysql.rds_set_external_source_with_auto_position 之前，务必调用 the section called "mysql.rds_set_external_source_gtid_purged"，以便使用来自外部源的指定 GTID 范围设置 gtid_purged 系统变量。

在调用 mysql.rds_set_external_source_with_auto_position 将 Amazon RDS 数据库实例配置为只读副本后，可对该只读副本调用 mysql.rds_start_replication 开始复制过程。您可以调用 mysql.rds_reset_external_source（RDS for MySQL 主要版本 8.4 及更高版本）删除只读副本配置。

在调用 mysql.rds_set_external_source_with_auto_position 时，Amazon RDS 在 set master 和 mysql.rds_history 表中记录时间、用户和 mysql.rds_replication_status 操作。

对于灾难恢复，您可以将该过程与 mysql.rds_start_replication_until 或 mysql.rds_start_replication_until_gtid 存储过程一起使用。要将延迟只读副本的更改向前滚动到发生灾难之前的时间，您可以运行 mysql.rds_set_external_source_with_auto_position 过程。在 mysql.rds_start_replication_until_gtid 过程停止复制后，您可以通过使用将只读副本提升为独立的数据库实例中的说明将只读副本提升为新的主数据库实例。

要使用 mysql.rds_rds_start_replication_until_gtid 过程，必须启用基于 GTID 的复制。要跳过已知会导致灾难的特定基于 GTID 的事务，您可以使用 mysql.rds_skip_transaction_with_gtid 存储过程。有关使用基于 GTID 的复制的更多信息，请参阅使用基于 GTID 的复制。

示例

在 MySQL 数据库实例上运行时，下例将该数据库实例配置为在 Amazon RDS 之外运行的某个 MySQL 实例的只读副本。它在 MySQL 数据库实例上将最短复制延迟时间设置为一小时（3600 秒）。在 Amazon RDS 外部运行的 MySQL 源数据库实例的更改在至少一个小时内不会在 MySQL 数据库实例只读副本上应用。

```
call mysql.rds_set_external_source_with_auto_position(
  'Externaldb.some.com',
  3306,
  'repl_user',
  'SomePassW0rd',
  1,
  3600);
```

mysql.rds_set_external_master_with_delay（RDS for MariaDB 和 RDS for MySQL 主要版本 8.0 及更低版本）

将 RDS for MySQL 数据库实例配置为在 Amazon RDS 外部运行的 MySQL 实例的只读副本并配置延迟复制。

> **Important**
>
> 要运行此过程，必须启用 autocommit。要启用它，请将 autocommit 参数设置为 1。有关修改参数的信息，请参阅 在 Amazon RDS 中修改数据库参数组中的参数。

语法

```
CALL mysql.rds_set_external_master_with_delay(
  host_name
  , host_port
  , replication_user_name
  , replication_user_password
  , mysql_binary_log_file_name
```

```
    , mysql_binary_log_file_location
    , ssl_encryption
    , delay
);
```

参数

host_name

在 Amazon RDS 之外运行并将变为源数据库实例的 MySQL 实例的主机名或 IP 地址。

host_port

在 Amazon RDS 之外运行的要配置为源数据库实例的 MySQL 实例使用的端口。如果网络配置包括转换端口号的 SSH 端口复制，则指定由 SSH 公开的端口号。

replication_user_name

对在 Amazon RDS 外部运行的 MySQL 实例具有 `REPLICATION CLIENT` 和 `REPLICATION SLAVE` 权限的用户的 ID。建议您向专用于复制的账户提供外部实例。

replication_user_password

在 `replication_user_name` 中指定的用户 ID 的密码。

mysql_binary_log_file_name

源数据库实例上包含复制信息的二进制日志的名称。

mysql_binary_log_file_location

`mysql_binary_log_file_name` 二进制日志中复制将开始读取复制信息的位置。

您可以通过在源数据库实例上运行 `SHOW MASTER STATUS` 来确定二进制日志文件名和位置。

ssl_encryption

指定是否在复制连接中使用安全套接字层（SSL）加密的值。1 表示使用 SSL 加密，0 表示不使用加密。默认值为 0。

> **Note**
>
> 不支持 `MASTER_SSL_VERIFY_SERVER_CERT` 选项。此选项设置为 0，这意味着连接已加密，但未验证证书。

delay

延迟从源数据库实例复制的最小秒数。

该参数的限制为一天（86400 秒）。

使用说明

主用户必须运行 mysql.rds_set_external_master_with_delay 过程。必须在要配置为在 Amazon RDS 外部运行的 MySQL 实例的只读副本的 MySQL 数据库实例上运行该过程。

运行 mysql.rds_set_external_master_with_delay 之前，您必须先将在 Amazon RDS 之外运行的 MySQL 实例配置为源数据库实例。要连接到在 Amazon RDS 外部运行的 MySQL 实例，您必须指定 replication_user_name 和 replication_user_password 值。这些值必须指示具有外部 MySQL 实例的 REPLICATION CLIENT 和 REPLICATION SLAVE 权限的复制用户。

将 MySQL 的外部实例配置为源数据库实例

1. 通过使用所选的 MySQL 客户端，连接到 MySQL 的外部实例并创建要用于复制的用户账户。以下是示例。

```
CREATE USER 'repl_user'@'mydomain.com' IDENTIFIED BY 'SomePassW0rd'
```

2. 对于 MySQL 的外部实例，向复制用户授予 REPLICATION CLIENT 和 REPLICATION SLAVE 权限。以下示例为您的域的 REPLICATION CLIENT 用户授予所有数据库的 REPLICATION SLAVE 和 'repl_user' 权限。

```
GRANT REPLICATION CLIENT, REPLICATION SLAVE ON *.* TO 'repl_user'@'mydomain.com'
IDENTIFIED BY 'SomePassW0rd'
```

有关更多信息，请参阅“配置与外部源实例之间的二进制日志文件位置复制”。

> **Note**
>
> 建议您尽可能使用只读副本管理两个 Amazon RDS 数据库实例之间的复制。在执行该操作时，建议您仅使用该存储过程以及其他与复制相关的存储过程。这些做法可以在 Amazon RDS 数据库实例之间实现更复杂的复制拓扑。我们提供这些存储过程主要是为了与在 Amazon RDS

外部运行的 MySQL 实例之间启用复制。有关管理 Amazon RDS 数据库实例之间的复制的信息，请参阅使用数据库实例只读副本。

在调用 `mysql.rds_set_external_master_with_delay` 将 Amazon RDS 数据库实例配置为只读副本后，可对该只读副本调用 mysql.rds_start_replication 开始复制过程。您可以调用 mysql.rds_reset_external_master（RDS for MariaDB 和 RDS for MySQL 主要版本 8.0 及更低版本）删除只读副本配置。

在调用 `mysql.rds_set_external_master_with_delay` 时，Amazon RDS 在 `set master` 和 `mysql.rds_history` 表中记录时间、用户和 `mysql.rds_replication_status` 操作。

对于灾难恢复，您可以将该过程与 mysql.rds_start_replication_until 或 mysql.rds_start_replication_until_gtid 存储过程一起使用。要将延迟只读副本的更改向前滚动到发生灾难之前的时间，您可以运行 `mysql.rds_set_external_master_with_delay` 过程。在 `mysql.rds_start_replication_until` 过程停止复制后，您可以通过使用将只读副本提升为独立的数据库实例中的说明将只读副本提升为新的主数据库实例。

要使用 `mysql.rds_rds_start_replication_until_gtid` 过程，必须启用基于 GTID 的复制。要跳过已知会导致灾难的特定基于 GTID 的事务，您可以使用 mysql.rds_skip_transaction_with_gtid 存储过程。有关使用基于 GTID 的复制的更多信息，请参阅使用基于 GTID 的复制。

`mysql.rds_set_external_master_with_delay` 过程在以下版本的 RDS for MySQL 中可用：

- MySQL 8.0.26 和更高的 8.0.x 版本
- 所有 5.7 版本

示例

在 MySQL 数据库实例上运行时，下例将该数据库实例配置为在 Amazon RDS 之外运行的某个 MySQL 实例的只读副本。它在 MySQL 数据库实例上将最短复制延迟时间设置为一小时（3600 秒）。在 Amazon RDS 外部运行的 MySQL 源数据库实例的更改在至少一个小时内不会在 MySQL 数据库实例只读副本上应用。

```
call mysql.rds_set_external_master_with_delay(
  'Externaldb.some.com',
  3306,
  'repl_user',
```

```
  'SomePassW0rd',
  'mysql-bin-changelog.000777',
  120,
  1,
  3600);
```

mysql.rds_set_external_source_with_delay（RDS for MySQL 主要版本 8.4 及更高版本）

将 RDS for MySQL 数据库实例配置为在 Amazon RDS 外部运行的 MySQL 实例的只读副本并配置延迟复制。

> **Important**
>
> 要运行此过程，必须启用 `autocommit`。要启用它，请将 `autocommit` 参数设置为 `1`。有关修改参数的信息，请参阅 [在 Amazon RDS 中修改数据库参数组中的参数]。

语法

```
CALL mysql.rds_set_external_source_with_delay (
  host_name
  , host_port
  , replication_user_name
  , replication_user_password
  , mysql_binary_log_file_name
  , mysql_binary_log_file_location
  , ssl_encryption
  , delay
);
```

参数

`host_name`

在 Amazon RDS 之外运行并将变为源数据库实例的 MySQL 实例的主机名或 IP 地址。

`host_port`

在 Amazon RDS 之外运行的要配置为源数据库实例的 MySQL 实例使用的端口。如果网络配置包括转换端口号的 SSH 端口复制，则指定由 SSH 公开的端口号。

replication_user_name

对在 Amazon RDS 外部运行的 MySQL 实例具有 REPLICATION CLIENT 和 REPLICATION SLAVE 权限的用户的 ID。建议您向专用于复制的账户提供外部实例。

replication_user_password

在 replication_user_name 中指定的用户 ID 的密码。

mysql_binary_log_file_name

源数据库实例上包含复制信息的二进制日志的名称。

mysql_binary_log_file_location

mysql_binary_log_file_name 二进制日志中复制将开始读取复制信息的位置。

您可以通过在源数据库实例上运行 SHOW MASTER STATUS 来确定二进制日志文件名和位置。

ssl_encryption

指定是否在复制连接中使用安全套接字层（SSL）加密的值。1 表示使用 SSL 加密，0 表示不使用加密。默认值为 0。

> **Note**
>
> 不支持 SOURCE_SSL_VERIFY_SERVER_CERT 选项。此选项设置为 0，这意味着连接已加密，但未验证证书。

delay

延迟从源数据库实例复制的最小秒数。

该参数的限制为一天（86400 秒）。

使用说明

管理用户必须运行 mysql.rds_set_external_source_with_delay 过程。必须在要配置为在 Amazon RDS 外部运行的 MySQL 实例的只读副本的 MySQL 数据库实例上运行该过程。

运行 mysql.rds_set_external_source_with_delay 之前，您必须先将在 Amazon RDS 之外运行的 MySQL 实例配置为源数据库实例。要连接到在 Amazon RDS 外部运行的 MySQL 实例，您必

须指定 replication_user_name 和 replication_user_password 值。这些值必须指示具有外部 MySQL 实例的 REPLICATION CLIENT 和 REPLICATION SLAVE 权限的复制用户。

将 MySQL 的外部实例配置为源数据库实例

1. 通过使用所选的 MySQL 客户端，连接到 MySQL 的外部实例并创建要用于复制的用户账户。以下是示例。

```
CREATE USER 'repl_user'@'mydomain.com' IDENTIFIED BY 'SomePassW0rd'
```

2. 对于 MySQL 的外部实例，向复制用户授予 REPLICATION CLIENT 和 REPLICATION SLAVE 权限。以下示例为您的域的 REPLICATION CLIENT 用户授予所有数据库的 REPLICATION SLAVE 和 'repl_user' 权限。

```
GRANT REPLICATION CLIENT, REPLICATION SLAVE ON *.* TO 'repl_user'@'mydomain.com'
IDENTIFIED BY 'SomePassW0rd'
```

有关更多信息，请参阅“配置与外部源实例之间的二进制日志文件位置复制”。

> **Note**
>
> 建议您尽可能使用只读副本管理两个 Amazon RDS 数据库实例之间的复制。在执行该操作时，建议您仅使用该存储过程以及其他与复制相关的存储过程。这些做法可以在 Amazon RDS 数据库实例之间实现更复杂的复制拓扑。我们提供这些存储过程主要是为了与在 Amazon RDS 外部运行的 MySQL 实例之间启用复制。有关管理 Amazon RDS 数据库实例之间的复制的信息，请参阅使用数据库实例只读副本。

在调用 mysql.rds_set_external_source_with_delay 将 Amazon RDS 数据库实例配置为只读副本后，可对该只读副本调用 mysql.rds_start_replication 开始复制过程。您可以调用 mysql.rds_reset_external_source（RDS for MySQL 主要版本 8.4 及更高版本） 删除只读副本配置。

在调用 mysql.rds_set_external_source_with_delay 时，Amazon RDS 在 set master 和 mysql.rds_history 表中记录时间、用户和 mysql.rds_replication_status 操作。

对于灾难恢复，您可以将该过程与 mysql.rds_start_replication_until 或 mysql.rds_start_replication_until_gtid 存储过程一起使用。要将延迟只读副本的更改向前滚动到发生灾难之前的时间，您可以运行 mysql.rds_set_external_source_with_delay 过程。在

mysql.rds_start_replication_until 过程停止复制后，您可以通过使用将只读副本提升为独立的数据库实例中的说明将只读副本提升为新的主数据库实例。

要使用 mysql.rds_rds_start_replication_until_gtid 过程，必须启用基于 GTID 的复制。要跳过已知会导致灾难的特定基于 GTID 的事务，您可以使用 mysql.rds_skip_transaction_with_gtid 存储过程。有关使用基于 GTID 的复制的更多信息，请参阅使用基于 GTID 的复制。

示例

在 MySQL 数据库实例上运行时，下例将该数据库实例配置为在 Amazon RDS 之外运行的某个 MySQL 实例的只读副本。它在 MySQL 数据库实例上将最短复制延迟时间设置为一小时（3600 秒）。在 Amazon RDS 外部运行的 MySQL 源数据库实例的更改在至少一个小时内不会在 MySQL 数据库实例只读副本上应用。

```
call mysql.rds_set_external_source_with_delay(
  'Externaldb.some.com',
  3306,
  'repl_user',
  'SomePassW0rd',
  'mysql-bin-changelog.000777',
  120,
  1,
  3600);
```

mysql.rds_set_external_source_gtid_purged

使用来自外部源的指定 GTID 范围设置 gtid_purged 系统变量。配置基于 GTID 的复制以使用自动定位来恢复复制时，需要 gtid_purged 值。

> **Important**
> 要运行此过程，必须启用 autocommit。要启用它，请将 autocommit 参数设置为 1。有关修改参数的信息，请参阅 在 Amazon RDS 中修改数据库参数组中的参数。

语法

```
CALL mysql.rds_set_external_source_gtid_purged(
  server_uuid
  , start_pos
```

```
  , end_pos
);
```

参数

server_uuid

从中导入 GTID 范围的外部服务器的通用唯一标识符（UUID）。

start_pos

要设置的 GTID 范围的起始位置。

end_pos

要设置的 GTID 范围的结束位置。

使用说明

`mysql.rds_set_external_source_gtid_purged` 过程仅适用于 MySQL 8.0.37 及更高的 8.0 版本。

先调用 `mysql.rds_set_external_source_gtid_purged`，然后再调用 mysql.rds_set_external_master_with_auto_position（RDS for MySQL 主要版本 8.0 及更低版本）、mysql.rds_set_external_source_with_auto_position（RDS for MySQL 主要版本 8.4 及更高版本）或 mysql.rds_set_external_source_with_auto_position_for_channel。

在调用 `mysql.rds_set_external_source_gtid_purged` 之前，请务必停止数据库的所有活动的复制通道。要检查通道的状态，请使用 `SHOW REPLICA STATUS` MySQL 语句。要停止通道上的复制，请调用 the section called “mysql.rds_stop_replication_for_channel”。

您指定的 GTID 范围必须是现有 `GTID_PURGED` 值的超集。此存储过程在设置 `GTID_PURGED` 值之前会检查以下各值：

- `server_uuid` 有效。
- `start_pos` 的值大于 `0` 并小于 `end_pos` 的值。
- `end_pos` 的值大于或等于 `start_pos` 的值。

如果在外部服务器上设置的 GTID 包含多个值范围，请考虑使用不同的 GTID 设置值多次调用该过程。

在调用 `mysql.rds_set_external_source_gtid_purged` 时，Amazon RDS 会在 `mysql.rds_history` 表中记录 `set gtid_purged` 的时间、用户和操作。

如果您没有为用于复制的备份适当地设置 gtid_purged 值，这可能导致复制过程中事务丢失或重复。执行以下步骤来设置正确的 gtid_purged 值。

对副本设置 gtid_purged 值

1. 确定要用作复制起点的时间点或特定的备份文件。这可能是逻辑备份（mysqldump 文件）或物理备份（Amazon RDS 快照）。
2. 确定 gtid_executed 值。此值表示在服务器上提交的所有 GTID 的集合。要检索此值，请在源实例上执行下列操作之一：
   - 在执行备份时运行 SQL 语句 SELECT @@GLOBAL.GTID_EXECUTED;。
   - 如果相应的备份实用程序中包含任何相关选项，请从备份文件中提取值。有关更多信息，请参阅 MySQL 文档中的 set-gtid-purged 选项。
3. 确定用于调用 mysql.rds_set_external_source_gtid_purged 的 gtid_purged 值。gtid_purged 值应包括已在源实例上执行但不再需要用于复制的所有 GTID。因此，gtid_purged 值应是您在上一步中检索到的 gtid_executed 值的子集。

   要确定 gtid_purged 值，请确定备份中未包含的且不再需要用于复制的 GTID。为此，可以通过分析二进制日志或使用 mysqlbinlog 等工具，来查找从二进制日志中清除的 GTID。

   或者，如果您有一个一致的备份，其中包括截至备份点的所有二进制日志，则可以将 gtid_purged 值设置为与备份点处的 gtid_executed 值相同。
4. 确定与备份一致的适当 gtid_purged 值后，在 RDS for MySQL 数据库实例上调用 mysql.rds_set_external_source_gtid_purged 存储过程来设置该值。

示例

在 MySQL 数据库实例上运行时，以下示例将从 UUID 为 12345678-abcd-1234-efgh-123456789abc 的外部 MySQL 服务器设置 GTID 范围，起始位置为 1，结束位置为 100。生成的 GTID 值设置为 +12345678-abcd-1234-efgh-123456789abc:1-100。

```
CALL mysql.rds_set_external_source_gtid_purged('12345678-abcd-1234-efgh-123456789abc',
 1, 100);
```

mysql.rds_set_master_auto_position（RDS for MySQL 主要版本 8.0 及更低版本）

将复制模式设置为基于二进制日志文件位置或全局事务标识符 (GTID)。

语法

```
CALL mysql.rds_set_master_auto_position (
auto_position_mode
);
```

参数

auto_position_mode

该值指示是使用日志文件位置复制还是基于 GTID 的复制：

- `0` – 使用基于二进制日志文件位置的复制方法。默认为 `0`。
- `1` – 使用基于 GTID 的复制方法。

使用说明

主用户必须运行 `mysql.rds_set_master_auto_position` 过程。

所有 RDS for MySQL 5.7 版本和 RDS for MySQL 8.0.26 及更高的 8.0 版本均支持此过程。

mysql.rds_set_source_auto_position（RDS for MySQL 主要版本 8.4 及更高版本）

将复制模式设置为基于二进制日志文件位置或全局事务标识符 (GTID)。

语法

```
CALL mysql.rds_set_source_auto_position (auto_position_mode);
```

参数

auto_position_mode

该值指示是使用日志文件位置复制还是基于 GTID 的复制：

- `0` – 使用基于二进制日志文件位置的复制方法。默认为 `0`。
- `1` – 使用基于 GTID 的复制方法。

使用说明

管理用户必须运行 `mysql.rds_set_source_auto_position` 过程。

mysql.rds_set_source_delay

设置延迟从源数据库实例复制到当前只读副本的最小秒数。在您连接到只读副本时使用该过程来从其源数据库实例延迟复制。

语法

```
CALL mysql.rds_set_source_delay(
delay
);
```

参数

delay

延迟从源数据库实例复制的最小秒数。

该参数的限制为一天（86400 秒）。

使用说明

主用户必须运行 `mysql.rds_set_source_delay` 过程。

对于灾难恢复，您可以将该过程与 mysql.rds_start_replication_until 或 mysql.rds_start_replication_until_gtid 存储过程一起使用。要将延迟只读副本的更改向前滚动到发生灾难之前的时间，您可以运行 `mysql.rds_set_source_delay` 过程。在 `mysql.rds_start_replication_until` 或 `mysql.rds_start_replication_until_gtid` 过程停止复制后，您可以使用将只读副本提升为独立的数据库实例中的说明将只读副本提升为新的主数据库实例。

要使用 `mysql.rds_rds_start_replication_until_gtid` 过程，必须启用基于 GTID 的复制。要跳过已知会导致灾难的特定基于 GTID 的事务，您可以使用 mysql.rds_skip_transaction_with_gtid 存储过程。有关基于 GTID 的复制的更多信息，请参阅 使用基于 GTID 的复制。

`mysql.rds_set_source_delay` 过程在以下版本的 RDS for MySQL 中可用：

- 所有 RDS for MySQL 8.4 版本
- MySQL 8.0.26 和更高的 8.0.x 版本
- 所有 5.7 版本

示例

要将从源数据库实例向当前只读副本的复制延迟至少一个小时（3600 秒），您可以使用以下参数调用 mysql.rds_set_source_delay：

```
CALL mysql.rds_set_source_delay(3600);
```

mysql.rds_skip_repl_error

跳过并删除 MySQL 数据库只读副本上的复制错误。

语法

```
CALL mysql.rds_skip_repl_error;
```

使用说明

主用户必须对只读副本运行 mysql.rds_skip_repl_error 过程。有关此过程的更多信息，请参阅调用 mysql.rds_skip_repl_error 过程。

要确定是否存在错误，请运行 MySQL SHOW REPLICA STATUS\G 命令。如果复制错误不太严重，您可以运行 mysql.rds_skip_repl_error 以跳过该错误。如果有多个错误，mysql.rds_skip_repl_error 会删除第一个错误，并警告存在其他错误。然后，您可以使用 SHOW REPLICA STATUS\G 确定要对下一个错误采取的适当操作。有关返回的值的信息，请参阅 MySQL 文档中的 SHOW REPLICA STATUS 语法。

想要了解更多有关解决 Amazon RDS 的复制错误的信息，请参阅 排查 MySQL 只读副本问题。

复制已停止错误

调用 mysql.rds_skip_repl_error 过程时，您可能会收到一条错误消息，指出副本已关闭或禁用。

如果您对于主实例而不是只读副本运行该过程，则会出现此错误消息。您必须对只读副本运行此过程，该过程才能正常运行。

如果您对只读副本运行该过程，但无法成功重新启动复制，也可能会出现此错误消息。

如果您需要跳过大量错误，复制滞后时间可能会超出二进制日志 (binlog) 文件的默认保留期。在这种情况下，您可能会遇到一个因二进制日志文件在只读副本上进行重放之前被清除而引发的严重错误。此清除会导致复制停止，而您将无法再调用 mysql.rds_skip_repl_error 命令以跳过复制错误。

您可以增加在源数据库实例上保留 binlog 文件的小时数以缓解该问题。在增加二进制日志保留时间后，您可以重新启动复制进程，并根据需要调用 `mysql.rds_skip_repl_error` 命令。

要设置 binlog 保留时间，请使用 mysql.rds_set_configuration 过程，并指定 `'binlog retention hours'` 配置参数以及在数据库集群上保留 binlog 文件的小时数。以下示例将 binlog 文件的保留期设置为 48 个小时。

```
CALL mysql.rds_set_configuration('binlog retention hours', 48);
```

mysql.rds_start_replication

从 RDS for MySQL 数据库实例发起复制。

> **Note**
>
> 您可以使用 mysql.rds_start_replication_until 或 mysql.rds_start_replication_until_gtid 存储过程从 RDS for MySQL 数据库实例中启动复制，并在指定的二进制日志文件位置停止复制。

语法

```
CALL mysql.rds_start_replication;
```

使用说明

主用户必须运行 `mysql.rds_start_replication` 过程。

要从 Amazon RDS 外部的 MySQL 实例导入数据，请在调用 mysql.rds_set_external_master（RDS for MariaDB 和 RDS for MySQL 主要版本 8.0 及更低版本） 或 mysql.rds_set_external_source（RDS for MySQL 主要版本 8.4 及更高版本） 以构建复制配置后，再对只读副本调用 `mysql.rds_start_replication` 以开始复制过程。有关更多信息，请参阅 将备份还原到 Amazon RDS for MySQL 数据库实例。

要将数据导出到 Amazon RDS 外部的 MySQL 实例，请对只读副本调用 `mysql.rds_start_replication` 和 `mysql.rds_stop_replication` 以控制某些复制操作，如清除二进制日志。有关更多信息，请参阅 使用复制从 MySQL 数据库实例中导出数据。

您还可以对只读副本调用 `mysql.rds_start_replication`，以便重新启动您先前通过调用 `mysql.rds_stop_replication` 停止的任何复制过程。有关更多信息，请参阅 使用数据库实例只读副本。

mysql.rds_start_replication_until

从 RDS for MySQL 数据库实例发起复制，并在指定的二进制日志文件位置停止复制。

语法

```
CALL mysql.rds_start_replication_until (
replication_log_file
  , replication_stop_point
);
```

参数

replication_log_file

源数据库实例上包含复制信息的二进制日志的名称。

replication_stop_point

`replication_log_file` 二进制日志中复制将停止的位置。

使用说明

主用户必须运行 `mysql.rds_start_replication_until` 过程。

`mysql.rds_start_replication_until` 过程在以下版本的 RDS for MySQL 中可用：

- 所有 RDS for MySQL 8.4 版本
- MySQL 8.0.26 和更高的 8.0.x 版本
- 所有 5.7 版本

您可以对延迟复制使用该过程以进行灾难恢复。如果您已配置延迟复制，则可以使用该过程来将延迟只读副本的更改向前滚动到灾难之前的时间。在该过程停止复制后，您可以通过使用将只读副本提升为独立的数据库实例中的说明，将只读副本提升为新的主数据库实例。

您可以使用以下存储过程配置延迟复制：

- mysql.rds_set_configuration
- mysql.rds_set_external_master_with_delay（RDS for MariaDB 和 RDS for MySQL 主要版本 8.0 及更低版本）

- mysql.rds_set_external_source_with_delay（RDS for MySQL 主要版本 8.4 及更高版本）
- mysql.rds_set_source_delay

为 replication_log_file 参数指定的文件名必须与源数据库实例二进制日志文件名匹配。

当 replication_stop_point 参数指定位于过去的某个停止位置时，即会立即停止复制。

示例

以下示例将启动复制并复制更改，直到它到达 120 二进制日志文件中的 mysql-bin-changelog.000777 位置。

```
call mysql.rds_start_replication_until(
  'mysql-bin-changelog.000777',
  120);
```

mysql.rds_stop_replication

停止从 MySQL 数据库实例中进行复制。

语法

```
CALL mysql.rds_stop_replication;
```

使用说明

主用户必须运行 mysql.rds_stop_replication 过程。

如果要配置复制，使其从在 Amazon RDS 外部运行的 MySQL 实例导入数据，则要在导入完成后，再对只读副本调用 mysql.rds_stop_replication 以停止复制过程。有关更多信息，请参阅 将备份还原到 Amazon RDS for MySQL 数据库实例。

如果要配置复制，使其将数据导出到在 Amazon RDS 外部运行的 MySQL 实例，则要对只读副本调用 mysql.rds_start_replication 和 mysql.rds_stop_replication 以控制某些复制操作，如清除二进制日志。有关更多信息，请参阅 使用复制从 MySQL 数据库实例中导出数据。

您还可以使用 mysql.rds_stop_replication 停止两个 Amazon RDS 数据库实例之间的复制。停止复制通常是为了对只读副本执行长时间运行的操作，如在只读副本上创建大型索引。您可以重新启动

您通过对只读副本调用 mysql.rds_start_replication 停止的任何复制过程。有关更多信息，请参阅 使用数据库实例只读副本。

结束会话或查询

以下存储过程结束会话或查询。

主题

- mysql.rds_kill
- mysql.rds_kill_query

mysql.rds_kill

结束与 MySQL 服务器的连接。

语法

```
CALL mysql.rds_kill(processID);
```

参数

processID

要结束的连接线程的标识。

使用说明

与 MySQL 服务器的每个连接在单独的线程中运行。要结束连接，请使用 `mysql.rds_kill` 过程并传入该连接的线程 ID。要获取线程 ID，请使用 MySQL SHOW PROCESSLIST 命令。

有关限制的信息，请参阅MySQL 存储过程限制。

示例

以下示例结束线程 ID 为 4243 的连接：

```
CALL mysql.rds_kill(4243);
```

mysql.rds_kill_query

结束针对 MySQL 服务器运行的查询。

语法

```
CALL mysql.rds_kill_query(processID);
```

参数

processID

运行正要结束的查询的进程或线程的身份。

使用说明

要停止针对 MySQL 服务器运行的查询，请使用 `mysql_rds_kill_query` 过程并传入正在运行查询的线程的连接 ID。然后，该过程将终止连接。

要获取 ID，请查询 MySQL INFORMATION_SCHEMA PROCESSLIST 表或使用 MySQL SHOW PROCESSLIST 命令。`SHOW PROCESSLIST` 或 `SELECT * FROM INFORMATION_SCHEMA.PROCESSLIST` 中 ID 列的值为 *processID*。

有关限制的信息，请参阅MySQL 存储过程限制。

示例

以下示例停止查询线程 ID 为 230040 的查询：

```
CALL mysql.rds_kill_query(230040);
```

管理主动-主动集群

以下存储过程用于设置和管理 RDS for MySQL 主动-主动集群。有关更多信息，请参阅 the section called “配置主动-主动集群”。

这些存储过程仅适用于运行以下版本的 RDS for MySQL 数据库实例：

- 所有 MySQL 8.4 版本
- MySQL 8.0.35 及更高的次要版本

主题

mysql.rds_group_replication_advance_gtid

在当前数据库实例上创建占位符 GTID。

语法

```
CALL mysql.rds_group_replication_advance_gtid(
begin_id
, end_id
, server_uuid
);
```

参数

begin_id

要创建的开始事务 ID。

end_id

要创建的结束事务 ID。

begin_id

要创建的事务的 group_replication_group_name。在与数据库实例关联的数据库参数组中将 group_replication_group_name 指定为 UUID。

使用说明

在主动-主动集群中，要使数据库实例加入组，在新数据库实例上执行的所有 GTID 事务都必须存在于集群中的其它成员上。在不寻常的情况下，在将实例加入组之前执行事务时，新的数据库实例可能会有更多的事务。在这种情况下，您无法移除任何现有事务，但您可以使用此过程在组中的其它数据库实例上创建相应的占位符 GTID。在执行此操作之前，请验证事务不会影响复制的数据。

当您调用此过程时，将创建 server_uuid:begin_id-end_id 的 GTID 事务，但内容为空。为避免复制问题，请勿在任何其它条件下使用此过程。

> **Important**
> 避免在主动-主动集群运行正常时调用此过程。除非您了解正在创建的事务可能产生的后果，否则不要调用此过程。调用此过程可能会导致数据不一致。

示例

以下示例在当前数据库实例上创建占位符 GTID：

```
CALL mysql.rds_group_replication_advance_gtid(5, 6,
 '11111111-2222-3333-4444-555555555555');
```

mysql.rds_group_replication_create_user

为数据库实例上的组复制创建复制用户 rdsgrprepladmin。

语法

```
CALL mysql.rds_group_replication_create_user(
replication_user_password
);
```

参数

replication_user_password

复制用户 rdsgrprepladmin 的密码。

使用说明

- 在主动-主动集群中的所有数据库实例上，复制用户 rdsgrprepladmin 的密码必须相同。
- rdsgrprepladmin 用户名保留用于组复制连接。任何其他用户（包括主用户）都不能拥有此用户名。

示例

以下示例为数据库实例上的组复制创建复制用户 rdsgrprepladmin：

```
CALL mysql.rds_group_replication_create_user('password');
```

mysql.rds_group_replication_set_recovery_channel

设置主动-主动集群的 group_replication_recovery 通道。该过程使用保留的 rdsgrprepladmin 用户来配置通道。

语法

```
CALL mysql.rds_group_replication_set_recovery_channel(
replication_user_password);
```

参数

replication_user_password

复制用户 rdsgrprepladmin 的密码。

使用说明

在主动-主动集群中的所有数据库实例上，复制用户 rdsgrprepladmin 的密码必须相同。调用 mysql.rds_group_replication_create_user 可指定密码。

示例

以下示例为主动-主动集群设置 group_replication_recovery 通道：

```
CALL mysql.rds_group_replication_set_recovery_channel('password');
```

mysql.rds_group_replication_start

在当前数据库实例上启动组复制。

语法

```
CALL mysql.rds_group_replication_start(
bootstrap
);
```

参数

bootstrap

一个值，它指定是初始化新组还是加入现有组。值为 1 时，将使用当前数据库实例初始化新组。值为 0 时，将通过连接到在与当前数据库实例关联的数据库参数组的 group_replication_group_seeds 参数中定义的端点，将该数据库实例加入到现有组。

示例

以下示例使用当前数据库实例初始化新组：

```
CALL mysql.rds_group_replication_start(1);
```

mysql.rds_group_replication_stop

在当前数据库实例上停止组复制。

语法

```
CALL mysql.rds_group_replication_stop();
```

使用说明

当您在数据库实例上停止复制时，它不会影响主动-主动集群中的任何其它数据库实例。

管理多源复制

以下存储过程在 RDS for MySQL 多源副本上设置和管理复制通道。有关更多信息，请参阅 the section called "配置多源复制"。

这些存储过程仅适用于运行以下引擎版本的 RDS for MySQL 数据库实例：

- 所有 8.4 版本
- 8.0.35 及更高的次要版本
- 5.7.44 及更高的次要版本

在使用这些存储过程管理复制时，如果其复制用户配置了 `caching_sha2_passwword`，则必须通过指定 `SOURCE_SSL=1` 来配置 TLS。`caching_sha2_password` 是 RDS for MySQL 8.4 的默认身份验证插件。

> **Note**
> 尽管本文档将源数据库实例称为 RDS for MySQL 数据库实例，但这些过程也适用于在 Amazon RDS 外部运行的 MySQL 实例。

主题

mysql.rds_next_source_log_for_channel

将源数据库实例日志位置更改为通道的源数据库实例上下一个二进制日志的开始位置。只有在多源副本上收到复制 I/O 错误 1236 时，才能使用该过程。

语法

```
CALL mysql.rds_next_source_log_for_channel(
curr_master_log,
channel_name
);
```

参数

curr_master_log

当前源日志文件的索引。例如，如果当前文件名为 mysql-bin-changelog.012345，则索引为 12345。要确定当前源日志文件名，请运行 SHOW REPLICA STATUS FOR CHANNEL '*channel_name*' 命令并查看 Source_Log_File 字段。

channel_name

多源副本上复制通道的名称。每个复制通道接收来自在特定主机和端口上运行的单个源 RDS for MySQL 数据库实例的二进制日志事件。

使用说明

主用户必须运行 mysql.rds_next_source_log_for_channel 过程。例如，如果出现 IO_Thread 错误，则可以使用此过程跳过当前二进制日志文件中的所有事件，并从在 channel_name 中指定的通道的下一个二进制日志文件中恢复复制。

示例

假设在多源副本的通道上复制失败。对多源副本运行 SHOW REPLICA STATUS FOR CHANNEL 'channel_1'\G 会返回以下结果：

```
mysql> SHOW REPLICA STATUS FOR CHANNEL 'channel_1'\G
*************************** 1. row ***************************
             Replica_IO_State: Waiting for source to send event
                  Source_Host: myhost.XXXXXXXXXXXXXXX.rr-rrrr-1.rds.amazonaws.com
                  Source_User: ReplicationUser
                  Source_Port: 3306
```

```
                 Connect_Retry: 60
               Source_Log_File: mysql-bin-changelog.012345
           Read_Source_Log_Pos: 1219393
                Relay_Log_File: replica-relay-bin.000003
                 Relay_Log_Pos: 30223388
         Relay_Source_Log_File: mysql-bin-changelog.012345
            Replica_IO_Running: No
           Replica_SQL_Running: Yes
               Replicate_Do_DB:.
               .
               .
                 Last_IO_Errno: 1236
                 Last_IO_Error: Got fatal error 1236 from master when reading data from
 binary log: 'Client requested master to start replication from impossible position;
 the first event 'mysql-bin-changelog.013406' at 1219393, the last event read from
 '/rdsdbdata/log/binlog/mysql-bin-changelog.012345' at 4, the last byte read from '/
rdsdbdata/log/binlog/mysql-bin-changelog.012345' at 4.'
                Last_SQL_Errno: 0
                Last_SQL_Error:
                .
                .
                  Channel_name: channel_1
               .
               .
 -- Some fields are omitted in this example output
```

`Last_IO_Errno` 字段显示该实例收到 I/O 错误 1236。`Source_Log_File` 字段显示文件名为 `mysql-bin-changelog.012345`，这意味着日志文件索引为 `12345`。要纠正该错误，您可以使用以下参数调用 `mysql.rds_next_source_log_for_channel`：

```
CALL mysql.rds_next_source_log_for_channel(12345,'channel_1');
```

mysql.rds_reset_external_source_for_channel

停止指定通道上的复制过程，并从多源副本中移除该通道和关联的配置。

> **Important**
>
> 要运行此过程，必须启用 `autocommit`。要启用它，请将 `autocommit` 参数设置为 `1`。有关修改参数的信息，请参阅 在 Amazon RDS 中修改数据库参数组中的参数。

语法

```
CALL mysql.rds_reset_external_source_for_channel (channel_name);
```

参数

channel_name

多源副本上复制通道的名称。每个复制通道接收来自在特定主机和端口上运行的单个源 RDS for MySQL 数据库实例的二进制日志事件。

使用说明

主用户必须运行 `mysql.rds_reset_external_source_for_channel` 过程。此过程将删除属于要移除的通道的所有中继日志。

mysql.rds_set_external_source_for_channel

在 RDS for MySQL 数据库实例上配置复制通道，以从另一个 RDS for MySQL 数据库实例复制数据。

> **Important**
>
> 要运行此过程，必须启用 `autocommit`。要启用它，请将 `autocommit` 参数设置为 `1`。有关修改参数的信息，请参阅 在 Amazon RDS 中修改数据库参数组中的参数。

> **Note**
>
> 您可以改用 the section called “mysql.rds_set_external_source_with_delay_for_channel” 存储过程为该通道配置延迟复制。

语法

```
CALL mysql.rds_set_external_source_for_channel (
  host_name
  , host_port
```

```
  , replication_user_name
  , replication_user_password
  , mysql_binary_log_file_name
  , mysql_binary_log_file_location
  , ssl_encryption
  , channel_name
);
```

参数

host_name

RDS for MySQL 源数据库实例的主机名或 IP 地址。

host_port

RDS for MySQL 源数据库实例使用的端口。如果网络配置包括转换端口号的安全 Shell (SSH) 端口复制，请指定由 SSH 公开的端口号。

replication_user_name

对 RDS for MySQL 源数据库实例具有 `REPLICATION CLIENT` 和 `REPLICATION SLAVE` 权限的用户的 ID。建议您向专用于复制的账户提供源数据库实例。

replication_user_password

在 `replication_user_name` 中指定的用户 ID 的密码。

mysql_binary_log_file_name

源数据库实例上包含复制信息的二进制日志的名称。

mysql_binary_log_file_location

`mysql_binary_log_file_name` 二进制日志中复制将开始读取复制信息的位置。

您可以通过在源数据库实例上运行 `SHOW BINARY LOG STATUS` 来确定二进制日志文件名和位置。

> **Note**
>
> 以前的 MySQL 版本使用的是 `SHOW MASTER STATUS`，而不是 `SHOW BINARY LOG STATUS`。如果您使用的 MySQL 版本低于 8.4，请使用 `SHOW MASTER STATUS`。

ssl_encryption

指定是否在复制连接中使用安全套接字层（SSL）加密的值。1 表示使用 SSL 加密，0 表示不使用加密。默认值为 0。

> **Note**
>
> 不支持 SOURCE_SSL_VERIFY_SERVER_CERT 选项。此选项设置为 0，这意味着连接已加密，但未验证证书。

channel_name

复制通道的名称。每个复制通道接收来自在特定主机和端口上运行的单个源 RDS for MySQL 数据库实例的二进制日志事件。

使用说明

主用户必须运行 `mysql.rds_set_external_source_for_channel` 过程。此过程必须在您要创建复制通道的目标 RDS for MySQL 数据库实例上运行。

在运行 `mysql.rds_set_external_source_for_channel` 之前，请为源数据库实例上的复制用户配置多源副本所需的权限。要将多源副本连接到源数据库实例，必须指定对源数据库实例拥有 `REPLICATION CLIENT` 和 `REPLICATION SLAVE` 权限的复制用户的 `replication_user_name` 和 `replication_user_password` 值。

在源数据库实例上配置复制用户

1. 通过使用所选的 MySQL 客户端，连接到源数据库实例并创建要用于复制的用户账户。示例如下：

   > **Important**
   >
   > 作为安全最佳实践，请指定除以下示例中所示占位符值以外的密码。

   ```
   CREATE USER 'repl_user'@'example.com' IDENTIFIED BY 'password';
   ```

2. 在源数据库实例上，向复制用户授予 `REPLICATION CLIENT` 和 `REPLICATION SLAVE` 权限。以下示例向您所在域的“repl_user”用户授予对所有数据库的 `REPLICATION CLIENT` 和 `REPLICATION SLAVE` 权限。

```
GRANT REPLICATION CLIENT, REPLICATION SLAVE ON *.* TO 'repl_user'@'example.com';
```

要使用加密复制，请将源数据库实例配置为使用 SSL 连接。

在调用 `mysql.rds_set_external_source_for_channel` 以配置此复制通道后，可以在副本上调用 mysql.rds_start_replication_for_channel 以启动通道上的复制过程。您可以调用 the section called “mysql.rds_reset_external_source_for_channel” 以停止通道上的复制，并从副本中移除通道配置。

当您调用 `mysql.rds_set_external_source_for_channel` 时，Amazon RDS 将时间、用户和 `set channel source` 的操作记录在 `mysql.rds_history` 表（不包括通道特定的详细信息）和 `mysql.rds_replication_status` 表中，并包含通道名称。此信息仅用于内部使用和监控目的。要记录完整的过程调用以进行审核，请考虑根据应用程序的特定要求启用审核日志或常规日志。

示例

在 RDS for MySQL 数据库实例上运行时，以下示例将此数据库实例上名为 `channel_1` 的复制通道配置为从由主机 `sourcedb.example.com` 和端口 `3306` 指定的源复制数据。

```
call mysql.rds_set_external_source_for_channel(
  'sourcedb.example.com',
  3306,
  'repl_user',
  'password',
  'mysql-bin-changelog.0777',
  120,
  0,
  'channel_1');
```

mysql.rds_set_external_source_with_auto_position_for_channel

在 RDS for MySQL 数据库实例上配置一个复制通道，复制延迟为可选项。复制基于全局事务标识符（GTID）。

> **Important**
>
> 要运行此过程，必须启用 `autocommit`。要启用它，请将 `autocommit` 参数设置为 `1`。有关修改参数的信息，请参阅 在 Amazon RDS 中修改数据库参数组中的参数。

语法

```
CALL mysql.rds_set_external_source_with_auto_position_for_channel (
  host_name
  , host_port
  , replication_user_name
  , replication_user_password
  , ssl_encryption
  , delay
  , channel_name
);
```

参数

host_name

RDS for MySQL 源数据库实例的主机名或 IP 地址。

host_port

RDS for MySQL 源数据库实例使用的端口。如果网络配置包括转换端口号的安全 Shell (SSH) 端口复制，请指定由 SSH 公开的端口号。

replication_user_name

对 RDS for MySQL 源数据库实例具有 `REPLICATION CLIENT` 和 `REPLICATION SLAVE` 权限的用户的 ID。建议您向专用于复制的账户提供源数据库实例。

replication_user_password

在 `replication_user_name` 中指定的用户 ID 的密码。

ssl_encryption

指定是否在复制连接中使用安全套接字层（SSL）加密的值。1 表示使用 SSL 加密，0 表示不使用加密。默认值为 0。

> **Note**
>
> 不支持 `SOURCE_SSL_VERIFY_SERVER_CERT` 选项。此选项设置为 0，这意味着连接已加密，但未验证证书。

delay

延迟从源数据库实例复制的最小秒数。

该参数的限制为一天（86400 秒）。

channel_name

复制通道的名称。每个复制通道接收来自在特定主机和端口上运行的单个源 RDS for MySQL 数据库实例的二进制日志事件。

使用说明

主用户必须运行 `mysql.rds_set_external_source_with_auto_position_for_channel` 过程。此过程必须在您要创建复制通道的目标 RDS for MySQL 数据库实例上运行。

在运行 `rds_set_external_source_with_auto_position_for_channel` 之前，请为源数据库实例上的复制用户配置多源副本所需的权限。要将多源副本连接到源数据库实例，必须指定对源数据库实例拥有 `REPLICATION CLIENT` 和 `REPLICATION SLAVE` 权限的复制用户的 `replication_user_name` 和 `replication_user_password` 值。

在源数据库实例上配置复制用户

1. 通过使用所选的 MySQL 客户端，连接到源数据库实例并创建要用于复制的用户账户。示例如下：

> **Important**
>
> 作为安全最佳实践，请指定除以下示例中所示占位符值以外的密码。

```
CREATE USER 'repl_user'@'example.com' IDENTIFIED BY 'password';
```

2. 在源数据库实例上，向复制用户授予 `REPLICATION CLIENT` 和 `REPLICATION SLAVE` 权限。以下示例向您所在域的“repl_user”用户授予对所有数据库的 `REPLICATION CLIENT` 和 `REPLICATION SLAVE` 权限。

```
GRANT REPLICATION CLIENT, REPLICATION SLAVE ON *.* TO 'repl_user'@'example.com';
```

要使用加密复制，请将源数据库实例配置为使用 SSL 连接。

在调用 `mysql.rds_set_external_source_with_auto_position_for_channel` 之前，务必调用 the section called “mysql.rds_set_external_source_gtid_purged”，以便使用来自外部源的指定 GTID 范围设置 `gtid_purged` 系统变量。

在调用 `mysql.rds_set_external_source_with_auto_position_for_channel` 以将 Amazon RDS 数据库实例配置为特定通道上的只读副本后，可以在只读副本上调用 the section called “mysql.rds_start_replication_for_channel” 以启动该通道上的复制过程。

在调用 `mysql.rds_set_external_source_with_auto_position_for_channel` 以配置此复制通道后，可以在副本上调用 mysql.rds_start_replication_for_channel 以启动通道上的复制过程。您可以调用 the section called “mysql.rds_reset_external_source_for_channel” 以停止通道上的复制，并从副本中移除通道配置。

示例

在 RDS for MySQL 数据库实例上运行时，以下示例将此数据库实例上名为 `channel_1` 的复制通道配置为从由主机 `sourcedb.example.com` 和端口 `3306` 指定的源复制数据。它将最小复制延迟设置为一小时（3600 秒）。这意味着，来自源 RDS for MySQL 数据库实例的更改在至少一个小时内不会在多源副本上应用。

```
call mysql.rds_set_external_source_with_auto_position_for_channel(
  'sourcedb.example.com',
  3306,
  'repl_user',
  'password',
  1,
  3600,
  'channel_1');
```

mysql.rds_set_external_source_with_delay_for_channel

在 RDS for MySQL 数据库实例上配置一个具有指定复制延迟的复制通道。

> **Important**
>
> 要运行此过程，必须启用 `autocommit`。要启用它，请将 `autocommit` 参数设置为 `1`。有关修改参数的信息，请参阅 在 Amazon RDS 中修改数据库参数组中的参数。

语法

```
CALL mysql.rds_set_external_source_with_delay_for_channel (
  host_name
  , host_port
  , replication_user_name
  , replication_user_password
  , mysql_binary_log_file_name
  , mysql_binary_log_file_location
  , ssl_encryption
  , delay
  , channel_name
);
```

参数

host_name

RDS for MySQL 源数据库实例的主机名或 IP 地址。

host_port

RDS for MySQL 源数据库实例使用的端口。如果网络配置包括转换端口号的安全 Shell (SSH) 端口复制，请指定由 SSH 公开的端口号。

replication_user_name

对 RDS for MySQL 源数据库实例具有 `REPLICATION CLIENT` 和 `REPLICATION SLAVE` 权限的用户的 ID。建议您向专用于复制的账户提供源数据库实例。

replication_user_password

在 `replication_user_name` 中指定的用户 ID 的密码。

mysql_binary_log_file_name

源数据库实例上包含复制信息的二进制日志的名称。

mysql_binary_log_file_location

`mysql_binary_log_file_name` 二进制日志中复制将开始读取复制信息的位置。

您可以通过在源数据库实例上运行 `SHOW BINARY LOG STATUS` 来确定二进制日志文件名和位置。

> **Note**
>
> 以前的 MySQL 版本使用的是 `SHOW MASTER STATUS`，而不是 `SHOW BINARY LOG STATUS`。如果您使用的 MySQL 版本低于 8.4，请使用 `SHOW MASTER STATUS`。

ssl_encryption

指定是否在复制连接中使用安全套接字层（SSL）加密的值。1 表示使用 SSL 加密，0 表示不使用加密。默认值为 0。

> **Note**
>
> 不支持 `SOURCE_SSL_VERIFY_SERVER_CERT` 选项。此选项设置为 0，这意味着连接已加密，但未验证证书。

delay

延迟从源数据库实例复制的最小秒数。

该参数的限制为一天（86400 秒）。

channel_name

复制通道的名称。每个复制通道接收来自在特定主机和端口上运行的单个源 RDS for MySQL 数据库实例的二进制日志事件。

使用说明

主用户必须运行 `mysql.rds_set_external_source_with_delay_for_channel` 过程。此过程必须在您要创建复制通道的目标 RDS for MySQL 数据库实例上运行。

在运行 `mysql.rds_set_external_source_with_delay_for_channel` 之前，请为源数据库实例上的复制用户配置多源副本所需的权限。要将多源副本连接到源数据库实例，必须指定对源数据库实例拥有 `REPLICATION CLIENT` 和 `REPLICATION SLAVE` 权限的复制用户的 `replication_user_name` 和 `replication_user_password` 值。

在源数据库实例上配置复制用户

1. 通过使用所选的 MySQL 客户端，连接到源数据库实例并创建要用于复制的用户账户。示例如下：

> **Important**
>
> 作为安全最佳实践，请指定除以下示例中所示占位符值以外的密码。

```
CREATE USER 'repl_user'@'example.com' IDENTIFIED BY 'password';
```

2. 在源数据库实例上，向复制用户授予 REPLICATION CLIENT 和 REPLICATION SLAVE 权限。以下示例向您所在域的“repl_user”用户授予对所有数据库的 REPLICATION CLIENT 和 REPLICATION SLAVE 权限。

```
GRANT REPLICATION CLIENT, REPLICATION SLAVE ON *.* TO 'repl_user'@'example.com';
```

要使用加密复制，请将源数据库实例配置为使用 SSL 连接。

在调用 `mysql.rds_set_external_source_with_delay_for_channel` 以配置此复制通道后，可以在副本上调用 mysql.rds_start_replication_for_channel 以启动通道上的复制过程。您可以调用 the section called “mysql.rds_reset_external_source_for_channel” 以停止通道上的复制，并从副本中移除通道配置。

当您调用 `mysql.rds_set_external_source_with_delay_for_channel` 时，Amazon RDS 将时间、用户和 `set channel source` 的操作记录在 `mysql.rds_history` 表（不包括通道特定的详细信息）和 `mysql.rds_replication_status` 表中，并包含通道名称。此信息仅用于内部使用和监控目的。要记录完整的过程调用以进行审核，请考虑根据应用程序的特定要求启用审核日志或常规日志。

示例

在 RDS for MySQL 数据库实例上运行时，以下示例将此数据库实例上名为 `channel_1` 的复制通道配置为从由主机 `sourcedb.example.com` 和端口 `3306` 指定的源复制数据。它将最小复制延迟设置为一小时（3600 秒）。这意味着，来自源 RDS for MySQL 数据库实例的更改在至少一个小时内不会在多源副本上应用。

```
call mysql.rds_set_external_source_with_delay_for_channel(
  'sourcedb.example.com',
  3306,
  'repl_user',
  'password',
```

```
  'mysql-bin-changelog.000777',
  120,
  1,
  3600,
  'channel_1');
```

mysql.rds_set_source_auto_position_for_channel

将指定通道的复制模式设置为基于二进制日志文件位置或全局事务标识符（GTID）。

语法

```
CALL mysql.rds_set_source_auto_position_for_channel (
auto_position_mode
 , channel_name
);
```

参数

auto_position_mode

该值指示是使用日志文件位置复制还是基于 GTID 的复制：

- `0` – 使用基于二进制日志文件位置的复制方法。默认为 `0`。
- 1 – 使用基于 GTID 的复制方法。

channel_name

多源副本上复制通道的名称。每个复制通道接收来自在特定主机和端口上运行的单个源 RDS for MySQL 数据库实例的二进制日志事件。

使用说明

主用户必须运行 `mysql.rds_set_source_auto_position_for_channel` 过程。此过程在指定的通道上重启复制，以应用指定的自动定位模式。

示例

以下示例将 channel_1 的自动定位模式设置为使用基于 GTID 的复制方法。

```
call mysql.rds_set_source_auto_position_for_channel(1,'channel_1');
```

mysql.rds_set_source_delay_for_channel

为指定的通道设置延迟从源数据库实例复制到多源副本的最小秒数。

语法

```
CALL mysql.rds_set_source_delay_for_channel(delay, channel_name);
```

参数

delay

延迟从源数据库实例进行复制的最小秒数。

该参数的限制为一天（86400 秒）。

channel_name

多源副本上复制通道的名称。每个复制通道接收来自在特定主机和端口上运行的单个源 RDS for MySQL 数据库实例的二进制日志事件。

使用说明

主用户必须运行 `mysql.rds_set_source_delay_for_channel` 过程。要使用此过程，请首先调用 `mysql.rds_stop_replication_for_channel` 以停止复制。然后，调用此过程来设置复制延迟值。设置延迟后，调用 `mysql.rds_start_replication_for_channel` 以重启复制。

示例

以下示例将从多源副本的 `channel_1` 上的源数据库实例进行复制的延迟设置为至少一小时（3600 秒）。

```
CALL mysql.rds_set_source_delay_for_channel(3600,'channel_1');
```

mysql.rds_skip_repl_error_for_channel

跳过二进制日志事件并删除指定通道的 MySQL DB 多源副本上的复制错误。

语法

```
CALL mysql.rds_skip_repl_error_for_channel(channel_name);
```

参数

channel_name

多源副本上复制通道的名称。每个复制通道接收来自在特定主机和端口上运行的单个源 RDS for MySQL 数据库实例的二进制日志事件。

使用说明

主用户必须对只读副本运行 `mysql.rds_skip_repl_error_for_channel` 过程。使用此过程的方式与使用 `mysql.rds_skip_repl_error` 跳过只读副本上的错误的方式类似。有关更多信息，请参阅 调用 mysql.rds_skip_repl_error 过程。

> **Note**
>
> 要跳过基于 GTID 的复制中的错误，建议您改用 the section called “mysql.rds_skip_transaction_with_gtid” 过程。

要确定是否存在错误，请运行 MySQL `SHOW REPLICA STATUS FOR CHANNEL 'channel_name'\G` 命令。如果复制错误不太严重，您可以运行 `mysql.rds_skip_repl_error_for_channel` 以跳过该错误。如果有多个错误，`mysql.rds_skip_repl_error_for_channel` 会删除指定复制通道上的第一个错误，然后警告存在其他错误。然后，您可以使用 `SHOW REPLICA STATUS FOR CHANNEL 'channel_name'\G` 确定要对下一个错误采取的适当操作。有关返回的值的信息，请参阅 MySQL 文档中的 SHOW REPLICA STATUS 语法。

mysql.rds_start_replication_for_channel

在指定的通道上发起从 RDS for MySQL 数据库实例到多源副本的复制。

> **Note**
>
> 您可以使用 mysql.rds_start_replication_until_for_channel 或 mysql.rds_start_replication_until_gtid_for_channel 存储过程从 RDS for MySQL 数据库实例中启动复制，并在指定的二进制日志文件位置停止复制。

语法

```
CALL mysql.rds_start_replication_for_channel(channel_name);
```

参数

channel_name

多源副本上复制通道的名称。每个复制通道接收来自在特定主机和端口上运行的单个源 RDS for MySQL 数据库实例的二进制日志事件。

使用说明

主用户必须运行 `mysql.rds_start_replication_for_channel` 过程。从源 RDS for MySQL 数据库实例导入数据后，在多源副本上运行此命令以在指定的通道上开始复制。

示例

以下示例开始在多源副本的 `channel_1` 上进行复制。

```
CALL mysql.rds_start_replication_for_channel('channel_1');
```

mysql.rds_start_replication_until_for_channel

在指定的通道上从 RDS for MySQL 数据库实例启动复制，并在指定的二进制日志文件位置停止复制。

语法

```
CALL mysql.rds_start_replication_until_for_channel (
replication_log_file
  , replication_stop_point
  , channel_name
);
```

参数

replication_log_file

源数据库实例上包含复制信息的二进制日志的名称。

replication_stop_point

`replication_log_file` 二进制日志中复制将停止的位置。

channel_name

多源副本上复制通道的名称。每个复制通道接收来自在特定主机和端口上运行的单个源 RDS for MySQL 数据库实例的二进制日志事件。

使用说明

主用户必须运行 `mysql.rds_start_replication_until_for_channel` 过程。通过此过程，复制将启动，然后在达到指定的二进制日志文件位置时停止。此过程会同时停止 `SQL_THREAD` 和 `IO_THREAD`。

为 `replication_log_file` 参数指定的文件名必须与源数据库实例二进制日志文件名匹配。

当 `replication_stop_point` 参数指定过去的某个停止位置时，即会立即停止复制。

示例

以下示例在 `channel_1` 上启动复制并复制更改，直到它到达 `mysql-bin-changelog.000777` 二进制日志文件中的位置 `120`。

```
call mysql.rds_start_replication_until_for_channel(
  'mysql-bin-changelog.000777',
  120,
  'channel_1'
  );
```

mysql.rds_start_replication_until_gtid_for_channel

在指定的通道上从 RDS for MySQL 数据库实例中启动复制，并在指定的全局事务标识符（GTID）处停止复制。

语法

```
CALL mysql.rds_start_replication_until_gtid_for_channel(gtid,channel_name);
```

参数

gtid

在其之后停止复制的 GTID。

channel_name

多源副本上复制通道的名称。每个复制通道接收来自在特定主机和端口上运行的单个源 RDS for MySQL 数据库实例的二进制日志事件。

使用说明

主用户必须运行 `mysql.rds_start_replication_until_gtid_for_channel` 过程。该过程在指定的通道上启动复制并应用所有更改，直至指定的 GTID 值。然后，它在该通道上停止复制。

在 `gtid` 参数指定副本已运行的事务时，将会立即停止复制。

在运行此过程之前，必须通过将 `replica_parallel_workers` 或 `slave_parallel_workers` 的值设置为 `0` 来禁用多线程复制。

示例

以下示例在 `channel_1` 上启动复制并复制更改，直至到达 GTID `3E11FA47-71CA-11E1-9E33-C80AA9429562:23`。

```
call mysql.rds_start_replication_until_gtid_for_channel('3E11FA47-71CA-11E1-9E33-
C80AA9429562:23','channel_1');
```

mysql.rds_stop_replication_for_channel

停止在指定的通道上从 MySQL 数据库实例进行复制。

语法

```
CALL mysql.rds_stop_replication_for_channel(channel_name);
```

参数

channel_name

多源副本上复制通道的名称。每个复制通道接收来自在特定主机和端口上运行的单个源 RDS for MySQL 数据库实例的二进制日志事件。

使用说明

主用户必须运行 `mysql.rds_stop_replication_for_channel` 过程。

示例

以下示例停止在多源副本的 `channel_1` 上进行复制。

```
CALL mysql.rds_stop_replication_for_channel('channel_1');
```

使用 GTID 复制事务

以下存储过程控制如何使用 RDS for MySQL 中的全局事务标识符（GTID）复制事务。有关根据 RDS for MySQL 中的 GTID 进行复制的更多信息，请参阅使用基于 GTID 的复制。

在使用这些存储过程管理复制时，如果其复制用户配置了 `caching_sha2_password`，则必须通过指定 `SOURCE_SSL=1` 来配置 TLS。`caching_sha2_password` 是 RDS for MySQL 8.4 的默认身份验证插件。

主题

- mysql.rds_skip_transaction_with_gtid
- mysql.rds_start_replication_until_gtid

mysql.rds_skip_transaction_with_gtid

在 MySQL 数据库实例上跳过复制具有指定全局事务标识符 (GTID) 的事务。

在已知特定 GTID 事务导致问题时，您可以使用该过程进行灾难恢复。请使用该存储过程跳过有问题的事务。有问题的事务示例包括禁用复制、删除重要数据或导致数据库实例变得不可用的事务。

语法

```
CALL mysql.rds_skip_transaction_with_gtid (
gtid_to_skip
);
```

参数

`gtid_to_skip`

要跳过的复制事务的 GTID。

使用说明

主用户必须运行 `mysql.rds_skip_transaction_with_gtid` 过程。

所有 RDS for MySQL 5.7 版本、所有 RDS for MySQL 8.0 版本和所有 RDS for MySQL 8.4 版本均支持此过程。

示例

以下示例将跳过复制具有 GTID 3E11FA47-71CA-11E1-9E33-C80AA9429562:23 的事务。

```
CALL mysql.rds_skip_transaction_with_gtid('3E11FA47-71CA-11E1-9E33-C80AA9429562:23');
```

mysql.rds_start_replication_until_gtid

从 RDS for MySQL 数据库实例中启动复制，并在指定的全局事务标识符（GTID）后面立即停止复制。

语法

```
CALL mysql.rds_start_replication_until_gtid(gtid);
```

参数

gtid

停止复制前的 GTID。

使用说明

主用户必须运行 mysql.rds_start_replication_until_gtid 过程。

所有 RDS for MySQL 5.7 版本、所有 RDS for MySQL 8.0 版本和所有 RDS for MySQL 8.4 版本均支持此过程。

您可以对延迟复制使用该过程以进行灾难恢复。如果您已配置延迟复制，则可以使用该过程来将延迟只读副本的更改向前滚动到灾难之前的时间。在该过程停止复制后，您可以通过使用将只读副本提升为独立的数据库实例中的说明，将只读副本提升为新的主数据库实例。

您可以使用以下存储过程配置延迟复制：

- mysql.rds_set_configuration
- mysql.rds_set_external_master_with_delay（RDS for MariaDB 和 RDS for MySQL 主要版本 8.0 及更低版本）
- mysql.rds_set_external_source_with_delay（RDS for MySQL 主要版本 8.4 及更高版本）
- mysql.rds_set_source_delay

在 gtid 参数指定副本已运行的事务时，将会立即停止复制。

示例

以下示例启动复制并复制更改，直至到达 GTID 3E11FA47-71CA-11E1-9E33-C80AA9429562:23。

```
call mysql.rds_start_replication_until_gtid('3E11FA47-71CA-11E1-9E33-C80AA9429562:23');
```

轮换查询日志

以下存储过程将 MySQL 日志交替到备份表。有关更多信息，请参阅 MySQL 数据库日志文件。

主题

- mysql.rds_rotate_general_log
- mysql.rds_rotate_slow_log

mysql.rds_rotate_general_log

将 mysql.general_log 表轮换到备份表。

语法

```
CALL mysql.rds_rotate_general_log;
```

使用说明

您可以通过调用 mysql.general_log 过程将 mysql.rds_rotate_general_log 表轮换到备份表。轮换日志表时，会将当前日志表复制到备份日志表，随后删除当前日志表中的条目。如果备份日志表已存在，则先将其删除，然后将当前日志表复制到备份。如果需要，您可以查询备份日志表。mysql.general_log 表的备份日志表名为 mysql.general_log_backup。

只有当 log_output 参数设置为 TABLE 时，您才能运行此过程。

mysql.rds_rotate_slow_log

将 mysql.slow_log 表轮换到备份表。

语法

```
CALL mysql.rds_rotate_slow_log;
```

使用说明

您可以通过调用 mysql.slow_log 过程将 mysql.rds_rotate_slow_log 表轮换到备份表。轮换日志表时，会将当前日志表复制到备份日志表，随后删除当前日志表中的条目。如果备份日志表已存在，则先将其删除，然后将当前日志表复制到备份。

如果需要，您可以查询备份日志表。`mysql.slow_log` 表的备份日志表名为 `mysql.slow_log_backup`。

设置和显示二进制日志配置

以下存储过程设置和显示配置参数，例如用于二进制日志文件保留。

主题

- mysql.rds_set_configuration
- mysql.rds_show_configuration

mysql.rds_set_configuration

指定要保留二进制日志的小时数或要延迟复制的秒数。

语法

```
CALL mysql.rds_set_configuration(name,value);
```

参数

name

要设置的配置参数的名称。

#

配置参数的值。

使用说明

`mysql.rds_set_configuration` 过程支持以下配置参数：

- 二进制日志保留小时数
- 源延迟
- target delay

配置参数将永久存储，可在任何数据库实例重启或失效转移后继续使用。

二进制日志保留小时数

`binlog retention hours` 参数用于指定要保留二进制日志文件的小时数。Amazon RDS 通常会尽快清除一个二进制日志，但对于 RDS 外部的 MySQL 数据库的复制，该二进制日志可能仍是必需的。

`binlog retention hours` 的默认值为 NULL。对于 RDS for MySQL，NULL 表示不保留二进制日志（0 小时）。

要指定在数据库实例上保留二进制日志的小时数，请使用 `mysql.rds_set_configuration` 存储过程并指定足以让复制发生的时段，如以下示例中所示。

```
call mysql.rds_set_configuration('binlog retention hours', 24);
```

> **Note**
>
> 不能将值 `0` 用于 `binlog retention hours`。

对于 MySQL 数据库实例，最大 `binlog retention hours` 值为 168 个小时 (7 天)。

在设置保留期后，监视数据库实例的存储用量以确认保留的二进制日志不会占用太多存储空间。

对于多可用区数据库集群部署，您只能从写入器数据库实例中配置二进制日志保留，并且该设置会异步传播到所有读取器数据库实例。如果数据库集群上的二进制日志超过本地总存储空间的一半，Amazon RDS 会自动将陈旧的日志移至 EBS 卷。但是，最新的日志仍保留在本地存储中，因此，如果出现需要更换主机的故障，或者您纵向扩展或缩减数据库，这些日志可能会丢失。

源延迟

使用只读副本中的 `source delay` 参数指定延迟从读取副本复制到其源数据库实例的秒数。Amazon RDS 通常会尽快复制更改，但您可能希望某些环境延迟复制。例如，在延迟复制后，您可以将延迟只读副本向前滚动到发生灾难之前的时间。如果意外删除一个表，您可以使用延迟的复制快速恢复该表。`target delay` 的默认值为 `0`（不延迟复制）。

当您使用此参数时，它会运行 mysql.rds_set_source_delay 并应用 CHANGE primary TO MASTER_DELAY = 输入值。如果成功，该过程将 `source delay` 参数保存到 `mysql.rds_configuration` 表中。

要指定 Amazon RDS 延迟复制到源数据库实例的秒数，请使用 `mysql.rds_set_configuration` 存储过程并指定要延迟复制的秒数。在以下示例中，复制至少延迟一小时（3600 秒）。

```
call mysql.rds_set_configuration('source delay', 3600);
```

然后，该程序运行 `mysql.rds_set_source_delay(3600)`。

`source delay` 参数的限制为一天（86400 秒）。

target delay

使用 `target delay` 参数指定在数据库实例与从该实例创建的任何将来的 RDS 托管式只读副本之间延迟复制的秒数。对于非 RDS 托管式只读副本，将忽略此参数。Amazon RDS 通常会尽快复制更改，但您可能希望某些环境延迟复制。例如，在延迟复制后，您可以将延迟只读副本向前滚动到发生灾难之前的时间。如果意外删除一个表，您可以使用延迟复制快速恢复该表。`target delay` 的默认值为 `0`（不延迟复制）。

对于灾难恢复，您可以将该配置参数与 mysql.rds_start_replication_until 或 mysql.rds_start_replication_until_gtid 存储过程一起使用。要将延迟只读副本的更改向前滚动到发生灾难之前的时间，您可以运行 `mysql.rds_set_configuration` 过程并设置该参数。在 `mysql.rds_start_replication_until` 或 `mysql.rds_start_replication_until_gtid` 过程停止复制后，您可以使用将只读副本提升为独立的数据库实例中的说明将只读副本提升为新的主数据库实例。

要使用 `mysql.rds_rds_start_replication_until_gtid` 过程，必须启用基于 GTID 的复制。要跳过已知会导致灾难的特定基于 GTID 的事务，您可以使用 mysql.rds_skip_transaction_with_gtid 存储过程。有关使用基于 GTID 的复制的更多信息，请参阅使用基于 GTID 的复制。

要指定 Amazon RDS 延迟只读副本的复制的秒数，请使用 `mysql.rds_set_configuration` 存储过程并指定要延迟复制的秒数。以下示例指定至少一个小时（3600 秒）内延迟复制。

```
call mysql.rds_set_configuration('target delay', 3600);
```

`target delay` 参数的限制为一天（86400 秒）。

mysql.rds_show_configuration

保留二进制日志的小时数。

语法

```
CALL mysql.rds_show_configuration;
```

使用说明

要验证 Amazon RDS 保留二进制日志的小时数，请使用 `mysql.rds_show_configuration` 存储过程。

示例

以下示例显示了保留期：

```
call mysql.rds_show_configuration;
                name                         value     description
                binlog retention hours       24        binlog retention hours specifies
 the duration in hours before binary logs are automatically deleted.
```

预热 InnoDB 缓存

以下存储过程在 RDS for MySQL 数据库实例上保存、加载或取消加载 InnoDB 缓冲池。有关更多信息，请参阅 Amazon RDS 上 MySQL 的 InnoDB 缓存预热。

主题

- mysql.rds_innodb_buffer_pool_dump_now
- mysql.rds_innodb_buffer_pool_load_abort
- mysql.rds_innodb_buffer_pool_load_now

mysql.rds_innodb_buffer_pool_dump_now

将缓冲池的当前状态转储到磁盘。

语法

```
CALL mysql.rds_innodb_buffer_pool_dump_now();
```

使用说明

主用户必须运行 `mysql.rds_innodb_buffer_pool_dump_now` 过程。

mysql.rds_innodb_buffer_pool_load_abort

在运行期间取消加载保存的缓冲池状态。

语法

```
CALL mysql.rds_innodb_buffer_pool_load_abort();
```

使用说明

主用户必须运行 `mysql.rds_innodb_buffer_pool_load_abort` 过程。

mysql.rds_innodb_buffer_pool_load_now

从磁盘加载保存的缓冲池状态。

语法

```
CALL mysql.rds_innodb_buffer_pool_load_now();
```

使用说明

主用户必须运行 `mysql.rds_innodb_buffer_pool_load_now` 过程。

Amazon RDS for Oracle

Amazon RDS 支持运行以下版本的 Oracle Database 的数据库实例：

- Oracle Database 21c (21.0.0.0)
- Oracle Database 19c (19.0.0.0)

> **Note**
>
> Oracle Database 11g、Oracle Database 12c 和 Oracle Database 18c 是旧版本，在 Amazon RDS 中不再受支持。

在创建数据库实例之前，需要完成本指南的设置 Amazon RDS 环境部分中的步骤。当您使用主账户创建数据库实例时，该账户将获得 DBA 权限，但有一些限制。此账户用于管理任务，如创建其他数据库账户。您不能使用 SYS、SYSTEM 或其他 Oracle 提供的管理账户。

您可执行以下操作：

- 数据库实例
- 数据库快照
- 时间点恢复
- 自动备份
- 手动备份

您可以在 VPC 内使用运行 Oracle 数据库的数据库实例。还可以通过启用各种选项（Oracle Spatial 或 Oracle Statspack）向数据库实例添加功能。Amazon RDS 支持对 Oracle 的多可用区部署，这是一种高可用性故障转移解决方案。

> **Important**
>
> 为了提供托管服务体验，Amazon RDS 未提供数据库实例的 Shell 访问权限。并且还限制了对需要高级权限的某些系统程序和表的访问。您可以使用标准 SQL 客户端（例如 Oracle SQL*Plus）访问您的数据库。但是，您无法使用远程登录或安全外壳 (SSH) 直接访问主机。

主题

Oracle on Amazon RDS 概述

您可以阅读以下部分以获取 RDS for Oracle 的概述。

主题

RDS for Oracle 的功能

Amazon RDS for Oracle 支持 Oracle Database 的大部分特性和功能。有些功能可能仅提供有限的支持或受限制的权限。有些功能仅在企业版中可用，有些功能需要额外许可证。有关特定 Oracle 数据库版本所具备的 Oracle 数据库功能的更多信息，请参阅您所用版本的 Oracle 数据库许可信息用户手册。

主题

RDS for Oracle 中的新功能

要查看 RDS for Oracle 中的新增功能，请使用以下方法：

- 在 文档历史记录 中搜索关键字 **Oracle**。
- 在数据库的新增功能？页面上筛选新的 Amazon RDS 功能。对于产品，请选择 Amazon RDS。然后搜索 **Oracle *YYYY***，其中 *YYYYYY* 代表年份，例如 **2025**。

RDS for Oracle 中支持的功能

Amazon RDS for Oracle 支持以下 Oracle Database 功能：

> Note
>
> 以下列表并非详尽内容。

- 高级压缩
- Oracle Application Express (APEX)

  有关更多信息，请参阅 Oracle Application Express (APEX)。
- 自动内存管理
- 自动撤消管理
- Automatic Workload Repository (AWR)

  有关更多信息，请参阅“使用 Automatic Workload Repository (AWR) 生成性能报告”。

- 活动 Data Guard 在相同 Amazon 区域或各个 Amazon 区域内具有最佳性能

  有关更多信息，请参阅 使用 Amazon RDS for Oracle 的只读副本。
- 区块链表（Oracle Database 21c 及更高版本）

  有关更多信息，请参阅 Oracle Database 文档中的管理区块链表。
- 持续查询通知

  有关更多信息，请参阅 Oracle 文档中的使用连续查询通知 (CQN)。
- 数据保留
- 持续查询通知

  有关详细信息，请参阅 Oracle 文档中的数据库更改通知 。
- 内存数据库
- 分布式查询和事务
- 基于版本的重新定义

  有关更多信息，请参阅 设置数据库实例的默认版本。
- EM Express（12c 及更高版本）

  有关更多信息，请参阅 Oracle Enterprise Manager。
- 细粒度审计
- 闪回表、闪回查询和闪回事务查询
- 逐步切换应用程序的密码（Oracle Database 21c 及更高版本）

  有关更多信息，请参阅 Oracle Database 文档中的在数据库中管理应用程序的逐步密码切换。
- 标准大页

  有关更多信息，请参阅"为 RDS for Oracle 实例开启大页"。
- 导入/导出（遗留和数据转储）和 SQL*Loader

  有关更多信息，请参阅"将数据导入到 Amazon RDS 上的 Oracle"。
- Java 虚拟机 (JVM)

  有关更多信息，请参阅 Oracle Java 虚拟机。
- JavaScript（Oracle Database 21c 及更高版本）

有关更多信息，请参阅 Oracle Database 文档中的 DBMS_MLE。

- Label Security

  有关更多信息，请参阅 Oracle Label Security。
- Locator

  有关更多信息，请参阅“Oracle Locator”。
- 具体化视图
- 多租户

  所有 Oracle Database 19c 及更高版本都支持 Oracle 多租户架构。有关更多信息，请参阅 在 RDS for Oracle 中使用 CDB。
- 网络加密

  有关更多信息，请参阅“Oracle 本机网络加密”和“Oracle 安全套接字层”。
- 分区
- 真实应用程序测试

  要使用完整的捕获和重播功能，您必须使用 Amazon Elastic File System（Amazon EFS）来访问 Oracle Real Application Testing 生成的文件。有关更多信息，请参阅 Amazon EFS 集成 以及博文 Use Oracle Real Application Testing features with Amazon RDS for Oracle。
- 应用程序级分片（但不是 Oracle 分片功能）
- 空间和图形

  有关更多信息，请参阅“Oracle Spatial”。
- 星形查询优化
- 流和高级队列
- 摘要管理 – 具体化视图查询重写
- 文本（文件和 URL 数据存储类型不受支持）
- 全面调用
- 透明数据加密 (TDE)

  有关更多信息，请参阅 Oracle 透明数据加密。
- 统一审计，混合模式

有关更多信息，请参阅 Oracle 文档中的混合模式审计。

- XML DB (无 XML DB Protocol Server)

  有关更多信息，请参阅“Oracle XML DB”。
- 虚拟专用数据库

RDS for Oracle 中不支持的功能

Amazon RDS for Oracle 不支持以下 Oracle Database 功能：

> **Note**
>
> 以下列表并非详尽内容。

- 自动存储管理 (ASM)
- 数据库文件库
- 闪回数据库

  > **Note**
  >
  > 有关替代解决方案，请参阅 Amazon 数据库博客文章 Amazon RDS for Oracle 中 Oracle 闪回数据库功能的替代方案。
- FTP 和 SFTP
- 混合分区表
- 消息网关
- Oracle Enterprise Manager 云控制管理存储库
- 真实应用程序集群 (RAC)
- 真正的应用程序安全性（RAS）
- 统一审计，纯净模式
- 工作空间管理器 (WMSYS) 架构

> **Warning**
>
> 通常，Amazon RDS 不会阻止您为不受支持的功能创建架构。但是，如果您为需要 SYSDBA 权限的 Oracle 功能和组件创建架构，则可能会损坏数据字典并影响数据库实例的可用性。仅使用 向 Oracle 数据库实例添加选项 中可用的受支持的功能和架构。

RDS for Oracle 版本

RDS for Oracle 支持多个 Oracle Database 版本。

> **Note**
>
> 有关升级版本的信息，请参阅升级 RDS for Oracle 数据库引擎。

主题

- 适用于 Amazon RDS 的 Oracle Database 21c
- 用于 Amazon RDS 的 Oracle Database 19c

适用于 Amazon RDS 的 Oracle Database 21c

Amazon RDS 支持 Oracle Database 21c，其中包括 Oracle 企业版和 Oracle 标准版 2。与上一版相比，Oracle Database 21c (21.0.0.0) 包含多项新功能和更新。一个关键的变化是，Oracle Database 21c 只支持多租户架构：您不能再将数据库创建为传统的非 CDB。要了解有关 CDB 和非 CDB 之间差异的更多信息，请参阅RDS for Oracle CDB 的限制。

在这一部分中，您可以找到对在 Amazon RDS 上使用 Oracle Database 21c (21.0.0.0) 十分重要的功能和更改。有关这些更改的完整列表，请参阅 Oracle Database 21c 文档。有关每个 Oracle Database 21c 版本支持的功能的完整列表，请参阅 Oracle 文档中的 Oracle Database 产品允许的功能、选件和管理包。

Oracle Database 21c (21.0.0.0) 的 Amazon RDS 参数更改

Oracle Database 21c (21.0.0.0) 包含一些新参数以及具有新范围和新默认值的旧参数。

主题

- 新参数
- 对兼容参数的更改
- 删除了参数

新参数

下表显示 Oracle Database 21c (21.0.0.0) 中新的 Amazon RDS 参数。

名称	值范围	默认值	可修改	描述
blockchain_table_max_no_drop	NONE | 0	NONE	Y	允许您控制创建区块链表时可以指定的最长空闲时间。
dbnest_enable	NONE | CDB_RESOURCE_PDB_ALL	NONE	否	允许您启用或禁用 dbNest。DbNest 为 PDB 提供操作系统资源隔离和管理、文件系统隔离和安全计算。
dbnest_pdb_fs_conf	NONE | *pathname*	NONE	否	指定 PDB 的 dbNest 文件系统配置文件。
diagnostics_control	ERROR | WARNING | IGNORE	IGNORE	Y	允许您控制和监控执行潜在不安全的数据库诊断操作的用户。
drcp_dedicated_opt	YES | NO	YES	Y	启用或禁用对数据库驻留连接池 (DRCP) 的专用优化。
enable_per_pdb_drcp	true | false	true	否	控制数据库驻留连接池 (DRCP) 是为整个 CDB 配置一个连接池，还是为每个 PDB 配置一个隔离连接池。
inmemory_deep_vectorization	true | false	true	Y	启用或禁用深度矢量化框架。

名称	值范围	默认值	可修改	描述
mandatory_user_profile	*profile_name*	不适用	否	指定 CDB 或 PDB 的强制用户配置文件。
optimizer_capture_sql_quarantine	true | false	false	Y	启用或禁用深度矢量化框架。
optimizer_use_sql_quarantine	true | false	false	Y	启用或禁用自动创建 SQL 隔离区配置。
result_cache_execution_threshold	0 到 68719476736	2	Y	指定 PL/SQL 函数在其结果存储到结果缓存中之前可以执行的最大次数。
result_cache_max_temp_result	0 到 100	5	Y	指定任何单个缓存查询结果可以使用的 RESULT_CACHE_MAX_TEMP_SIZE 百分比。
result_cache_max_temp_size	0 到 2199023255552	RESULT_CACHE_SIZE * 10	Y	指定结果缓存可以使用的最大临时表空间量（以字节为单位）。
sga_min_size	0 到 2199023255552 （最大值为 sga_target 的 50%）	0	Y	表示插入式数据库 (PDB) 的 SGA 使用量可能具有的最低值。

名称	值范围	默认值	可修改	描述
tablespace_encryption_default_algorithm	GOST256 | SEED128 | ARIA256 | ARIA192 | ARIA128 | 3DES168 | AES256 | AES192 | AES128	AES128	Y	指定在加密表空间时数据库使用的默认算法。

对兼容参数的更改

对于 Amazon RDS 上的 Oracle Database 21c (21.0.0.0)，`compatible` 参数具有新的最大值。下表显示新的默认值。

参数名称	Oracle Database 21c (21.0.0.0) 最大值
compatible	21.0.0

删除了参数

Oracle Database 21c (21.0.0.0) 中删除了以下参数：

- `remote_os_authent`
- `sec_case_sensitive_logon`
- `unified_audit_sga_queue_size`

用于 Amazon RDS 的 Oracle Database 19c

Amazon RDS 支持 Oracle Database 19c，它包括 Oracle 企业版和 Oracle 标准版 2。

与上一版相比，Oracle Database 19c (19.0.0.0) 包含多项新功能和更新。在本部分中，您可以找到对 Amazon RDS 使用 Oracle Database 19c (19.0.0.0) 十分重要的功能和更改。有关这些更改的完整列

表，请参阅 Oracle Database 19c 文档。有关每个 Oracle Database 19c 版本支持的功能的完整列表，请参阅 Oracle 文档中的 Oracle Database 产品允许的功能、选件和管理包。

Oracle Database 19c (19.0.0.0) 的 Amazon RDS 参数更改

Oracle Database 19c (19.0.0.0) 包含一些新参数以及具有新范围和新默认值的旧参数。

主题

- 新参数
- 对兼容参数的更改
- 删除了参数

新参数

下表显示 Oracle Database 19c (19.0.0.0) 中新的 Amazon RDS 参数。

名称	值	可修改	描述
lob_signature_enable	TRUE、FALSE（默认值）	是	启用或禁用 LOB 定位器签名功能。
max_datapump_parallel_per_job	1 到 1024，或 AUTO	是	指定每个 Oracle Data Pump 作业允许的最大并行进程数。

对兼容参数的更改

对于 Amazon RDS 上的 Oracle Database 19c (19.0.0.0)，`compatible` 参数具有新的最大值。下表显示新的默认值。

参数名称	Oracle Database 19c (19.0.0.0) 最大值
compatible	19.0.0

删除了参数

Oracle Database 19c (19.0.0.0) 中删除了以下参数：

- exafusion_enabled
- max_connections
- o7_dictionary_access

RDS for Oracle 许可选项

Amazon RDS for Oracle 具有两种许可选项：附带许可 (LI) 和自带许可 (BYOL)。在 Amazon RDS 上创建 Oracle 数据库实例后，您可以通过修改数据库实例来更改许可模型。有关更多信息，请参阅 修改 Amazon RDS 数据库实例。

> **Important**
>
> 确保您拥有与数据库实例类和 Oracle Database 版本对应的 Oracle Database 许可证以及软件更新许可证和支持。此外，请确保您对任何单独许可的 Oracle Database 功能拥有许可证。

主题

SE2 附带许可模式

在附带许可模式下，您不需要单独购买 Oracle Database 许可证；Amazon 持有 Oracle 数据库软件的许可证。仅在 Amazon RDS for Oracle Database 标准版 2（SE2）上支持附带许可证模式。

在此模式下，如果您拥有可享受案例支持的Amazon Web Services 支持账户，请联系 Amazon Web Services 支持 解决 Amazon RDS 和 Oracle 数据库服务请求。您对 RDS for Oracle 的 LI 选项的使用受 Amazon 服务条款的第 10.3.1 节限制。

EE 和 SE2 自带许可（BYOL）

在 BYOL 模式下，可以使用您现有的 Oracle Database 许可证在 Amazon RDS 上部署数据库。Amazon RDS 仅对于 Oracle Database 企业版（EE）和 Oracle Database 标准版 2（SE2）支持 BYOL 模式。

确保您拥有与数据库实例类和要运行的 Oracle Database 版本对应的 Oracle Database 许可证（及软件更新许可证和支持）。您也必须遵循 Oracle 对云计算环境中 Oracle Database 软件的授权政策。有关 Oracle 对 Amazon EC2 的授权政策的更多信息，请参阅在云计算环境中授权 Oracle 软件。

在此模式下，可以继续使用有效的 Oracle 支持账户，并能直接联系 Oracle 解决 Oracle 数据库服务请求。如果您拥有可享受案例支持的Amazon Web Services 支持账户，可以联系 Amazon Web Services 支持 了解 Amazon RDS 问题。Amazon Web Services 和 Oracle 为需要两个组织提供帮助的案例提供多供应商支持流程。

与 Amazon License Manager 集成

为了更轻松地监控 BYOL 模型中的 Oracle 许可证使用情况，Amazon License Manager 需与 Amazon RDS for Oracle 集成。License Manager 基于虚拟核心（vCPU）对 RDS for Oracle 引擎版本和许可包的跟踪提供支持。您也可以将 License Manager 与 Amazon Organizations 一起使用以集中管理组织的所有账户。

下表显示了 RDS for Oracle 的产品信息筛选器。

筛选	名称	描述
引擎版本	`oracle-ee`	Oracle Database 企业版 (EE)
	`oracle-se2`	Oracle Database 标准版 2（SE2）
许可证包	`data guard`	请参阅 使用 Amazon RDS for Oracle 的只读副本 (Oracle Active Data Guard)
	`olap`	请参阅 Oracle OLAP
	`ols`	请参阅 Oracle Label Security
	`diagnostic pack sqlt`	请参阅 Oracle SQLT
	`tuning pack sqlt`	请参阅 Oracle SQLT

要跟踪 Oracle 数据库实例的许可证使用情况，可以使用 Amazon License Manager 创建自行管理的许可证。在这种情况下，与产品信息筛选条件匹配的 RDS for Oracle 资源会自动与自管理许可证相关联。Oracle 数据库实例的发现操作可能需要长达 24 小时。也可以使用 Amazon Resource Access Manager 跨账户跟踪许可证。

控制台

在 Amazon License Manager 中创建自行管理的许可证，来跟踪 RDS for Oracle 数据库实例的许可证使用情况

1. 转到 https://console.aws.amazon.com/license-manager/。
2. 选择创建自管理许可证。

   有关说明，请参阅《Amazon License Manager 用户指南》中的 Create a self-managed license。

   在 Product Information (产品信息) 面板中为 RDS Product Information Filter (RDS 产品信息筛选器) 添加规则。

   有关更多信息，请参阅 Amazon License Manager API 参考 中的 产品信息。
3. （仅限跨账户跟踪）借助 Amazon Resource Access Manager，使用任何 Amazon 账户或通过 Amazon Organizations 共享自行管理的许可证。有关更多信息，请参阅共享 Amazon 资源。

Amazon CLI

要使用 Amazon CLI 创建自管理许可证，请调用 create-license-configuration 命令。使用 `--cli-input-json` 或 `--cli-input-yaml` 参数将参数传递给命令。

Example

下面的示例为 Oracle Enterprise Edition 创建自管理许可证。

```
aws license-manager create-license-configuration --cli-input-json file://rds-oracle-ee.json
```

下面是示例中使用的示例 `rds-oracle-ee.json` 文件。

```
{
    "Name": "rds-oracle-ee",
    "Description": "RDS Oracle Enterprise Edition",
    "LicenseCountingType": "vCPU",
    "LicenseCountHardLimit": false,
    "ProductInformationList": [
        {
            "ResourceType": "RDS",
            "ProductInformationFilterList": [
```

```
                {
                    "ProductInformationFilterName": "Engine Edition",
                    "ProductInformationFilterValue": ["oracle-ee"],
                    "ProductInformationFilterComparator": "EQUALS"
                }
            ]
        }
    ]
}
```

有关产品信息的更多信息，请参阅 Amazon License Manager 用户指南 中的资源清单自动发现。

有关 `--cli-input` 参数的更多信息，请参阅 Amazon CLI 用户指南 中的从 JSON 或 YAML 输入文件生成 Amazon CLI 骨架和输入参数。

在 Oracle 数据库版本之间迁移

如果您有适用于您计划运行的数据库实例的版本和类别但未使用的 BYOL Oracle 数据库许可证，则可以从标准版 2（SE2）迁移到企业版（EE）。无法从企业版迁移到其它版本。

更改 Oracle 数据库版本并保留数据

1. 创建数据库实例的快照。

   有关更多信息，请参阅“为 Amazon RDS 的单可用区数据库实例创建数据库快照”。
2. 将快照还原到新的数据库实例，然后选择要使用的 Oracle 数据库版本。

   有关更多信息，请参阅“还原到数据库实例”。
3. （可选）删除旧数据库实例，除非您希望将其保持运行并具有相应的 Oracle Database 许可证。

   有关更多信息，请参阅“删除数据库实例”。

为 Oracle 多可用区部署授予许可

Amazon RDS 支持对 Oracle 的多可用区部署，这是一种高可用性故障转移解决方案。建议对生产工作负载使用多可用区。有关更多信息，请参阅“配置和管理 Amazon RDS 的多可用区部署”。

如果使用自带许可模式，多可用区部署中必须具有主数据库实例和备用数据库实例的许可。

RDS for Oracle 用户和权限

当您创建 Amazon RDS for Oracle 数据库实例时，原定设置主用户拥有该数据库实例的大部分最大用户权限。使用主用户账户执行任何管理任务，例如，在数据库中创建其他用户账户。由于 RDS 是一项托管式服务，不允许您以 SYS 和 SYSTEM 身份登录，因此您没有 SYSDBA 权限。

主题

- Oracle DBA 权限的限制
- 如何管理对于 SYS 对象的权限

Oracle DBA 权限的限制

在数据库中，角色是可授予用户或从用户撤消的特权的集合。Oracle 数据库使用角色来提供安全性。有关更多信息，请参阅 Oracle 数据库文档中的配置权限和角色授权。

预定义角色 DBA 通常拥有对 Oracle 数据库的所有管理权限。创建数据库实例时，您的主用户账户将获得 DBA 权限（但有一些限制）。为了提供托管体验，RDS for Oracle 数据库不为 DBA 角色提供以下权限：

- ALTER DATABASE
- ALTER SYSTEM
- CREATE ANY DIRECTORY
- DROP ANY DIRECTORY
- GRANT ANY PRIVILEGE
- GRANT ANY ROLE

有关 RDS for Oracle 系统权限和角色的更多信息，请参阅主用户账户权限。

如何管理对于 SYS 对象的权限

您可以使用 rdsadmin.rdsadmin_util 包来管理对 SYS 对象的权限。例如，如果您创建数据库用户 myuser，您可以使用 rdsadmin.rdsadmin_util.grant_sys_object 过程向 myuser 授予对 V_$SQLAREA 的 SELECT 权限。有关更多信息，请参阅以下主题：

- 向 SYS 对象授予 SELECT 或 EXECUTE 权限
- 撤销 SYS 对象的 SELECT 或 EXECUTE 权限
- 向非主用户授予权限

RDS for Oracle 数据库实例类

RDS for Oracle 数据库实例的计算和内存容量由其实例类决定。您需要的数据库实例类取决于您的处理能力和内存要求。

支持的 RDS for Oracle 数据库实例类

支持的 RDS for Oracle 实例类是 RDS 数据库实例类的子集。有关 RDS 实例类的完整列表，请参阅 [数据库实例类](#)。

RDS for Oracle 预配置的数据库实例类

对于每个 vCPU 需要额外内存、存储和输入/输出的工作负载，RDS for Oracle 还会为其提供预配置的实例类。这些实例类使用以下命名约定：

```
db.r5b.instance_size.tpcthreads_per_core.memratio
db.r5.instance_size.tpcthreads_per_core.memratio
```

以下是已预配置来获得额外内存的实例类示例：

```
db.r5b.4xlarge.tpc2.mem2x
```

前面的实例类名称的组成部分如下所示：

- `db.r5b.4xlarge` – 实例类的名称。
- `tpc2` – 每个内核的线程数。值为 2 表示多线程处于启用状态。值为 1 表示多线程处于关闭状态。
- `mem2x` – 实例类的附加内存与标准内存的比率。在此示例中，优化提供的内存是标准 db.r5.4xlarge 数据库实例的两倍。

RDS for Oracle 中支持的版本、实例类和许可组合

如果您使用的是 RDS 控制台，则可以通过选择创建数据库并指定其他选项，来了解是否支持特定版本、实例类和许可证组合。在 Amazon CLI 中，您可以运行以下命令：

```
aws rds describe-orderable-db-instance-options --engine engine-type --license-
model license-type
```

下表列出了 RDS for Oracle 支持的所有版本、实例类和许可证类型。有关每种类型的内存属性的信息，请参阅 RDS for Oracle 实例类型。有关定价信息，请参阅 Amazon RDS for Aurora 定价模式。

Oracle 版本	Oracle Database 19c 及更高版本
企业版（EE） 自带许可（BYOL）	标准实例类 db.m7i.large–db.m7i.48xlarge db.m6id.large–db.m6id.32xlarge db.m6i.large – db.m6i.32xlarge db.m5d.large–db.m5d.24xlarge db.m5.large–db.m5.24xlarge 内存优化型实例类 db.r7i.large–db.r7i.48xlarge db.r6id.large–db.r6id.32xlarge db.r6i.large–db.r6i.32xlarge db.r5d.large–db.r5d.24xlarge db.r5b.large–db.r5b.24xlarge db.r5.large–db.r5.24xlarge db.x2iedn.xlarge–db.x2iedn.32xlarge db.x2iezn.2xlarge–db.x2iezn.12xlarge db.x2idn.16xlarge–db.x2idn.32xlarge db.x1e.xlarge–db.x1e.32xlarge db.x1.16xlarge–db.x1.32xlarge[1] db.z1d.large–db.z1d.12xlarge

Oracle 版本	Oracle Database 19c 及更高版本
	内存优化型预配置实例类

Oracle 版本	Oracle Database 19c 及更高版本
	db.r6i.8xlarge.tpc2.mem4x db.r6i.8xlarge.tpc2.mem3x db.r6i.6xlarge.tpc2.mem4x db.r6i.4xlarge.tpc2.mem4x db.r6i.4xlarge.tpc2.mem3x db.r6i.4xlarge.tpc2.mem2x db.r6i.2xlarge.tpc2.mem8x db.r6i.2xlarge.tpc2.mem4x db.r6i.2xlarge.tpc1.mem2x db.r6i.xlarge.tpc2.mem4x db.r6i.xlarge.tpc2.mem2x db.r6i.large.tpc1.mem2x db.r5b.8xlarge.tpc2.mem3x db.r5b.6xlarge.tpc2.mem4x db.r5b.4xlarge.tpc2.mem4x db.r5b.4xlarge.tpc2.mem3x db.r5b.4xlarge.tpc2.mem2x db.r5b.2xlarge.tpc2.mem8x db.r5b.2xlarge.tpc2.mem4x db.r5b.2xlarge.tpc1.mem2x db.r5b.xlarge.tpc2.mem4x

Oracle 版本	Oracle Database 19c 及更高版本
	db.r5b.xlarge.tpc2.mem2x db.r5b.large.tpc1.mem2x db.r5.12xlarge.tpc2.mem2x db.r5.8xlarge.tpc2.mem3x db.r5.6xlarge.tpc2.mem4x db.r5.4xlarge.tpc2.mem4x db.r5.4xlarge.tpc2.mem3x db.r5.4xlarge.tpc2.mem2x db.r5.2xlarge.tpc2.mem8x db.r5.2xlarge.tpc2.mem4x db.r5.2xlarge.tpc1.mem2x db.r5.xlarge.tpc2.mem4x db.r5.xlarge.tpc2.mem2x db.r5.large.tpc1.mem2x
	可突增性能实例类
	db.t3.small–db.t3.2xlarge
标准版 2 (SE2) 自带许可（BYOL）	标准实例类

Oracle 版本	Oracle Database 19c 及更高版本
	db.m7i.large–db.m7i.4xlarge db.m6id.large–db.m6id.4xlarge db.m6i.large–db.m6i.4xlarge db.m5d.large–db.m5d.4xlarge db.m5.large–db.m5.4xlarge
	内存优化型实例类
	db.r7i.large–db.r7i.4xlarge db.r6id.large–db.r6id.4xlarge db.r6i.large–db.r6i.4xlarge db.r5d.large–db.r5d.4xlarge db.r5b.large–db.r5b.4xlarge db.r5.large–db.r5.4xlarge db.x2iedn.xlarge–db.x2iedn.4xlarge db.x2iezn.2xlarge–db.x2iezn.4xlarge db.z1d.large–db.z1d.3xlarge
	内存优化型预配置实例类

Oracle 版本	Oracle Database 19c 及更高版本
	db.r6i.4xlarge.tpc2.mem4x db.r6i.4xlarge.tpc2.mem3x db.r6i.4xlarge.tpc2.mem2x db.r6i.2xlarge.tpc2.mem8x db.r6i.2xlarge.tpc2.mem4x db.r6i.2xlarge.tpc1.mem2x db.r6i.xlarge.tpc2.mem4x db.r6i.xlarge.tpc2.mem2x db.r6i.large.tpc1.mem2x db.r5b.4xlarge.tpc2.mem4x db.r5b.4xlarge.tpc2.mem3x db.r5b.4xlarge.tpc2.mem2x db.r5b.2xlarge.tpc2.mem8x db.r5b.2xlarge.tpc2.mem4x db.r5b.2xlarge.tpc1.mem2x db.r5b.xlarge.tpc2.mem4x db.r5b.xlarge.tpc2.mem2x db.r5b.large.tpc1.mem2x db.r5.4xlarge.tpc2.mem4x db.r5.4xlarge.tpc2.mem3x db.r5.4xlarge.tpc2.mem2x

Oracle 版本	Oracle Database 19c 及更高版本
	db.r5.2xlarge.tpc2.mem8x db.r5.2xlarge.tpc2.mem4x db.r5.2xlarge.tpc1.mem2x db.r5.xlarge.tpc2.mem4x db.r5.xlarge.tpc2.mem2x db.r5.large.tpc1.mem2x
	可突增性能实例类
	db.t3.small–db.t3.2xlarge
标准版 2 (SE2) 附带许可	标准实例类
	db.m5.large–db.m5.4xlarge
	内存优化型实例类
	db.r6i.large–db.r6i.4xlarge db.r5.large–db.r5.4xlarge
	可突增性能实例类
	db.t3.small–db.t3.2xlarge

[1] 您不能再使用 X1.* 实例类系列创建 RDS for Oracle 数据库实例（仍支持 X1e.* 实例类）。如果您当前使用的是 X1.* 类，请尽快切换到新一代实例类。从 2025 年 1 月 22 日起，RDS 将在您定义的维护时段内开始自动升级。在升级期间，RDS 会选择等效的 X2iedn 实例类型并对其进行升级。欲了解更多信息，请参阅 re:Post 文章 [Amazon RDS for Oracle is ending support for X1 Database Instances on January 22, 2025]。

> **Note**
>
> 我们鼓励所有 BYOL 客户查阅其许可协议，以评估 Amazon RDS for Oracle 弃用的影响。有关 RDS for Oracle 支持的数据库实例类别的计算容量的更多信息，请参阅数据库实例类和在 RDS for Oracle 中为数据库实例类配置处理器。

> **Note**
>
> 如果您具有使用已弃用的数据库实例类的数据库实例的数据库快照，则可以在还原数据库快照时选择尚未弃用的数据库实例类。有关更多信息，请参阅 还原到数据库实例。

已弃用 RDS for Oracle 数据库实例类

以下是 RDS for Oracle 已弃用的数据库实例类：

- db.m1、db.m2、db.m3、db.m4
- db.t1、db.t2
- db.r1、db.r2、db.r3、db.r4

上述数据库实例类已由性能更好的数据库实例类取代，后者的价格更低。如果您的数据库实例使用已弃用的数据库实例类，则可以使用以下选项：

- 允许 Amazon RDS 自动修改每个数据库实例，以使用类似的未弃用数据库实例类。有关弃用时间表，请参阅 数据库实例类类型。
- 通过修改数据库实例来自行更改数据库实例类。有关更多信息，请参阅 修改 Amazon RDS 数据库实例。

如果您具有使用已弃用的数据库实例类的数据库实例的数据库快照，则可以在还原数据库快照时选择尚未弃用的数据库实例类。有关更多信息，请参阅 还原到数据库实例。

RDS for Oracle 数据库架构

Oracle 多租户架构（也称为 CDB 架构），使 Oracle 数据库能够作为多租户容器数据库（CDB）使用。CDB 可以包括客户创建的可插拔数据库 (PDB)。非 CDB 是使用传统架构的 Oracle 数据库，不能包含 PDB。有关多租户架构的更多信息，请参阅 Oracle 多租户管理员指南。

对于 Oracle Database 19c 及更高版本，您可以创建使用 CDB 架构的 RDS for Oracle 数据库实例。在 RDS for Oracle 中，PDB 称为租户数据库。客户端应用程序在租户数据库（PDB）级别而非 CDB 级别进行连接。RDS for Oracle 支持 CDB 架构的以下配置：

多租户配置

此 RDS 平台可让 RDS for Oracle CDB 实例包含 1 到 30 个租户数据库，具体取决于数据库版本和任何所需的选件许可证。您可以使用 RDS API 添加、修改和删除租户数据库。RDS for Oracle 中的多租户配置不支持应用程序 PDB 或代理 PDB，它们是特殊类型的 PDB。有关应用程序 PDB 和代理 PDB 的更多信息，请参阅 Oracle Database 文档中的 [Types of PDBs](#)。

> **Note**
>
> Amazon RDS 配置之所以称为“多-租户”而不是“多租户”，是因为它是 RDS 平台的一项功能，而不仅仅是 Oracle 数据库引擎的功能。同样，RDS 术语“租户”是指 RDS 配置中的任何租户，而不仅仅是 Oracle PDB。在 RDS 文档中，未加连字号的术语“Oracle 多租户”仅指 Oracle 数据库 CDB 架构，该架构与本地部署和 RDS 部署均兼容。

单租户配置

此 RDS 平台功能将 RDS for Oracle CDB 实例限制为 1 个租户数据库（PDB）。您无法使用 RDS API 添加更多 PDB。单租户配置与非 CDB 架构使用相同的 RDS API。因此，使用采用单租户配置的 CDB 的体验与使用非 CDB 的体验大致相同。

您可以将使用单租户配置的 CDB 转换为多租户配置，从而允许您将 PDB 添加到 CDB 中。这种架构变更是永久且不可逆的。有关更多信息，请参阅 [将单租户配置转换为多租户配置](#)。

> **Note**
>
> 您无法访问 CDB 本身。

在 Oracle Database 21c 及更高版本中，所有数据库都是 CDB。相比之下，您可以创建一个 Oracle Database 19c 数据库实例作为 CDB 或非 CDB。您无法将非 CDB 升级为 CDB，但可以将 Oracle Database 19c 非 CDB 转换为 CDB，然后对其进行升级。您无法将 CDB 升级为非 CDB。

有关更多信息，请参阅以下资源：

- 在 RDS for Oracle 中使用 CDB
- RDS for Oracle CDB 的限制
- 创建 Amazon RDS 数据库实例

RDS for Oracle 初始化参数

在 Amazon RDS 中，您使用数据库参数组管理参数。使用此组，您可以自定义初始化参数。例如，可以使用 `sort_area_size` 配置排序区域大小。与特定数据库参数组关联的所有 RDS for Oracle 数据库实例都使用相同的参数设置。有关更多信息，请参阅 Amazon RDS 的参数组。

RDS for Oracle 中支持的初始化参数

数据库实例支持的参数取决于 Oracle 数据库版本。要查看特定 Oracle Database 版本支持的初始化参数，可以运行 Amazon CLI 命令 `describe-engine-default-parameters`。例如，要列出 Oracle Database 19c 企业版支持的初始化参数的名称，请运行以下命令（包括示例输出）。

```
aws rds describe-engine-default-parameters \
    --db-parameter-group-family oracle-ee-19 \
    --output json | jq -r '.EngineDefaults.Parameters[].ParameterName'

_add_col_optim_enabled
_adg_parselock_timeout
_allow_insert_with_update_check
_allow_level_without_connect_by
_always_semi_join
_autotask_max_window
_awr_disabled_flush_tables
_awr_mmon_cpuusage
_awr_mmon_deep_purge_all_expired
_b_tree_bitmap_plans
_bct_bitmaps_per_file
_bloom_filter_enabled
_buffered_publisher_flow_control_threshold
_bug29394014_allow_triggers_on_vpd_table
_cleanup_rollback_entries
_client_enable_auto_unregister
_clusterwide_global_transactions
_complex_view_merging
_connect_by_use_union_all
_cost_equality_semi_join
_cursor_features_enabled
```

```
_cursor_obsolete_threshold
_datafile_write_errors_crash_instance
_db_block_buffers
...
```

要描述单个初始化参数，请使用以下命令，并将 sga_max_size 替换为参数的名称（包括示例输出）。

```
aws rds describe-engine-default-parameters \
    --db-parameter-group-family oracle-ee-19 \
    --query 'EngineDefaults.Parameters[?ParameterName==`sga_max_size`]' \
    --output json

[
    {
        "ParameterName": "sga_max_size",
        "Description": "max total SGA size",
        "Source": "engine-default",
        "ApplyType": "static",
        "DataType": "integer",
        "AllowedValues": "0-2199023255552",
        "IsModifiable": true
    }
]
```

要查找 Oracle 数据库初始化参数的一般文档，请参阅 Oracle Database 文档中的 Initialization Parameters。请注意，在 RDS for Oracle 中，参数 ARCHIVE_LAG_TARGET 有特殊注意事项。此参数在指定的时间过去后强制切换联机重做日志。在 RDS for Oracle 中，ARCHIVE_LAG_TARGET 设置为 300，因为恢复点目标（RPO）为 5 分钟。为了实现这一目标，RDS for Oracle 每 5 分钟切换一次在线重做日志，并将其存储在 Amazon S3 存储桶中。

如果联机重做日志切换的频率导致 RDS for Oracle 数据库的性能降级，则可以扩展数据库实例和存储以使用更高的 IOPS 和吞吐量。或者，如果您使用 RDS Custom for Oracle 或在 Amazon EC2 上部署 Oracle 数据库，则可以调整 ARCHIVE_LAG_TARGET 初始化参数的设置。

RDS for Oracle 中的有效参数值

在 RDS for Oracle 中，只有以下字符对参数值有效：

- 字母（A-Z 和 a-z）
- 数字 (0-9)

- 空格（空格、制表符和换行符）
- 以下特殊字符：`_ / . : + = ( ) ' * , % $ -`（连字符）

RDS for Oracle 字符集

RDS for Oracle 支持两种类型的字符集：数据库字符集和国家/地区字符集。

数据库字符集

Oracle 数据库字符集用于 `CHAR`、`VARCHAR2` 和 `CLOB` 数据类型中。数据库还将此字符集用于元数据，如表名、列名和 SQL 语句。Oracle 数据库字符集通常称为数据库字符集。

您可以在创建数据库实例时设置该字符集。创建数据库后，无法更改数据库字符集。

支持的数据库字符集

下表列出了 Amazon RDS 中支持的 Oracle 数据库字符集。您可结合使用本表中的值和 Amazon CLI create-db-instance 命令的 `--character-set-name` 参数，或 Amazon RDS API CreateDBInstance 操作的 `CharacterSetName` 参数。

> **Note**
> CDB 的字符集始终是 AL32UTF8。您只能为 PDB 设置不同的字符集。

值	描述
AL32UTF8	Unicode 5.0 UTF-8 通用字符集 (默认)
AR8ISO8859P6	ISO 8859-6 拉丁语/阿拉伯语
AR8MSWIN1256	Microsoft Windows 代码页 1256 8 位拉丁语/阿拉伯语
BLT8ISO8859P13	ISO 8859-13 波罗的海文
BLT8MSWIN1257	Microsoft Windows 代码页 1257 8 位波罗的海文
CL8ISO8859P5	ISO 88559-5 拉丁语/西里尔语

值	描述
CL8MSWIN1251	Microsoft Windows 代码页 1251 8 位拉丁语/西里尔语
EE8ISO8859P2	ISO 8859-2 东欧国家语言
EL8ISO8859P7	ISO 8859-7 拉丁语/希腊语
EE8MSWIN1250	Microsoft Windows 代码页 1250 8 位东欧国家语言
EL8MSWIN1253	Microsoft Windows 代码页 1253 8 位拉丁语/希腊语
IW8ISO8859P8	ISO 8859-8 拉丁语/希伯来语
IW8MSWIN1255	Microsoft Windows 代码页 1255 8 位拉丁语/希伯来语
JA16EUC	EUC 24 位日语
JA16EUCTILDE	除波浪短划线和波形符至 Unicode 及来自 Unicode 的映射外，与 JA16EUC 相同
JA16SJIS	Shift-JIS 16 位日语
JA16SJISTILDE	除波浪短划线和波形符至 Unicode 及来自 Unicode 的映射外，与 JA16SJIS 相同
KO16MSWIN949	Microsoft Windows 代码页 949 韩语
NE8ISO8859P10	ISO 8859-10 北欧国家语言
NEE8ISO8859P4	ISO 8859-4 北欧和东北欧国家语言
TH8TISASCII	泰国工业标准 620-2533-ASCII 8 位
TR8MSWIN1254	Microsoft Windows 代码页 1254 8 位土耳其语
US7ASCII	ASCII 7 位美国英语

值	描述
UTF8	Unicode 3.0 UTF-8 通用字符集 (符合 CESU-8 标准)
VN8MSWIN1258	Microsoft Windows 代码页 1258 8 位越南语
WE8ISO8859P1	西欧 8 位 ISO 8859 第 1 部分
WE8ISO8859P15	ISO 8859-15 西欧国家语言
WE8ISO8859P9	ISO 8859-9 西欧国家语言和土耳其语
WE8MSWIN1252	Microsoft Windows 代码页 1252 8 位西欧国家语言
ZHS16GBK	GBK 16 位简体中文
ZHT16HKSCS	Microsoft Windows 代码页 950，香港附属字符集 HKSCS-2001。字符集转换基于 3.0。
ZHT16MSWIN950	Microsoft Windows 代码页 950 繁体中文
ZHT32EUC	EUC 32 位繁体中文

NLS_LANG 环境变量

区域设置 是一组满足语言和文化要求的信息，这些信息对应于给定的语言和国家/地区。在客户端环境中设置 NLS_LANG 环境变量，是为 Oracle 指定区域设置行为最简单的方法。该变量可设置客户端应用程序和数据库服务器所使用的语言和地区。它还指示客户端字符集，该字符集对应于客户端应用程序输入或显示的数据的字符集。有关 NLS_LANG 和字符集的更多信息，请参阅 Oracle 文档中的什么是字符集或代码页？。

NLS 初始化参数

您还可在 Amazon RDS 中为 Oracle 数据库实例设置实例级别的以下区域语言支持 (NLS) 初始化参数：

- NLS_DATE_FORMAT
- NLS_LENGTH_SEMANTICS

- NLS_NCHAR_CONV_EXCP
- NLS_TIME_FORMAT
- NLS_TIME_TZ_FORMAT
- NLS_TIMESTAMP_FORMAT
- NLS_TIMESTAMP_TZ_FORMAT

有关修改实例参数的信息，请参阅Amazon RDS 的参数组。

您可以在 SQL 客户端中设置其他 NLS 初始化参数。例如，以下语句在连接到 Oracle 数据库实例的 SQL 客户端中将 NLS_LANGUAGE 初始化参数设置为 GERMAN：

```
ALTER SESSION SET NLS_LANGUAGE=GERMAN;
```

有关使用 SQL 客户端连接到 Oracle 数据库实例的信息，请参阅连接到 Oracle 数据库实例。

国家/地区字符集

国家/地区字符集用于 `NCHAR`、`NVARCHAR2` 和 `NCLOB` 数据类型中。国家/地区字符集通常称为 NCHAR 字符集。与数据库字符集不同，NCHAR 字符集不影响数据库元数据。

NCHAR 字符集支持以下字符集：

- AL16UTF16（默认值）
- UTF8

您可以使用 create-db-instance 命令的 `--nchar-character-set-name` 参数指定值（仅 Amazon CLI 版本 2）。如果您使用 Amazon RDS API，请指定 CreateDBInstance 操作的 `NcharCharacterSetName` 参数。创建数据库后，无法更改国家/地区字符集。

有关 Oracle 数据库中 Unicode 的更多信息，请参阅 Oracle 文档中的使用 Unicode 支持多语言数据库。

RDS for Oracle 的限制

在以下各节中，您可以找到有关使用 RDS for Oracle 的重要限制。有关 CDB 的特定限制，请参阅 RDS for Oracle CDB 的限制。

> **Note**
>
> 此列表并不详尽。

主题

- Amazon RDS 中的 Oracle 文件大小限制
- Oracle 提供的架构的公共同义词
- 不支持的功能的架构
- Oracle DBA 权限的限制
- 弃用 TLS 1.0 和 1.1 传输层安全性

Amazon RDS 中的 Oracle 文件大小限制

RDS for Oracle 数据库实例上单个文件的最大大小为 16TiB。此限制由实例使用的 ext4 文件系统施加。因此，Oracle 大文件数据文件限制为 16TiB。如果您尝试将大文件表空间中的数据文件大小调整为超过限制的值，您会收到类似以下错误。

```
ORA-01237: cannot extend datafile 6
ORA-01110: data file 6: '/rdsdbdata/db/mydir/datafile/myfile.dbf'
ORA-27059: could not reduce file size
Linux-x86_64 Error: 27: File too large
Additional information: 2
```

Oracle 提供的架构的公共同义词

请勿为 Oracle 提供的架构（包括 `SYS`、`SYSTEM` 和 `RDSADMIN`）创建或修改公共同义词。这样做可能会导致核心数据库组件失效并影响数据库实例的可用性。

您可以在自己的架构中创建公共同义词引用对象。

不支持的功能的架构

通常，Amazon RDS 不会阻止您为不受支持的功能创建架构。但是，如果您为需要 SYS 权限的 Oracle 功能和组件创建架构，可能会损坏数据字典并影响实例的可用性。仅使用 向 Oracle 数据库实例添加选项 中可用的受支持的功能和架构。

Oracle DBA 权限的限制

在数据库中，角色是可授予用户或从用户撤消的特权的集合。Oracle 数据库使用角色来提供安全性。

预定义角色 `DBA` 通常拥有对 Oracle 数据库的所有管理权限。创建数据库实例时，您的主用户账户将获得 DBA 权限（但有一些限制）。为了提供托管体验，RDS for Oracle 数据库不为 `DBA` 角色提供以下权限：

- `ALTER DATABASE`
- `ALTER SYSTEM`
- `CREATE ANY DIRECTORY`
- `DROP ANY DIRECTORY`
- `GRANT ANY PRIVILEGE`
- `GRANT ANY ROLE`

使用主用户账户执行管理任务，例如，在数据库中创建其他用户账户。您不能使用 `SYS`、`SYSTEM` 和 Oracle 提供的其他管理账户。

弃用 TLS 1.0 和 1.1 传输层安全性

不建议使用传输层安全性协议 1.0 和 1.1 版（TLS 1.0 和 TLS 1.1）。根据安全最佳实践，Oracle 不再建议使用 TLS 1.0 和 TLS 1.1。为了满足您的安全要求，Oracle RDS 强烈建议您改用 TLS 1.2。

连接到 Oracle 数据库实例

在 Amazon RDS 预调配 Oracle 数据库实例后，您可以使用任何标准 SQL 客户端应用程序登录到数据库实例。由于 RDS 是一项托管式服务，因此您无法以 SYS 或 SYSTEM 身份登录。有关更多信息，请参阅 RDS for Oracle 用户和权限。

在本主题中，您将学习如何使用 Oracle SQL Developer 或 SQL*Plus 连接到 RDS for Oracle 数据库实例。有关指导您完成创建和连接到示例数据库实例的示例，请参阅 创建并连接到 Oracle 数据库实例。

主题

查找 RDS for Oracle 数据库实例的端点

每个 Amazon RDS 数据库实例都有一个终端节点，每个终端节点都包含数据库实例的 DNS 名称和端口号。要使用 SQL 客户端应用程序连接到数据库实例，您需要该数据库实例的 DNS 名称和端口号。

您可使用 Amazon RDS 控制台或 Amazon CLI 查找数据库实例的终端节点。

> **Note**
>
> 如果您使用的是 Kerberos 身份验证，请参阅使用 Kerberos 身份验证连接到 Oracle。

控制台

使用控制台查找终端节点

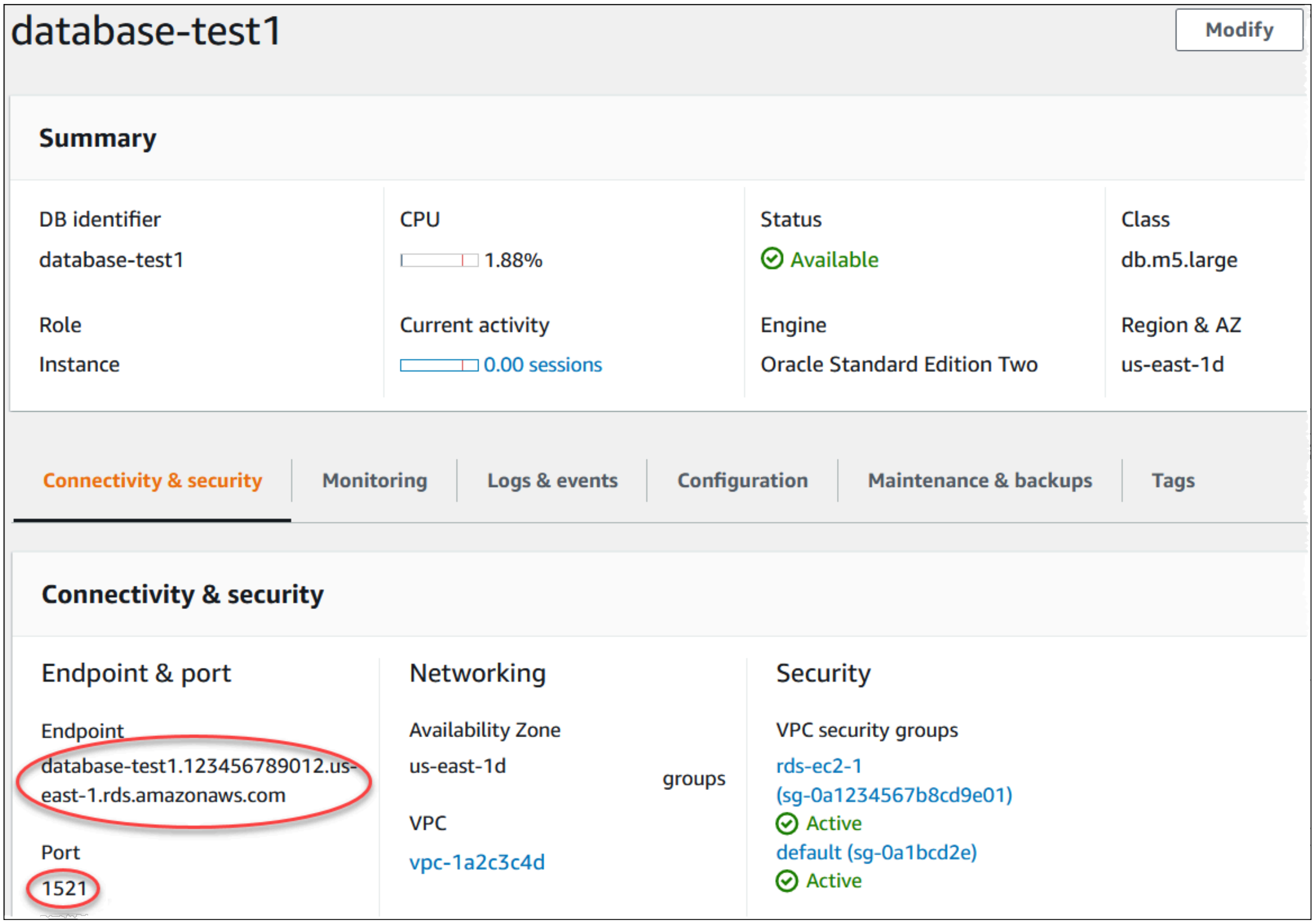
1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在控制台的右上角，选择数据库实例的 Amazon 区域。

3. 找到数据库实例的 DNS 名称和端口号。

   a. 选择 Databases (数据库) 以显示您的数据库实例的列表。

   b. 选择 Oracle 数据库实例名称以显示实例详细信息。

   c. 在连接和安全性选项卡上，复制终端节点。另请注意端口号。您需要终端节点和端口号才能连接到数据库实例。



Amazon CLI

要使用 Amazon CLI 查找 Oracle 数据库实例的终端节点，请调用 describe-db-instances 命令。

Example 使用 Amazon CLI 查找终端节点

```
aws rds describe-db-instances
```

在输出中搜索 `Endpoint` 以查找数据库实例的 DNS 名称和端口号。输出中的 `Address` 行包含 DNS 名称。下面是 JSON 终端节点输出的示例。

```
"Endpoint": {
    "HostedZoneId": "Z1PVIF0B656C1W",
    "Port": 3306,
    "Address": "myinstance.123456789012.us-west-2.rds.amazonaws.com"
},
```

> **Note**
> 该输出可能包含多个数据库实例的信息。

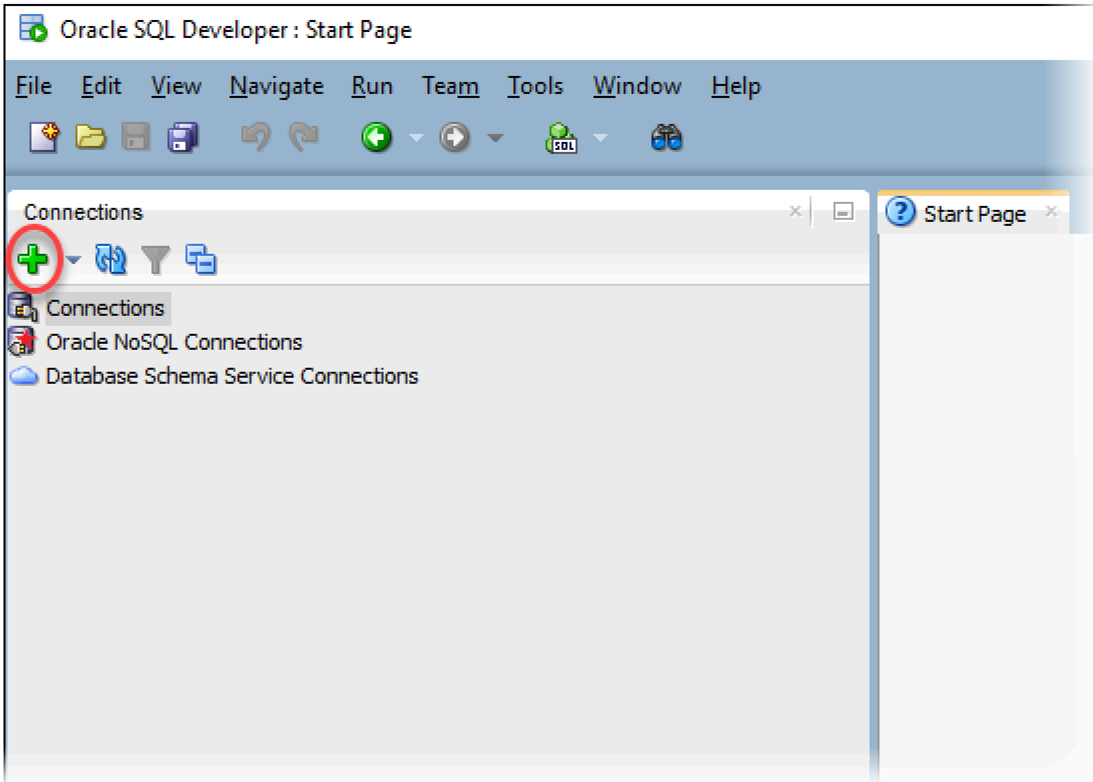
使用 Oracle SQL Developer 连接到数据库实例

在此过程中，您将使用 Oracle SQL Developer 连接到数据库实例。要下载此实用工具的独立版本，请参阅 Oracle SQL 开发人员下载页。

要连接到数据库实例，您需要该实例的 DNS 名称和端口号。有关查找数据库实例的 DNS 名称和端口号的信息，请参阅查找 RDS for Oracle 数据库实例的端点。

使用 SQL Developer 连接到数据库实例

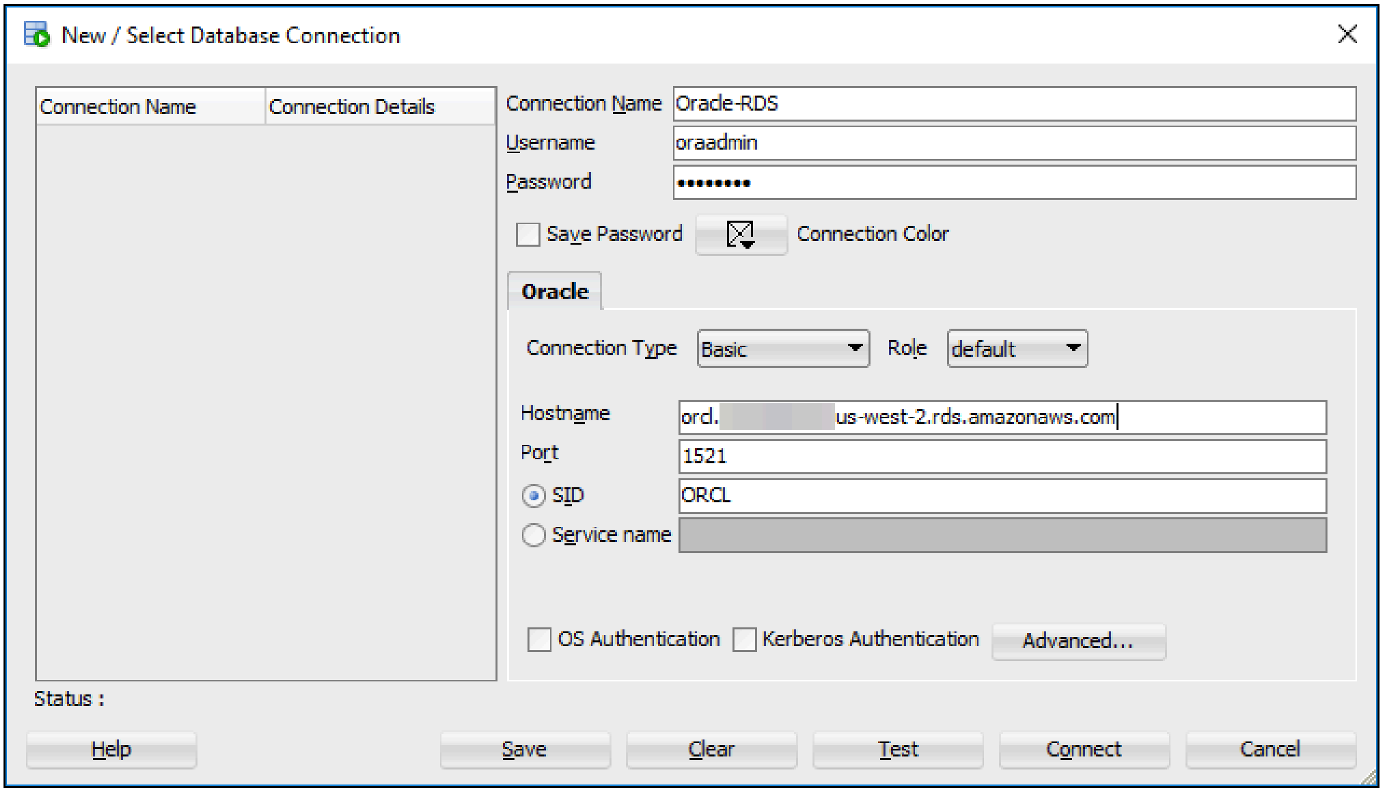
1. 启动 Oracle SQL Developer。
2. 在 Connections 选项卡上，选择 add (+) 图标。



3. 在 New/Select Database Connection 对话框中，提供您的数据库实例的信息：

   - 对于 Connection Name (连接名称)，输入可描述连接的名称，如 `Oracle-RDS`。
   - 对于 Username (用户名)，输入数据库实例的数据库管理员的名称。
   - 对于 Password (密码)，输入数据库管理员的密码。
   - 对于 Hostname (主机名)，输入数据库实例的 DNS 名称。
   - 对于 Port (端口)，输入端口号。
   - 对于 SID，输入数据库名称。您可以在数据库详细信息页面的 Configuration（配置）选项卡上找到数据库名称。

   完成后的对话框应该与下面类似。

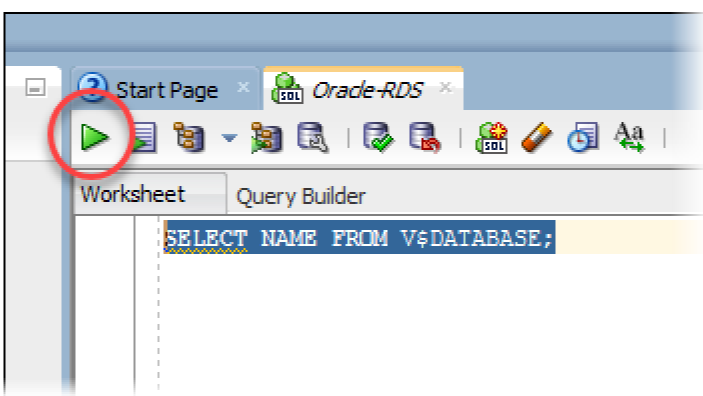


4. 选择 Connect。
5. 您现在可以开始创建您自己的数据库并像往常一样对数据库实例和数据库运行查询。要对数据库实例运行测试查询，请执行以下操作：

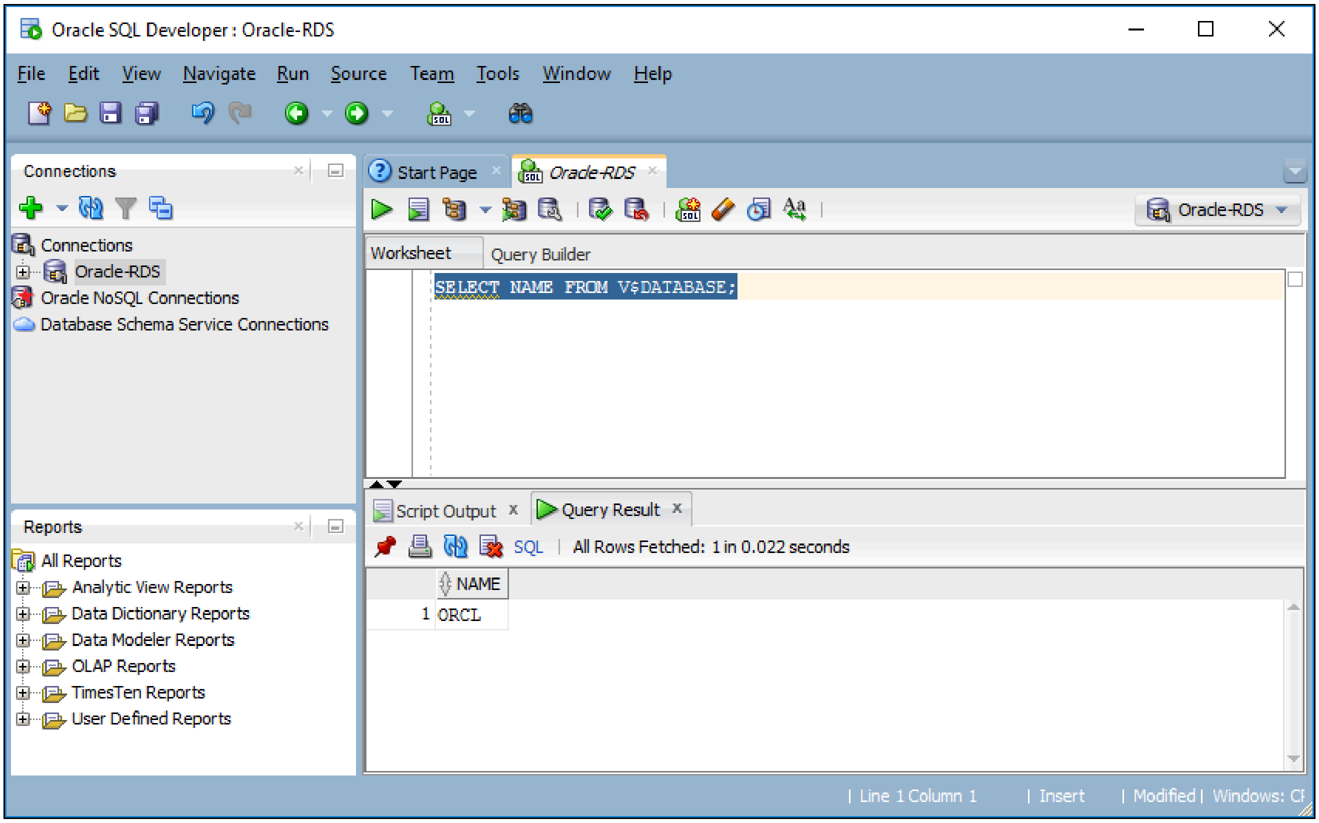
   a. 在您的连接的 Worksheet (工作表) 选项卡中，输入以下 SQL 查询。

   ```
   SELECT NAME FROM V$DATABASE;
   ```

   b. 选择 execute (执行) 图标以运行查询。

   SQL Developer 将返回数据库名称。



使用 SQL*Plus 连接到数据库实例

您可以使用实用程序 (如 SQL*Plus) 连接到运行 Oracle 的 Amazon RDS 数据库实例。要下载包含独立版本 SQL*Plus 的 Oracle 即时客户端，请参阅 Oracle 即时客户端下载。

要连接到数据库实例，您需要该实例的 DNS 名称和端口号。有关查找数据库实例的 DNS 名称和端口号的信息，请参阅查找 RDS for Oracle 数据库实例的端点。

Example 使用 SQL*Plus 连接到 Oracle 数据库实例

在以下示例中，请将用户名替换为您的数据库实例管理员。另外，请将 DNS 名称替换为您的数据库实例，然后包含端口号和 Oracle SID。SID 值是您创建数据库实例时指定的数据库实例数据库的名称，而不是数据库实例的名称。

对于 Linux、macOS 或 Unix：

```
sqlplus 'user_name@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=dns_name)(PORT=port))
(CONNECT_DATA=(SID=database_name)))'
```

对于 Windows：

```
sqlplus user_name@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=dns_name)(PORT=port))
(CONNECT_DATA=(SID=database_name)))
```

您应该可以看到类似于如下所示的输出内容。

```
SQL*Plus: Release 12.1.0.2.0 Production on Mon Aug 21 09:42:20 2017
```

在您输入用户的密码之后，将出现 SQL 提示。

```
SQL>
```

> **Note**
>
> 较短格式的连接字符串（EZ 连接）（例如 `sqlplus USER/PASSWORD@longer-than-63-chars-rds-endpoint-here:1521/database-identifier`）可能会遇到最大字符限制，因此建议您不要使用它进行连接。

安全组注意事项

要连接到数据库实例，它必须与包含必要的 IP 地址和网络配置的安全组关联。您的数据库实例可能使用默认安全组。如果在创建数据库实例时分配了一个默认的未配置安全组，则防火墙将阻止连接。有关创建新安全组的信息，请参阅 使用安全组控制访问权限。

在创建新安全组后，您应修改数据库实例以使其与该安全组关联。有关更多信息，请参阅"修改 Amazon RDS 数据库实例"。

可通过使用 SSL 加密与数据库实例的连接来增强安全性。有关更多信息，请参阅 Oracle 安全套接字层。

进程架构注意事项

服务器进程处理与 Oracle 数据库实例的用户连接。默认情况下，Oracle 数据库实例使用专用服务器进程。使用专用服务器进程，每个服务器进程仅服务一个用户进程。您可以选择配置共享服务器进程。使用共享服务器进程，每个服务器进程可服务多个用户进程。

当大量用户会话占用服务器上的太多内存时，您可以考虑使用共享服务器进程。当会话非常频繁时连接和断开连接，从而导致性能问题时，您也可以考虑使用共享服务器进程。使用共享服务器进程也有缺点。例如，它们可能会耗尽 CPU 资源，并且配置和管理起来更加复杂。

有关专用和共享服务器进程的更多信息，请参阅 Oracle 文档中的关于专用和共享服务器进程。有关在 RDS for Oracle 数据库实例上配置共享服务器进程的更多信息，请参阅知识中心中的如何将 Amazon RDS for Oracle Database 配置为使用共享服务器？。

排除与 Oracle 数据库实例的连接故障

下面是您在尝试连接到您的 Oracle 数据库实例时可能遇到的问题。

问题	故障排除建议
无法连接到数据库实例。	对于新创建的数据库实例，数据库实例的状态为 creating，直到该数据库实例可供使用。当状态变为 Available 时，您可以连接到该数据库实例。根据数据库实例类和存储量，新数据库实例可能需要等待 20 分钟时间才可用。
无法连接到数据库实例。	如果您不能通过在创建数据库实例时指定的端口收发消息，您就无法连接到数据库实例。请联系您的网络管理员，核对您为数据库实例指定的端口是否可以进行入站和出站通信。
无法连接到数据库实例。	由本地防火墙强制执行的访问规则与您在数据库实例的安全组中授权用于访问数据库实例的 IP 地址可能不匹配。此问题很可能是由防火墙上的入站或出站规则所致。 您可以在安全组中添加或编辑入站规则。对于 Source，选择 My IP。这允许从浏览器中检测到的 IP 地址访问数据库实例。有关更多信息，请参阅“Amazon VPC 和 Amazon RDS”。 有关安全组的更多信息，请参阅 使用安全组控制访问权限。 要演练为您的安全组设置规则的过程，请参阅教程：创建 VPC 以用于数据库实例（仅限 IPv4）。
Connect failed because target host or object does not exist – Oracle, Error: ORA-12545	确保您指定了正确的服务器名称和端口号。对于 Server name (服务器名称)，输入来自控制台的 DNS 名称。 有关查找数据库实例的 DNS 名称和端口号的信息，请参阅查找 RDS for Oracle 数据库实例的端点。

问题	故障排除建议
无效用户名称/密码；登录失败 – Oracle, 错误: ORA-01017	您可以访问数据库实例，但连接已被拒绝。这通常是提供的用户名或密码不正确导致的。请验证用户名和密码，然后重试。
TNS:listener 目前不知道连接描述符中给出的 SID – 错误: ORA-12505	确保输入了正确的 SID。SID 与您的数据库名称相同。在您实例的 Databases（数据库）页面的 Configuration（配置）选项卡中找到数据库名称。您还可以使用 Amazon CLI 找到数据库名称： `aws rds describe-db-instances --query 'DBInstances[*].[DBInstanceIdentifier,DBName]' --output text`

有关连接问题的更多信息，请参阅无法连接到 Amazon RDS 数据库实例。

使用 sqlnet.ora 参数修改连接属性

sqlnet.ora 文件包括在 Oracle 数据库服务器和客户端上配置 Oracle Net 功能的参数。使用 sqlnet.ora 文件中的参数，您可以修改传入和传出数据库的连接的属性。

有关可能需要设置 sqlnet.ora 参数的原因的更多信息，请参阅 Oracle 文档中的对配置文件参数进行配置。

设置 sqlnet.ora 参数

Amazon RDS for Oracle 参数组包含一个 sqlnet.ora 参数子集。您可以使用与设置其他 Oracle 参数相同的方法来设置它们。`sqlnetora.` 前缀标识哪些参数是 sqlnet.ora 参数。例如，在 Amazon RDS 的 Oracle 参数组中，`default_sdu_size` sqlnet.ora 参数为 `sqlnetora.default_sdu_size`。

有关管理参数组和设置参数值的信息，请参阅 Amazon RDS 的参数组。

支持的 sqlnet.ora 参数

Amazon RDS 支持以下 sqlnet.ora 参数。对动态 sqlnet.ora 参数的更改立即生效。

参数	有效值	静态/动态	描述
`sqlnetora.default_sdu_size`	512 到 209715	动态	会话数据单位 (SDU) 大小 (字节)。 SDU 是一次性放入缓冲区并通过网络发送的数据量。
`sqlnetora.diag_adr_enabled`	ON, OFF	动态	一个值，用于启用或禁用自动诊断存储库 (ADR) 跟踪。 ON 指定使用 ADR 文件跟踪。 OFF 指定使用非 ADR 文件跟踪。
`sqlnetora.recv_buf_size`	8192 到 268435	动态	用于会话接收操作的缓冲区空间限制，支持 TCP/IP、使用 SSL 的 TCP/IP 以及 SDP 协议。
`sqlnetora.send_buf_size`	8192 到 268435	动态	用于会话发送操作的缓冲区空间限制，支持 TCP/IP、使用 SSL 的 TCP/IP 以及 SDP 协议。
`sqlnetora.sqlnet.allowed_logon_version_client`	8, 10, 11, 12	动态	客户端和充当客户端的服务器与 Oracle 数据库实例建立连接所允许的最低身份验证协议版本。
`sqlnetora.sqlnet.allowed_logon_version_server`	8, 9, 10, 11, 12, 12a	动态	与 Oracle 数据库实例建立连接所允许的最低身份验证协议版本。

参数	有效值	静态/动态	描述
sqlnetora.sqlnet.expire_time	0 到 1440	动态	发送检查以验证客户端/服务器连接是否活动的时间间隔，以分钟为单位。
sqlnetora.sqlnet.inbound_connect_timeout	0 或 10 到 7200	动态	客户端连接到数据库服务器并提供所需身份验证信息的时间，以秒为单位。
sqlnetora.sqlnet.outbound_connect_timeout	0 或 10 到 7200	动态	客户端建立与数据库实例的 Oracle Net 连接的时间，以秒为单位。
sqlnetora.sqlnet.recv_timeout	0 或 10 到 7200	动态	数据库服务器在建立连接之后等待客户端数据的时间，以秒为单位。
sqlnetora.sqlnet.send_timeout	0 或 10 到 7200	动态	数据库服务器在建立连接之后，完成对客户端的发送操作的时间，以秒为单位。
sqlnetora.tcp.connect_timeout	0 或 10 到 7200	动态	客户端建立与数据库服务器的 TCP 连接的时间，以秒为单位。
sqlnetora.trace_level_server	0, 4, 10, 16, OFF, USER, ADMIN, SUPPOF	动态	对于非 ADR 跟踪，在指定级别启用服务器跟踪或禁用服务器跟踪。

每个支持的 sqlnet.ora 参数的默认值是该版本的 Oracle Database 默认值。

查看 sqlnet.ora 参数

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 SQL 客户端查看 sqlnet.ora 参数及其设置。

使用控制台查看 sqlnet.ora 参数

有关查看参数组中参数的信息，请参阅Amazon RDS 的参数组。

在 Oracle 参数组中，`sqlnetora.` 前缀标识哪些参数是 sqlnet.ora 参数。

使用 Amazon CLI 查看 sqlnet.ora 参数

要查看在 Oracle 参数组中配置的 sqlnet.ora 参数，请使用 Amazon CLI describe-db-parameters 命令。

要查看 Oracle 数据库实例的所有 sqlnet.ora 参数，请调用 Amazon CLI download-db-log-file-portion 命令。指定数据库实例标识符、日志文件名和输出的类型。

Example

以下代码列出 `mydbinstance` 的所有 sqlnet.ora 参数。

对于 Linux、macOS 或 Unix：

```
aws rds download-db-log-file-portion \
    --db-instance-identifier mydbinstance \
    --log-file-name trace/sqlnet-parameters \
    --output text
```

对于 Windows：

```
aws rds download-db-log-file-portion ^
    --db-instance-identifier mydbinstance ^
    --log-file-name trace/sqlnet-parameters ^
    --output text
```

使用 SQL 客户端查看 sqlnet.ora 参数

在 SQL 客户端中连接到 Oracle 数据库实例之后，以下查询列出 sqlnet.ora 参数。

```
SELECT * FROM TABLE
```

```
    (rdsadmin.rds_file_util.read_text_file(
         p_directory => 'BDUMP',
         p_filename  => 'sqlnet-parameters'));
```

有关在 SQL 客户端中连接到 Oracle 数据库实例的信息，请参阅连接到 Oracle 数据库实例。

保护 Oracle 数据库实例连接安全

Amazon RDS for Oracle 支持 SSL/TLS 加密连接以及 Oracle 本机网络加密 (NNE) 选项来加密应用程序与 Oracle 数据库实例之间的连接。有关 Oracle Native Network Encryption 选项的更多信息，请参阅Oracle 本机网络加密。

主题

对 RDS for Oracle 数据库实例使用 SSL

安全套接字层 (SSL) 是用于保护客户端和服务器之间的网络连接的行业标准协议。在 SSL 3.0 版之后，此名称已更改为传输层安全性 (TLS)，但我们通常仍将此协议称为 SSL。Amazon RDS 为 Oracle 数据库实例提供 SSL 加密。使用 SSL，您可加密应用程序客户端和 Oracle 数据库实例之间的连接。所有 Amazon 区域均支持对 Oracle 使用 SSL。

通过向与数据库实例关联的选项组添加 Oracle SSL 选项，可为 Oracle 数据库实例提供 SSL 加密。根据 Oracle 的要求，Amazon RDS 使用另一个端口连接 SSL。这样使数据库实例与 Oracle 客户端之间能够同时进行明文通信和 SSL 加密通信。例如，您可以使用用于明文通信的端口与 VPC 内的其他资源通信，同时使用用于 SSL 加密通信的端口与 VPC 外部的资源通信。

有关更多信息，请参阅“Oracle 安全套接字层”。

> **Note**
>
> 您不能在同一数据库实例上同时使用 SSL 和 Oracle 本机网络加密 (NNE)。您必须先禁用任何其他连接加密，然后才能使用 SSL 加密。

更新应用程序以使用新的 SSL/TLS 证书连接到 Oracle 数据库实例

自 2023 年 1 月 13 日起，Amazon RDS 发布了新的证书颁发机构（CA）证书，以便使用安全套接字层或传输层安全性协议（SSL/TLS）连接到 RDS 数据库实例。接下来，您可以找到有关更新应用程序以使用新证书的信息。

本主题可帮助您确定是否有任何客户端应用程序使用 SSL/TLS 连接到您的数据库实例。

> **Important**
>
> 更改 Amazon RDS for Oracle 数据库实例的证书时，仅有数据库侦听器会重新启动。数据库实例不重新启动。现有数据库连接不受影响，但在重新启动侦听器时，新连接将在短时间内遇到错误。
> 我们建议您重新启动 Oracle 数据库，以防止出现任何连接错误。

> **Note**
>
> 对于使用 SSL/TLS 连接到数据库实例的客户端应用程序，您必须更新客户端应用程序信任存储，以包括新的 CA 证书。

更新客户端应用程序信任存储中的 CA 证书后，可以在数据库实例上轮换这些证书。强烈建议在生产环境中实现这些过程之前，先在开发或测试环境中测试它们。

有关证书轮换的更多信息，请参阅轮换 SSL/TLS 证书。有关下载证书的更多信息，请参阅使用 SSL/TLS 加密与数据库实例或集群的连接。有关对 Oracle 数据库实例使用 SSL/TLS 的信息，请参阅Oracle 安全套接字层。

主题

- 确认应用程序是否使用 SSL 进行连接
- 更新应用程序信任存储
- 用于建立 SSL 连接的示例 Java 代码

确认应用程序是否使用 SSL 进行连接

如果 Oracle 数据库实例使用添加了 SSL 选项的选项组，则可能在使用 SSL。按照列出选项组的选项和选项设置中的说明进行检查。有关 SSL 选项的信息，请参阅 Oracle 安全套接字层。

查看侦听器日志以确定是否存在 SSL 连接。下面是侦听器日志中的示例输出。

```
date time * (CONNECT_DATA=(CID=(PROGRAM=program)
(HOST=host)(USER=user))(SID=sid)) *
(ADDRESS=(PROTOCOL=tcps)(HOST=host)(PORT=port)) * establish * ORCL * 0
```

当一个条目的 `PROTOCOL` 具有 `tcps` 值时，这显示 SSL 连接。不过，当 `HOST` 为 `127.0.0.1` 时，您可以忽略该条目。来自 `127.0.0.1` 的连接是数据库实例上的本地管理代理。这些连接不是外部 SSL 连接。因此，如果您看到 `PROTOCOL` 为 `tcps` 而 `HOST` 不是 `127.0.0.1` 的侦听器日志条目，则您的应用程序使用 SSL 进行连接。

要查看侦听器日志，您可以将该日志发布到 Amazon CloudWatch Logs。有关更多信息，请参阅"将 Oracle 日志发布到 Amazon CloudWatch Logs"。

更新应用程序信任存储

您可以更新使用 SQL*Plus 或 JDBC 的应用程序的信任存储以进行 SSL/TLS 连接。

更新 SQL*Plus 应用程序信任存储

您可以更新使用 SQL*Plus 的应用程序的信任存储以进行 SSL/TLS 连接。

> **Note**
>
> 更新信任存储时，除了添加新证书外，还可以保留较旧证书。

更新 SQL*Plus 应用程序信任存储

1. 下载适用于所有 Amazon 区域的新的根证书，并将此文件放在 `ssl_wallet` 目录中。

   有关下载根证书的信息，请参阅 使用 SSL/TLS 加密与数据库实例或集群的连接。

2. 运行以下命令以更新 Oracle wallet。

   ```
   prompt>orapki wallet add -wallet $ORACLE_HOME/ssl_wallet -trusted_cert -cert
         $ORACLE_HOME/ssl_wallet/ssl-cert.pem -auto_login_only
   ```

   将文件名替换为您下载的文件名。

3. 运行以下命令以确认已成功更新 wallet。

```
prompt>orapki wallet display -wallet $ORACLE_HOME/ssl_wallet
```

您的输出应包含以下内容。

```
Trusted Certificates:
Subject: CN=Amazon RDS Root 2019 CA,OU=Amazon RDS,O=Amazon Web Services\,
 Inc.,L=Seattle,ST=Washington,C=US
```

更新 JDBC 应用程序信任存储

您可以更新使用 JDBC 的应用程序的信任存储以进行 SSL/TLS 连接。

有关下载根证书的信息，请参阅 使用 SSL/TLS 加密与数据库实例或集群的连接。

有关导入证书的示例脚本，请参阅 将证书导入信任存储的示例脚本。

用于建立 SSL 连接的示例 Java 代码

以下代码说明如何设置使用 JDBC 的 SSL 连接。

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.util.Properties;

public class OracleSslConnectionTest {
    private static final String DB_SERVER_NAME = "<dns-name-provided-by-amazon-rds>";
    private static final Integer SSL_PORT = "<ssl-option-port-configured-in-option-
group>";
    private static final String DB_SID = "<oracle-sid>";
    private static final String DB_USER = "<user name>";
    private static final String DB_PASSWORD = "<password>";
    // This key store has only the prod root ca.
    private static final String KEY_STORE_FILE_PATH = "<file-path-to-keystore>";
    private static final String KEY_STORE_PASS = "<keystore-password>";

    public static void main(String[] args) throws SQLException {
        final Properties properties = new Properties();
        final String connectionString = String.format(
                "jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCPS)(HOST=%s)(PORT=
%d))(CONNECT_DATA=(SID=%s)))",
```

```
                DB_SERVER_NAME, SSL_PORT, DB_SID);
        properties.put("user", DB_USER);
        properties.put("password", DB_PASSWORD);
        properties.put("oracle.jdbc.J2EE13Compliant", "true");
        properties.put("javax.net.ssl.trustStore", KEY_STORE_FILE_PATH);
        properties.put("javax.net.ssl.trustStoreType", "JKS");
        properties.put("javax.net.ssl.trustStorePassword", KEY_STORE_PASS);
        final Connection connection = DriverManager.getConnection(connectionString,
 properties);
        // If no exception, that means handshake has passed, and an SSL connection can
 be opened
    }
}
```

Important

在确定了数据库连接使用 SSL/TLS 并更新了应用程序信任存储之后，可以更新数据库以使用 rds-ca-rsa2048-g1 证书。有关说明，请参阅通过修改数据库实例或集群来更新 CA 证书中的步骤 3。

对 RDS for Oracle 数据库实例使用原生网络加密

Oracle Database 提供两种通过网络加密数据的方式：原生网络加密（NNE）和传输层安全性协议（TLS）。NNE 是 Oracle 的专有安全功能，而 TLS 是行业标准。RDS for Oracle 对于所有 Oracle Database 版本均支持 NNE。

与 TLS 相比，NNE 具有以下优势：

- 您可以使用 NNE 选项中的设置控制客户端和服务器上的 NNE：
  - `SQLNET.ALLOW_WEAK_CRYPTO_CLIENTS` 和 `SQLNET.ALLOW_WEAK_CRYPTO`
  - `SQLNET.CRYPTO_CHECKSUM_CLIENT` 和 `SQLNET.CRYPTO_CHECKSUM_SERVER`
  - `SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENT` 和 `SQLNET.CRYPTO_CHECKSUM_TYPES_SERVER`
  - `SQLNET.ENCRYPTION_CLIENT` 和 `SQLNET.ENCRYPTION_SERVER`
  - `SQLNET.ENCRYPTION_TYPES_CLIENT` 和 `SQLNET.ENCRYPTION_TYPES_SERVER`
- 在大多数情况下，您不需要配置您的客户端或服务器。相比之下，TLS 要求您同时配置客户端和服务器。

- 不需要证书。在 TLS 中，服务器需要证书（该证书最终会过期），而客户端要求颁发服务器证书的证书颁发机构提供可信根证书。

通过向与数据库实例关联的选项组添加 Oracle NNE 选项，可为 Oracle 数据库实例提供 NNE 加密。有关更多信息，请参阅 Oracle 本机网络加密。

> **Note**
> 您不能在同一个数据库实例上同时使用 NNE 和 TLS。

为 Amazon RDS for Oracle 配置 Kerberos 身份验证

当用户连接到 Amazon RDS for Oracle 数据库实例时，您可以使用 Kerberos 身份验证来验证用户的身份。在此配置中，您的数据库实例可以使用 Amazon Directory Service for Microsoft Active Directory（也称为 Amazon Managed Microsoft AD）。当用户对联接到信任域的 RDS for Oracle 数据库实例进行身份验证时，身份验证请求将转发至您使用 Amazon Directory Service 创建的目录。

将所有凭证保存在同一目录中可以节省您的时间和精力。您有一个集中位置用来存储和管理多个数据库实例的凭证。目录还可以改善您的整体安全配置文件。

区域和版本可用性

功能可用性和支持因每个数据库引擎的特定版本以及 Amazon Web Services 区域而异。有关使用 Kerberos 身份验证的 RDS for Oracle 的版本和区域可用性的更多信息，请参阅支持 Amazon RDS 中 Kerberos 身份验证功能的区域和数据库引擎。

> **Note**
> 对于 RDS for Oracle 数据库实例已弃用的数据库实例类，不支持 Kerberos 身份验证。有关更多信息，请参阅 RDS for Oracle 数据库实例类。

主题

- 为 Oracle 数据库实例设置 Kerberos
- 在域中管理数据库实例
- 使用 Kerberos 身份验证连接到 Oracle

为 Oracle 数据库实例设置 Kerberos

使用 Amazon Directory Service for Microsoft Active Directory（也称为 Amazon Managed Microsoft AD）为 Oracle 数据库实例设置 Kerberos 身份验证。要设置 Kerberos 身份验证，请完成以下步骤：

- 步骤 1：使用 Amazon Managed Microsoft AD 创建目录
- 步骤 2：创建信任
- 步骤 3：配置 Amazon RDS 的 IAM 权限
- 步骤 4：创建和配置用户
- 步骤 5：在目录和数据库实例之间启用跨 VPC 流量
- 步骤 6：创建或修改 Oracle 数据库实例
- 步骤 7：创建 Kerberos 身份验证 Oracle 登录名
- 步骤 8：配置 Oracle 客户端

> **Note**
>
> 在设置过程中，RDS 会创建一个名为 *managed_service_user@example.com* 的 Oracle 数据库用户（其具有 `CREATE SESSION` 权限），其中的 *example.com* 即为域名。此用户对应于 Directory Service 在 Managed Active Directory 中创建的用户。RDS 会定期使用 Directory Service 提供的凭证登录 Oracle 数据库。之后，RDS 会立即销毁票证缓存。

步骤 1：使用 Amazon Managed Microsoft AD 创建目录

Amazon Directory Service 将在 Amazon 云中创建一个完全托管的 Active Directory。创建 Amazon Managed Microsoft AD 目录时，Amazon Directory Service 将代表您创建两个域控制器和域名系统 (DNS) 服务器。目录服务器在 VPC 中的不同子网中创建。这种冗余有助于确保始终可以访问目录，即使发生了故障。

创建 Amazon Managed Microsoft AD 目录时，Amazon Directory Service 代表您执行以下任务：

- 在 VPC 中设置 Active Directory。
- 使用用户名 Admin 和指定密码创建目录管理员账户。您可以使用此账户管理您的目录。

> **Note**
>
> 请务必保存此密码。Amazon Directory Service 不会存储它。您可以重置它，但无法检索它。

- 为目录控制器创建安全组。

在启动 Amazon Managed Microsoft AD 时，Amazon 创建一个组织单位 (OU)，其中包含目录的所有对象。此 OU 具有您在创建目录时键入的 NetBIOS 名称且位于域根目录中。此域根目录由 Amazon 拥有和管理。

使用您的 Amazon Managed Microsoft AD 目录创建的 Admin 账户对您的 OU 的最常见管理活动具有以下权限：

- 创建、更新或删除用户
- 将资源添加到域（如文件或打印服务器），然后为 OU 中的用户分配这些资源的权限
- 创建额外的 OU 和容器
- 委托授权
- 从 Active Directory 回收站还原删除的对象
- 在 Active Directory Web 服务上运行 AD 和 DNS Windows PowerShell 模块

Admin 账户还具有执行下列域范围活动的权限：

- 管理 DNS 配置（添加、删除或更新记录、区域和转发器）
- 查看 DNS 事件日志
- 查看安全事件日志

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 Amazon Directory Service API 创建目录。确保打开目录安全组上的相关出站端口，以便目录可以与 Oracle 数据库实例进行通信。

使用 Amazon Managed Microsoft AD 创建目录

1. 登录 Amazon Web Services Management Console，然后打开 Amazon Directory Service 控制台，网址为：https://console.aws.amazon.com/directoryservicev2/。

2. 在导航窗格中，选择 Directories (目录)，然后选择 Set up Directory (设置目录)。
3. 选择 Amazon Managed Microsoft AD。Amazon Managed Microsoft AD 是当前唯一可以与 Amazon RDS 一起使用的选项。
4. 输入以下信息：

   目录 DNS 名称

      目录的完全限定名称，例如 **`corp.example.com`**。

   目录 NetBIOS 名称

      目录的短名称，如 **`CORP`**。

   目录描述

      （可选）目录的描述。

   管理员密码

      目录管理员的密码。目录创建过程将使用用户名 Admin 和此密码创建一个管理员账户。

      目录管理员密码不能包含单词“admin”。此密码区分大小写，且长度必须介于 8 – 64 个字符之间。至少，它还必须包含下列四种类别中三种类别的一个字符：

      - 小写字母 (a–z)
      - 大写字母 (A–Z)
      - 数字 (0–9)
      - 非字母数字字符 (~!@#$%^&*_-+=`|\(){}[]:;"'<>,.?/)

   确认密码

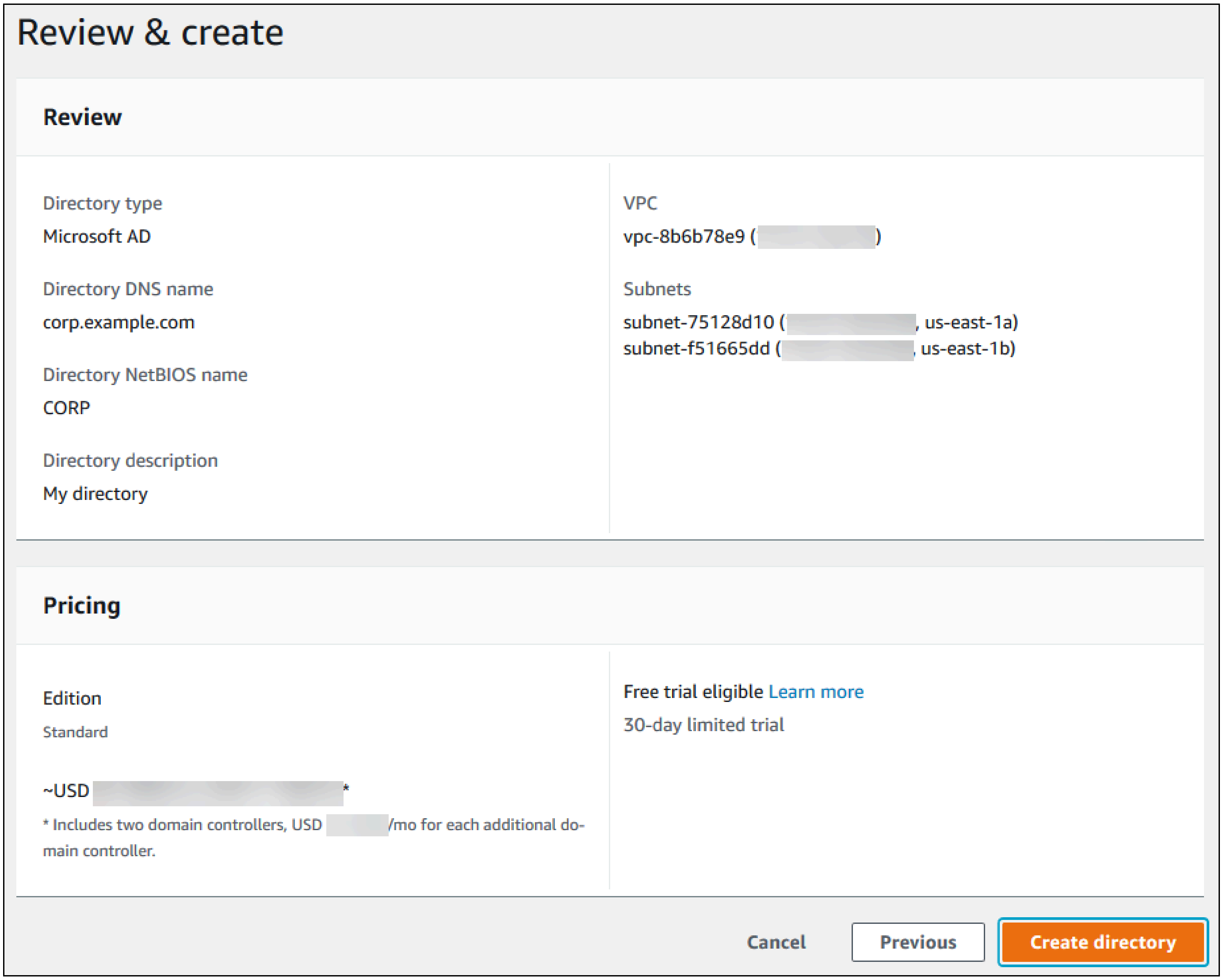
      已重新键入管理员密码。
5. 选择下一步。
6. 在 Networking (网络) 部分中输入以下信息，然后选择 Next (下一步)：

   VPC

      目录的 VPC。在同一 VPC 中创建 Oracle 数据库实例。

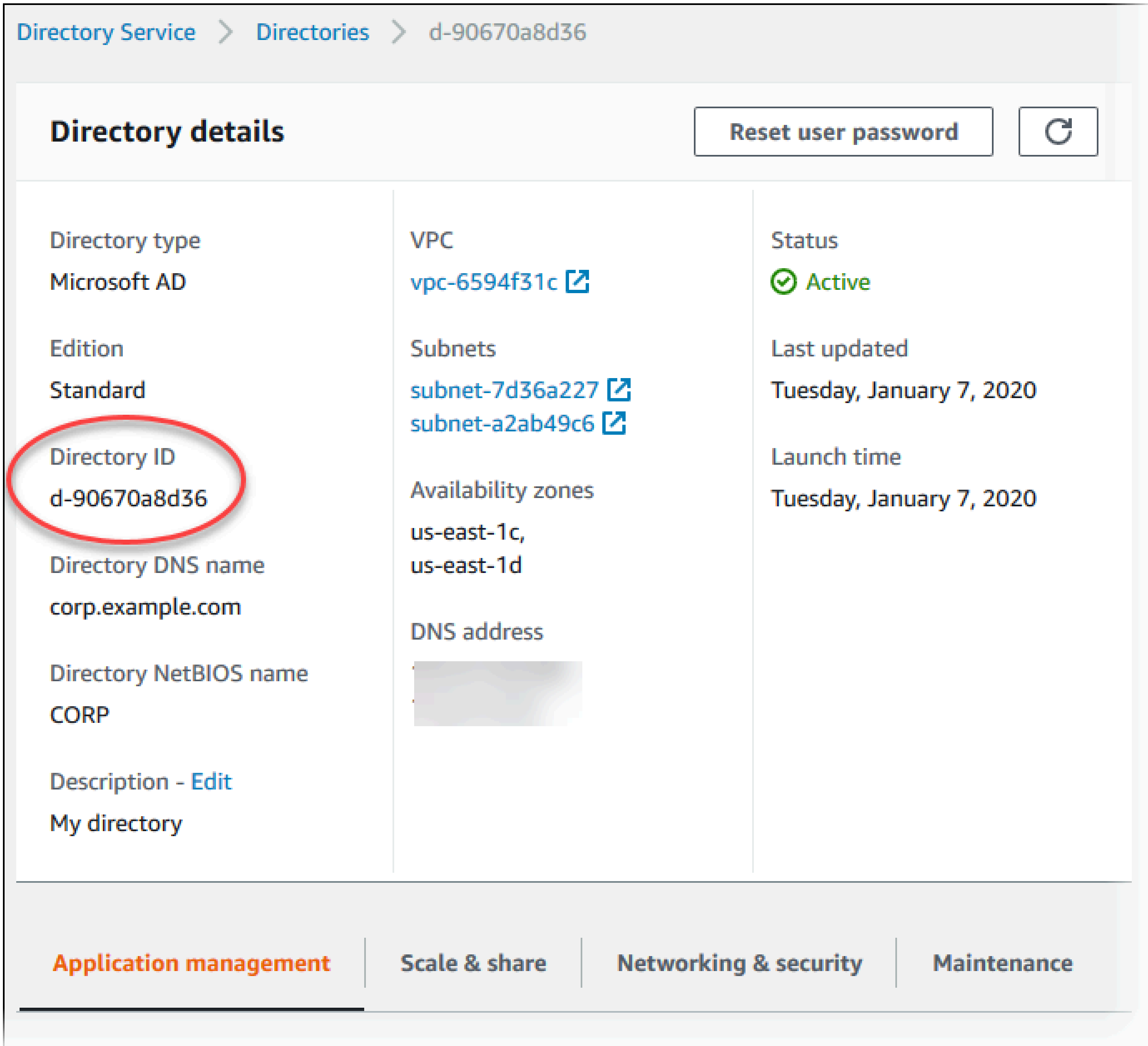
   子网

      目录服务器的子网。两个子网必须位于不同的可用区。
7. 查看目录信息并进行必要的更改。如果信息正确，请选择 Create directory (创建目录)。



目录创建需要几分钟时间。创建成功后，Status (状态) 值将更改为 Active (活动)。

要查看有关您的目录的信息，请在目录列表中选择目录名称。请记下 Directory ID (目录 ID) 值，因为您在创建或修改 Oracle 数据库实例时需要此值。



步骤 2：创建信任

如果您计划仅使用 Amazon Managed Microsoft AD，请转到步骤 3：配置 Amazon RDS 的 IAM 权限。

要使用自行管理的 Active Directory 启用 Kerberos 身份验证，您必须在自行管理的 Active Directory 与在上一步中创建的 Amazon Managed Microsoft AD 之间创建林信任关系。信任可以是单向的，其中 Amazon Managed Microsoft AD 信任自行管理的 Active Directory。信任也可以是双向的，此时两个 Active Directory 相互信任。有关使用 Amazon Directory Service 设置林信任的更多信息，请参阅 Amazon Directory Service 管理指南 中的何时创建信任关系。

步骤 3：配置 Amazon RDS 的 IAM 权限

要为您调用 Amazon Directory Service，Amazon RDS 需要一个使用托管式 IAM policy `AmazonRDSDirectoryServiceAccess` 的 IAM 角色。该角色允许 Amazon RDS 调用 Amazon Directory Service。

> **Note**
>
> 为了让角色允许访问，Amazon Security Token Service（Amazon STS）端点必须在您的 Amazon Web Services 账户的正确 Amazon Web Services 区域中激活。Amazon STS 端点原定设置为在所有 Amazon Web Services 区域中保持活跃，您无需执行任何进一步操作即可使用它们。有关更多信息，请参阅《IAM 用户指南》中的在 Amazon Web Services 区域中激活和停用 Amazon STS。

创建 IAM 角色

当您使用 Amazon Web Services Management Console创建数据库实例并且控制台用户具有 `iam:CreateRole` 权限时，控制台将自动创建 `rds-directoryservice-kerberos-access-role`。否则，您必须手动创建 IAM 角色。当手动创建 IAM 角色时，请选择 `Directory Service`，然后将 Amazon 托管式策略 `AmazonRDSDirectoryServiceAccess` 附加到该角色。

有关为服务创建 IAM 角色的更多信息，请参阅 IAM 用户指南中的创建向Amazon服务委托权限的角色。

> **Note**
>
> 用于 RDS for Microsoft SQL Server 的 Windows 身份验证 IAM 角色不能用于 RDS for Oracle。

手动创建 IAM 信任策略

或者，您可以创建具有所需权限的资源策略，而不是使用托管式 IAM policy `AmazonRDSDirectoryServiceAccess`。同时指定 `directoryservice.rds.amazonaws.com` 和 `rds.amazonaws.com` 作为主体。

要限制 Amazon RDS 为其他服务提供对资源的权限，我们建议在资源策略中使用 `aws:SourceArn` 和 `aws:SourceAccount` 全局条件上下文键。防范混淆代理问题最有效的方法是使用 `aws:SourceArn`

全局条件上下文键和 Amazon RDS 资源的完整 ARN。有关更多信息，请参阅 防范跨服务混淆代理问题。

以下示例演示如何使用 Amazon RDS 中的 `aws:SourceArn` 和 `aws:SourceAccount` 全局条件上下文键来防范混淆代理问题。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "directoryservice.rds.amazonaws.com",
          "rds.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "ArnLike": {
          "aws:SourceArn": "arn:aws:rds:us-east-1:123456789012:db:mydbinstance"
        },
        "StringEquals": {
          "aws:SourceAccount": "123456789012"
        }
      }
    }
  ]
}
```

对于选择加入区域，您还必须以 `directoryservice.rds.`*`region_name`*`.amazonaws.com` 形式包含该区域的服务主体。例如，在非洲（开普敦）区域中，使用以下信任策略：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "directoryservice.rds.amazonaws.com",
```

```
            "directoryservice.rds.af-south-1.amazonaws.com",
            "rds.amazonaws.com"
          ]
        },
        "Action": "sts:AssumeRole",
        "Condition": {
          "ArnLike": {
            "aws:SourceArn": "arn:aws:rds:af-south-1:123456789012:db:mydbinstance"
          },
          "StringEquals": {
            "aws:SourceAccount": "123456789012"
          }
        }
      }
    ]
}
```

角色还必须具有以下 IAM policy。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "ds:DescribeDirectories",
        "ds:AuthorizeApplication",
        "ds:UnauthorizeApplication",
        "ds:GetAuthorizedApplicationDetails"
      ],
    "Effect": "Allow",
    "Resource": "*"
    }
  ]
}
```

步骤 4：创建和配置用户

您可使用 Active Directory 用户和计算机工具（Active Directory 域服务和 Active Directory 轻型目录服务工具之一）创建用户。在这种情况下，用户表示有权访问您的目录的独立个人或实体。

要在 Amazon Directory Service 目录中创建用户和组，您必须连接到属于 Amazon Directory Service 目录成员的 Windows Amazon EC2 实例。同时，您必须以具有创建用户权限的用户身份登录。有关

在 Microsoft Active Directory 中创建用户的更多信息，请参阅 Amazon Managed Microsoft AD 管理指南中的管理 Amazon Directory Service 中的用户和组。

步骤 5：在目录和数据库实例之间启用跨 VPC 流量

如果您打算将目录和数据库实例放在同一 VPC 中，请跳过该步骤，然后转到步骤 6：创建或修改 Oracle 数据库实例。

如果您计划在不同Amazon账户或 VPC 中查找目录和数据库实例，请使用 VPC 对等连接或Amazon Transit Gateway 配置跨 VPC 流量。以下过程使用 VPC 对等连接启用 VPC 之间的流量。请按照 Amazon Virtual Private Cloud 对等连接指南中的什么是 VPC 对等连接？操作。

使用 VPC 对等连接启用跨 VPC 流量

1. 设置适合的 VPC 路由规则，以便确保网络流量可以双向流动。
2. 确保数据库实例的安全组可以从目录的安全组接收入站流量。有关更多信息，请参阅 Amazon Managed Microsoft AD 管理指南 中的 Amazon Directory Service 最佳实践。
3. 确保没有任何网络访问控制列表 (ACL) 规则会阻止流量。

如果该目录由不同的 Amazon 账户拥有，则您必须共享该目录。

在 Amazon 账户之间共享目录

1. 按照 Amazon 管理指南中的教程：共享 Amazon Managed Microsoft AD 目录以实现无缝 EC2 域加入中的说明，开始与将要在其中创建数据库实例的 Amazon Directory Service 账户共享目录。
2. 使用数据库实例的账户登录到 Amazon Directory Service 控制台，并确保在处理之前域具有 `SHARED` 状态。
3. 使用数据库实例的账户登录 Amazon Directory Service 控制台时，请记录目录 ID 值。您可以使用此目录 ID 将数据库实例加入域。

步骤 6：创建或修改 Oracle 数据库实例

创建或修改 Oracle 数据库实例，以便与您的目录一起使用。您可以使用控制台、CLI 或 RDS API 将数据库实例与目录关联。您可以通过下列方式之一来执行该操作：

- 使用控制台、 create-db-instance CLI 命令或 CreateDBInstance RDS API 操作创建新的 Oracle 数据库实例。

  有关说明，请参阅创建 Amazon RDS 数据库实例。

- 使用控制台、modify-db-instance CLI 命令或 ModifyDBInstance RDS API 操作修改现有 Oracle 数据库实例。

  有关说明，请参阅修改 Amazon RDS 数据库实例。
- 使用控制台、restore-db-instance-from-db-snapshot CLI 命令或 RestoreDBInstanceFromDBSnapshot RDS API 操作从数据库快照还原 Oracle 数据库实例。

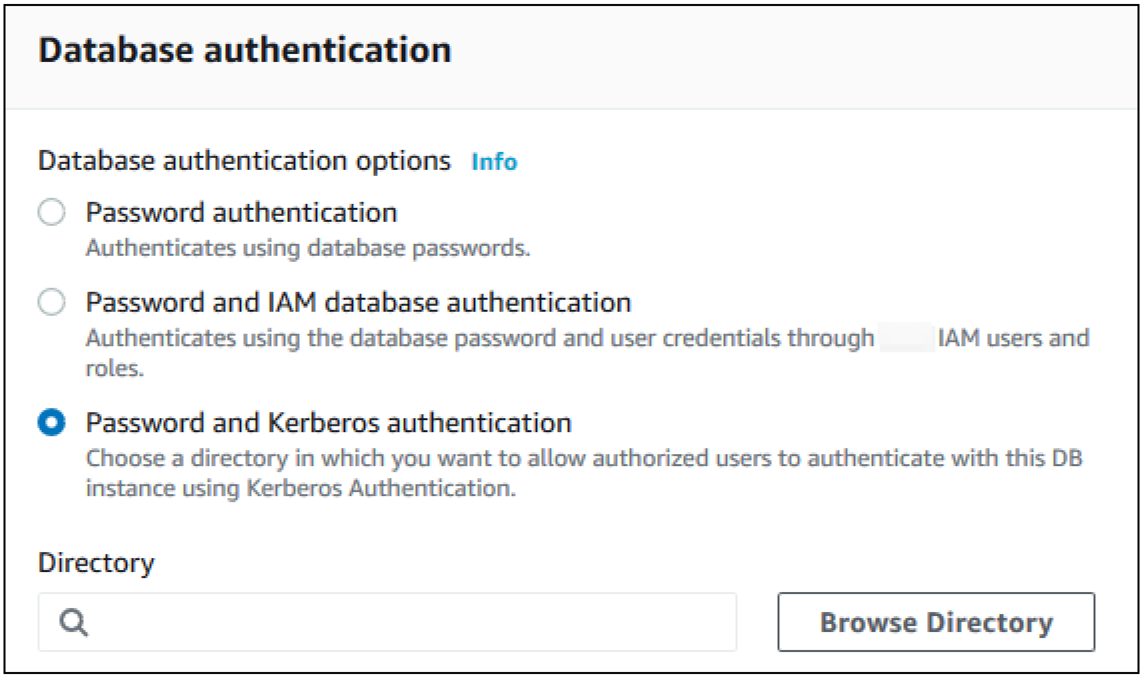
  有关说明，请参阅还原到数据库实例。
- 使用控制台、restore-db-instance-to-point-in-time CLI 命令或 RestoreDBInstanceToPointInTime RDS API 操作将 Oracle 数据库实例还原到某个时间点。

  有关说明，请参阅将 Amazon RDS 的数据库实例还原到指定时间。

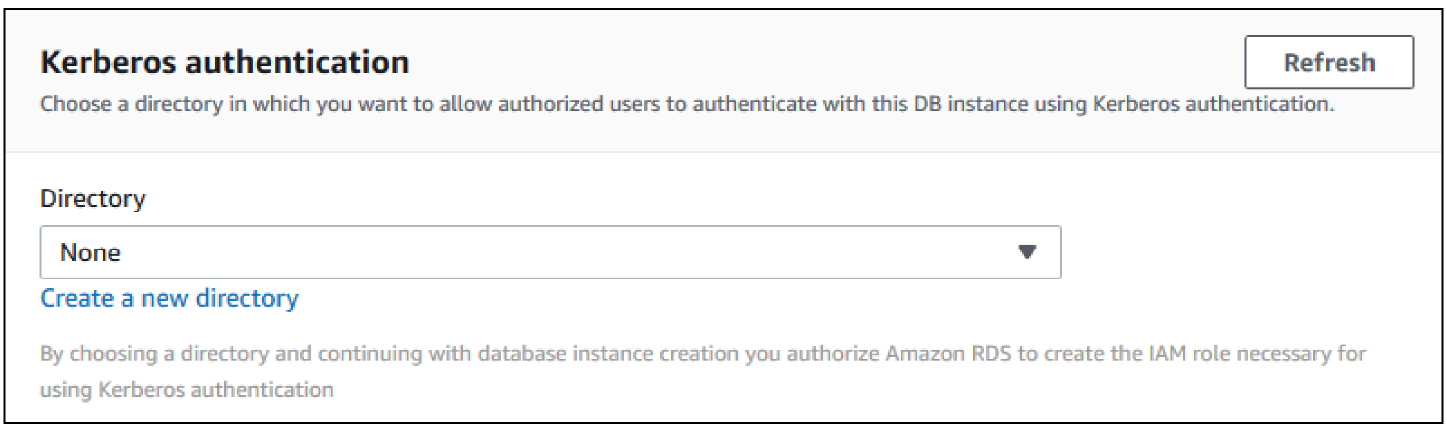
只有 VPC 中的 Oracle 数据库实例才支持 Kerberos 身份验证。数据库实例可以与目录在同一 VPC 中或在不同 VPC 中。创建或修改数据库实例时，请执行以下操作：

- 请提供创建目录时生成的域标识符（d-* 标识符）。
- 还要提供您创建的 IAM 角色的名称。
- 确保数据库实例安全组可以从目录安全组接收入站流量并将出站流量发送到目录。

在使用控制台创建数据库实例时，请选择 Database authentication (数据库身份验证) 部分中的 Password and Kerberos authentication (密码和 Kerberos 身份验证)。选择 Browse Directory (浏览目录)，然后选择目录或选择 Create a new directory (创建新目录)。



在使用控制台修改或还原数据库实例时，请在 Kerberos authentication (Kerberos 身份验证) 部分中选择目录，或选择 Create a new directory (创建新目录)。



使用 Amazon CLI 时，数据库实例需要以下参数才能使用您创建的目录：

- 对于 `--domain` 参数，请使用创建目录时生成的域标识符（“d-*”标识符）。
- 对于 `--domain-iam-role-name` 参数，请使用您使用托管 IAM 策略 `AmazonRDSDirectoryServiceAccess` 创建的角色。

例如，以下 CLI 命令会修改数据库实例以使用目录。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier mydbinstance \
    --domain d-ID \
    --domain-iam-role-name role-name
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier mydbinstance ^
    --domain d-ID ^
    --domain-iam-role-name role-name
```

Important

如果您修改数据库实例以启用 Kerberos 身份验证，请在进行更改之后重启数据库实例。

Note

MANAGED_SERVICE_USER 是一个服务账户，名称由 RDS 的 Directory Service 随机生成。在 Kerberos 身份验证设置期间，RDS for Oracle 会使用同一名称创建用户并为其分配 `CREATE SESSION` 权限。在外部将 Oracle 数据库用户标识为 *MANAGED_SERVICE_USER@EXAMPLE.COM*，其中 *EXAMPLE.COM* 是您的域名。RDS 会定期使用 Directory Service 提供的凭证登录 Oracle 数据库。之后，RDS 会立即销毁票证缓存。

步骤 7：创建 Kerberos 身份验证 Oracle 登录名

使用 Amazon RDS 主用户凭证连接到 Oracle 数据库实例，就像连接到任何其他数据库实例一样。数据库实例加入到 Amazon Managed Microsoft AD 域。这样，您可以根据从域中的 Microsoft Active Directory 用户预置 Oracle 登录名和用户。要管理数据库权限，可以授予和撤消这些登录的标准 Oracle 权限。

要允许 Microsoft Active Directory 用户向 Oracle 验证身份

1. 请使用 Amazon RDS 主用户凭证连接到 Oracle 数据库实例。
2. 在 Oracle 数据库中创建在外部验证身份的用户。

   在以下示例中，使用您的用户名和域名替换 *KRBUSER@CORP.EXAMPLE.COM*。

```
CREATE USER "KRBUSER@CORP.EXAMPLE.COM" IDENTIFIED EXTERNALLY;
GRANT CREATE SESSION TO "KRBUSER@CORP.EXAMPLE.COM";
```

域中用户（人和应用程序）现在均可使用 Kerberos 身份验证从加入域的客户端计算机连接到 Oracle 数据库实例。

步骤 8：配置 Oracle 客户端

要配置 Oracle 客户端，请满足以下要求：

- 创建一个名为 krb5.conf (Linux) 或 krb5.ini (Windows) 的配置文件，以指向域。将 Oracle 客户端配置为使用此配置文件。
- 验证流量是否可以在 DNS 端口 53 上通过 TCP/UDP、在 Kerberos 端口（用于托管 Amazon Directory Service 的 88 和 464）上通过 TCP 以及在 LDAP 端口 389 上通过 TCP 在客户端主机和 Amazon Directory Service 之间进行流动。
- 验证流量是否可以通过数据库端口在客户端主机和数据库实例之间流动。

以下是 Amazon Managed Microsoft AD 的示例内容。

```
[libdefaults]
 default_realm = EXAMPLE.COM
[realms]
 EXAMPLE.COM = {
  kdc = example.com
  admin_server = example.com
 }
[domain_realm]
 .example.com = CORP.EXAMPLE.COM
 example.com = CORP.EXAMPLE.COM
```

以下是本地 Microsoft AD 的示例内容。在 krb5.conf 或 krb5.ini 文件中，将 *on-prem-ad-server-name* 替换为本地 AD 服务器的名称。

```
[libdefaults]
 default_realm = ONPREM.COM
[realms]
 AWSAD.COM = {
  kdc = awsad.com
```

```
  admin_server = awsad.com
 }
 ONPREM.COM = {
  kdc = on-prem-ad-server-name
  admin_server = on-prem-ad-server-name
 }
[domain_realm]
 .awsad.com = AWSAD.COM
 awsad.com= AWSAD.COM
 .onprem.com = ONPREM.COM
 onprem.com= ONPREM.COM
```

> **Note**
>
> 配置 krb5.ini 或 krb5.conf 文件后，我们建议您重启服务器。

以下是 SQL*Plus 配置的示例 sqlnet.ora 内容：

```
SQLNET.AUTHENTICATION_SERVICES=(KERBEROS5PRE,KERBEROS5)
SQLNET.KERBEROS5_CONF=path_to_krb5.conf_file
```

有关 SQL Developer 配置示例，请参阅 Oracle 支持网站中的文档 1609359.1。

在域中管理数据库实例

您可以使用控制台、CLI 或 RDS API 来管理您的数据库实例及其与 Microsoft Active Directory 的关系。例如，您可以关联 Microsoft Active Directory 以启用 Kerberos 身份验证。您也可以取消 Microsoft Active Directory 的关联以禁用 Kerberos 身份验证。您也可以将由一个 Microsoft Active Directory 在外部进行身份验证的数据库实例移动到另一个 Active Directory。

例如，使用 CLI，您可以执行下列操作：

- 要重新尝试为失败的成员启用 Kerberos 身份验证，请使用 modify-db-instance CLI 命令并为 `--domain` 选项指定当前成员的目录 ID。
- 要禁用数据库实例上的 Kerberos 身份验证，请使用 modify-db-instance CLI 命令并为 none 选项指定 `--domain`。
- 要将数据库实例从一个域移至另一个域，请使用 modify-db-instance CLI 命令并为 `--domain` 选项指定新域的域标识符。

查看域成员资格的状态

在创建或修改数据库实例后，数据库实例将成为域的成员。您可以在控制台中查看数据库实例的域成员身份状态，也可以通过运行 describe-db-instances CLI 命令来查看。数据库实例的状态可以是以下状态之一：

- `kerberos-enabled` – 数据库实例已启用 Kerberos 身份验证。
- `enabling-kerberos` – Amazon是在此数据库实例上启用 Kerberos 身份验证的过程。
- `pending-enable-kerberos` – 启用 Kerberos 身份验证正在此数据库实例上等待处理。
- `pending-maintenance-enable-kerberos` – Amazon将尝试在下一个计划的维护时段在数据库实例上启用 Kerberos 身份验证。
- `pending-disable-kerberos` – 禁用 Kerberos 身份验证正在此数据库实例上等待处理。
- `pending-maintenance-disable-kerberos` – Amazon将尝试在下一个计划的维护时段在数据库实例上禁用 Kerberos 身份验证。
- `enable-kerberos-failed` – 出现一个配置问题，导致Amazon无法在数据库实例上启用 Kerberos 身份验证。在重新发出命令以修改数据库实例之前纠正配置问题。
- `disabling-kerberos` – Amazon是在此数据库实例上启用 Kerberos 身份验证的过程。

启用 Kerberos 身份验证的请求可能因网络连接问题或不正确的 IAM 角色而失败。如果在创建或修改数据库实例时尝试启用 Kerberos 身份验证失败，请确保您使用的是正确的 IAM 角色。然后修改数据库实例以加入域。

> **Note**
>
> 只有对 Amazon RDS for Oracle 进行 Kerberos 身份验证才会将流量发送到域的 DNS 服务器。所有其他 DNS 请求被视为对运行 Oracle 的数据库实例进行出站网络访问。有关 Amazon RDS for Oracle 出站网络访问的更多信息，请参阅设置自定义 DNS 服务器。

强制轮换 Kerberos 密钥

私有密钥是在 Amazon Managed Microsoft AD 和 Amazon RDS for Oracle 数据库实例之间共享的。该密钥每 45 天自动轮换一次。您可以使用以下 Amazon RDS 过程强制轮换该密钥：

```
SELECT rdsadmin.rdsadmin_kerberos_auth_tasks.rotate_kerberos_keytab AS TASK_ID FROM
 DUAL;
```

> **Note**
> 在只读副本配置中，只能在源数据库实例上使用该过程，而不能在只读副本上使用该过程。

`SELECT` 语句返回 `VARCHAR2` 数据类型的任务 ID。您可以查看 bdump 文件中的持续任务的状态。bdump 文件位于 `/rdsdbdata/log/trace` 目录下。每个 bdump 文件名称格式如下。

```
dbtask-task-id.log
```

您可以通过显示任务的输出文件来查看结果。

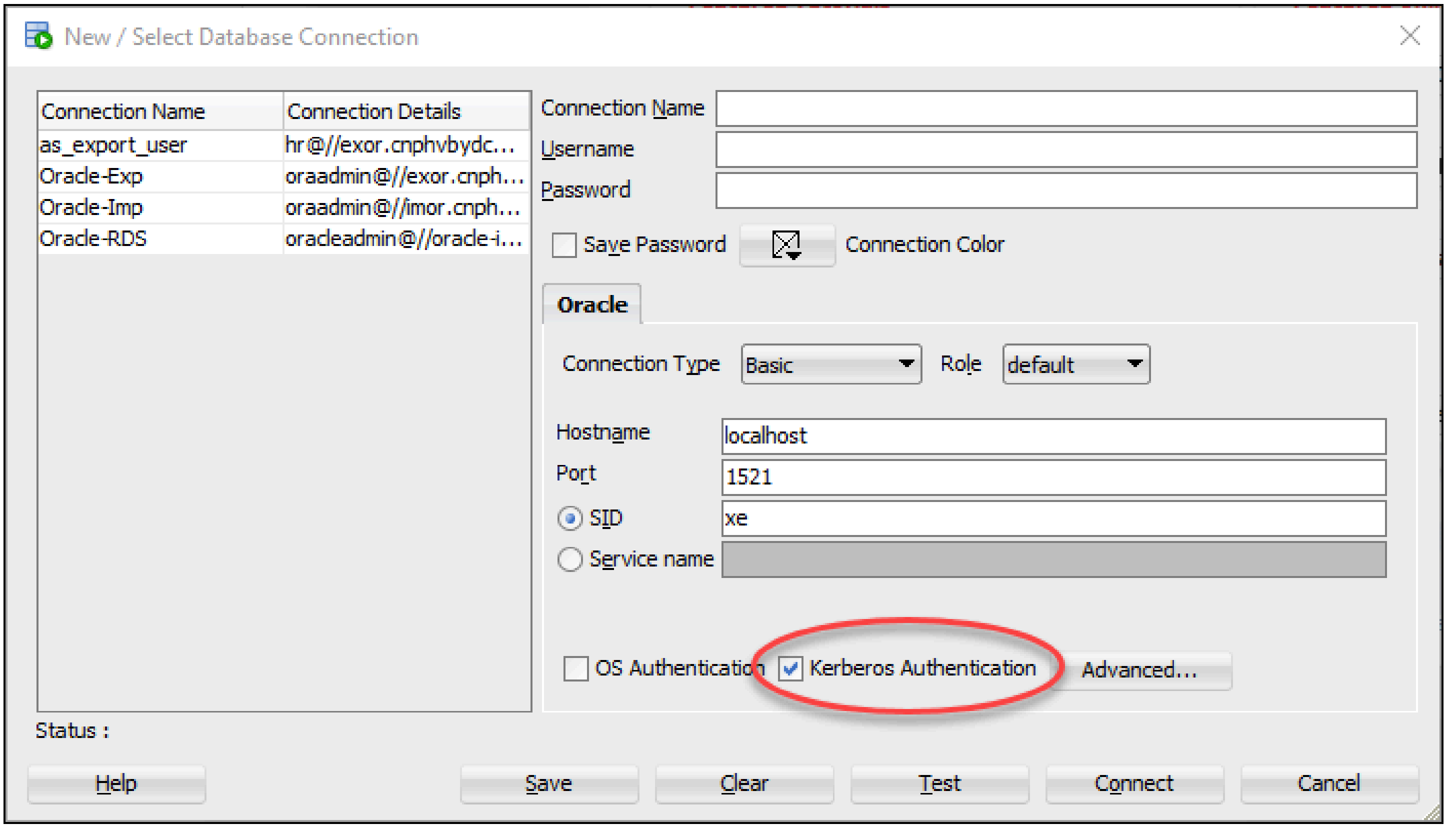
```
SELECT text FROM table(rdsadmin.rds_file_util.read_text_file('BDUMP','dbtask-task-id.log'));
```

将 *task-id* 替换为由此过程返回的任务 ID。

> **Note**
> 任务以异步方式执行。

使用 Kerberos 身份验证连接到 Oracle

这部分假定您已按步骤 8：配置 Oracle 客户端中所述设置 Oracle 客户端。要使用 Kerberos 身份验证连接到 Oracle 数据库，请使用 Kerberos 身份验证类型登录。例如，在启动 Oracle SQL Developer 之后，请选择 Kerberos 身份验证作为身份验证类型，如下面的示例所示。



要通过 SQL*Plus 使用 Kerberos 身份验证连接到 Oracle，请执行以下操作：

1. 在命令提示符处，运行以下命令：

```
kinit username
```

   将 *username* 替换为用户名，并在提示符下输入 Microsoft Active Directory 为用户存储的密码。

2. 打开 SQL*Plus，使用 Oracle 数据库实例的 DNS 名称和端口号连接。

   有关在 SQL*Plus 中连接到 Oracle 数据库实例的更多信息，请参阅使用 SQL*Plus 连接到数据库实例。

> **Tip**
>
> 如果您使用的是原生 Windows 缓存，也可以在 sqlnet.ora 文件中将 `SQLNET.KERBEROS5_CC_NAME` 参数设置为 `OSMSFT://` 或 `MSLSA`，以便使用存储在 Microsoft Active Directory 中的凭证。

使用证书和 Oracle Wallet 配置 UTL_HTTP 访问

Amazon RDS 支持对 RDS for Oracle 数据库实例进行出站网络访问。要将数据库实例连接到网络，您可以使用以下 PL/SQL 软件包：

`UTL_HTTP`

此软件包从 SQL 和 PL/SQL 发出 HTTP 调用。您可以使用它通过 HTTP 访问 Internet 上的数据。有关详细信息，请参阅 Oracle 文档中的 UTL_HTTP。

`UTL_TCP`

此软件包在 PL/SQL 中提供 TCP/IP 客户端访问功能。此软件包对于使用 Internet 协议和电子邮件的 PL/SQL 应用程序很有用。有关更多信息，请参阅 Oracle 文档中的 UTL_TCP。

`UTL_SMTP`

此软件包提供 SMTP 命令的接口，以便客户端能够向 SMTP 服务器发送电子邮件。有关详细信息，请参阅 Oracle 文档中的 UTL_SMTP。

您可以通过完成以下任务来配置 `UTL_HTTP.REQUEST`，以便与 SSL 握手期间需要客户端身份验证证书的网站搭配使用。您还可以通过修改 Oracle Wallet 生成命令和 `DBMS_NETWORK_ACL_ADMIN.APPEND_WALLET_ACE` 过程，为 UTL_HTTP 访问网站配置密码身份验证。有关更多信息，请参阅 Oracle Database 文档中的 DBMS_NETWORK_ACL_ADMIN。

> **Note**
>
> 您可为 `UTL_SMTP` 调整以下任务，以便通过 SSL/TLS（包括 Amazon Simple Email Service）发送电子邮件。

主题

配置 UTL_HTTP 访问时的注意事项

在配置访问之前，请考虑以下事项：

- 您可以将 SMTP 与 UTL_MAIL 选项结合使用。有关更多信息，请参阅 Oracle UTL_MAIL。
- 远程主机的域名服务器 (DNS) 名称满足以下所有条件：
  - 可公开解析。
  - Amazon RDS 数据库实例的终端节点。
  - 可通过自定义 DNS 服务器解析。有关更多信息，请参阅“设置自定义 DNS 服务器”。
  - 同一 VPC 或对等 VPC 中的 Amazon EC2 实例的私有 DNS 名称。在这种情况下，请确认该名称可通过自定义 DNS 服务器解析。或者，要使用 Amazon 提供的 DNS，您可以启用 VPC 设置中的 `enableDnsSupport` 属性，并启用对 VPC 对等连接的 DNS 解析支持。有关更多信息，请参阅 VPC 中的 DNS 支持和修改 VPC 对等连接。
  - 若要安全地连接到远程 SSL/TLS 资源，建议创建和上传自定义 Oracle Wallet。使用与 Amazon RDS for Oracle 功能集成的 Amazon S3，您可以将 Wallet 从 Amazon S3 下载到 Oracle 数据库实例。有关 Oracle 的 Amazon S3 集成的信息，请参阅Amazon S3 集成。
- 如果为每个实例配置了 Oracle SSL 选项，您可以在 SSL/TLS 终端节点上的 Oracle 数据库实例之间建立数据库连接。无需作进一步配置。有关更多信息，请参阅 Oracle 安全套接字层。

步骤 1：获取网站的根证书

要在 RDS for Oracle 数据库实例与网站之间建立安全连接，请添加根 CA 证书。Amazon RDS 使用根证书将网站证书签署到 Oracle Wallet。

您可以通过各种方式获取根证书。例如，您可以执行以下操作：

1. 使用 Web 服务器访问受证书保护的网站。
2. 下载用于签名的根证书。

对于 Amazon 服务，根证书通常位于 Amazon Trust Services 存储库中。

步骤 2：创建 Oracle Wallet

创建包含 Web 服务器证书和客户端身份验证证书的 Oracle Wallet。RDS Oracle 实例使用 Web 服务器证书建立与网站的安全连接。网站需要客户端证书才能对 Oracle 数据库用户进行身份验证。

您可能希望在不使用客户端证书进行身份验证的情况下配置安全连接。在这种情况下，您可以跳过以下过程中的 Java 密钥库步骤。

创建 Oracle Wallet

1. 将根证书和客户端证书放在一个目录中，然后更改到此目录。
2. 将 .p12 客户端证书转换为 Java 密钥库。

> **Note**
> 如果没有使用客户端证书进行身份验证，可以跳过此步骤。

以下示例将名为 *client_certificate.p12* 的客户端证书转换为名为 *client_keystore.jks* 的 Java 密钥库。该密钥库随后会包含在 Oracle Wallet 中。密钥库密码为 *P12PASSWORD*。

```
orapki wallet pkcs12_to_jks -wallet ./client_certificate.p12 -
jksKeyStoreLoc ./client_keystore.jks -jksKeyStorepwd P12PASSWORD
```

3. 为 Oracle Wallet 创建与证书目录不同的目录。

以下示例会创建 `/tmp/wallet` 目录。

```
mkdir -p /tmp/wallet
```

4. 在 Wallet 目录中创建 Oracle Wallet。

以下示例将 Oracle Wallet 密码设置为 *P12PASSWORD*，这与之前步骤中 Java 密钥库使用的密码相同。虽然使用相同的密码很方便，但没有必要这样做。`-auto_login` 参数会打开自动登录功能，无需在每次访问时指定密码。

> **Note**
> 作为安全最佳实践，请指定除此处所示提示以外的密码。

```
orapki wallet create -wallet /tmp/wallet -pwd P12PASSWORD -auto_login
```

5. 将 Java 密钥库添加到 Oracle Wallet 中。

   > **Note**
   >
   > 如果没有使用客户端证书进行身份验证，可以跳过此步骤。

   以下示例将密钥库 *client_keystore.jks* 添加到名为 */mp/wallet* 的 Oracle Wallet 中。在此示例中，您为 Java 密钥库和 Oracle Wallet 指定了相同密码。

   ```
   orapki wallet jks_to_pkcs12 -wallet /tmp/wallet -pwd P12PASSWORD -
   keystore ./client_keystore.jks -jkspwd P12PASSWORD
   ```

6. 将目标网站的根证书添加到 Oracle Wallet。

   以下示例将添加名为 *Root_CA.cer* 的证书。

   ```
   orapki wallet add -wallet /tmp/wallet -trusted_cert -cert ./Root_CA.cer -
   pwd P12PASSWORD
   ```

7. 添加任何中间证书。

   以下示例将添加名为 *Intermediate.cer* 的证书。根据需要多次重复此步骤，以便加载所有中间证书。

   ```
   orapki wallet add -wallet /tmp/wallet -trusted_cert -cert ./Intermediate.cer -
   pwd P12PASSWORD
   ```

8. 确认新创建的 Oracle Wallet 具有所需证书。

   ```
   orapki wallet display -wallet /tmp/wallet -pwd P12PASSWORD
   ```

步骤 3：将 Oracle Wallet 下载到 RDS for Oracle 实例

在此步骤中，将 Oracle Wallet 上传到 Amazon S3，然后将 Wallet 从 Amazon S3 下载到 RDS for Oracle 实例。

将 Oracle Wallet 下载到 RDS for Oracle 数据库实例

1. 完成 Amazon S3 与 Oracle 集成的先决条件，然后将 S3_INTEGRATION 选项添加到您的 Oracle 数据库实例。确保该选项的 IAM 角色具有您正在使用的 Amazon S3 存储桶的访问权限。

   有关更多信息，请参阅 Amazon S3 集成。
2. 以主用户身份登录数据库实例，然后创建 Oracle 目录来保存 Oracle Wallet。

   以下示例将创建名为 *WALLET_DIR* 的 Oracle 目录。

   ```
   EXEC rdsadmin.rdsadmin_util.create_directory('WALLET_DIR');
   ```

   有关更多信息，请参阅 在主数据存储空间中创建和删除目录。
3. 将 Oracle Wallet 上传到 Amazon S3 存储桶。

   您可以使用任何受支持的上传技术。
4. 如果要重新上传 Oracle Wallet，请删除现有 Wallet。否则，请跳到下一步。

   以下示例将删除名为 *cwallet.sso* 的现有 Wallet。

   ```
   EXEC UTL_FILE.FREMOVE ('WALLET_DIR','cwallet.sso');
   ```
5. 将 Oracle Wallet 从 Amazon S3 存储桶下载到 Oracle 数据库实例。

   以下示例会将名为 *cwallet.sso* 的 Wallet 从名为 *my_s3_bucket* 的 Amazon S3 存储桶下载到名为 *WALLET_DIR* 的数据库实例目录。

   ```
   SELECT rdsadmin.rdsadmin_s3_tasks.download_from_s3(
         p_bucket_name    =>  'my_s3_bucket',
         p_s3_prefix      =>  'cwallet.sso',
         p_directory_name =>  'WALLET_DIR')
      AS TASK_ID FROM DUAL;
   ```
6. （可选）下载受密码保护的 Oracle Wallet。

   仅在每次使用 Wallet 都需要密码时，才下载此 Wallet。以下示例将下载受密码保护的 Wallet *ewallet.p12*。

   ```
   SELECT rdsadmin.rdsadmin_s3_tasks.download_from_s3(
         p_bucket_name    =>  'my_s3_bucket',
   ```

```
        p_s3_prefix       =>  'ewallet.p12',
        p_directory_name =>  'WALLET_DIR')
   AS TASK_ID FROM DUAL;
```

7. 检查数据库任务的状态。

   在以下示例中，将上述步骤返回的任务 ID 替换为 *dbtask-1234567890123-4567.log*。

   ```
   SELECT TEXT FROM
    TABLE(rdsadmin.rds_file_util.read_text_file('BDUMP','dbtask-1234567890123-4567.log'));
   ```

8. 检查用于存储 Oracle Wallet 的目录的内容。

   ```
   SELECT * FROM TABLE(rdsadmin.rds_file_util.listdir(p_directory => 'WALLET_DIR'));
   ```

   有关更多信息，请参阅 列出数据库实例目录中的文件。

步骤 4：授予用户 Oracle Wallet 权限

您可以创建新的数据库用户或配置现有用户。无论采用哪种方法，都必须配置用户使用证书访问 Oracle Wallet，以便实现安全连接和客户端身份验证。

授予用户 Oracle Wallet 权限

1. 以主用户身份登录 RDS for Oracle 数据库实例。
2. 如果不想配置现有数据库用户，可以选择创建新用户。否则，请跳到下一步。

   以下示例将创建名为 *my-user* 的数据库用户。

   ```
   CREATE USER my-user IDENTIFIED BY my-user-pwd;
   GRANT CONNECT TO my-user;
   ```

3. 授予数据库用户对包含 Oracle Wallet 的目录的权限。

   以下示例将授予用户 *my-user* 对目录 *WALLET_DIR* 的读取访问权限。

   ```
   GRANT READ ON DIRECTORY WALLET_DIR TO my-user;
   ```

4. 授予数据库用户对 UTL_HTTP 软件包的使用权限。

   以下 PL/SQL 计划授予用户 *my-user* 对 UTL_HTTP 的访问权限。

```
BEGIN
  rdsadmin.rdsadmin_util.grant_sys_object('UTL_HTTP', UPPER('my-user'));
  END;
/
```

5. 授予数据库用户对 UTL_FILE 软件包的使用权限。

   以下 PL/SQL 计划授予用户 *my-user* 对 UTL_FILE 的访问权限。

```
BEGIN
  rdsadmin.rdsadmin_util.grant_sys_object('UTL_FILE', UPPER('my-user'));
  END;
/
```

步骤 5：配置从数据库实例访问网站

在此步骤中，您将配置 Oracle 数据库用户，以便其可以使用 UTL_HTTP、上传的 Oracle Wallet 和客户端证书连接到目标网站。有关更多信息，请参阅 Oracle Database 文档中的配置对 Oracle Wallet 的访问控制。

配置从 RDS for Oracle 数据库实例访问网站

1. 以主用户身份登录 RDS for Oracle 数据库实例。
2. 在安全端口上为用户和目标网站创建主机访问控制条目 (ACE)。

   以下示例将 *my-user* 配置为在安全端口 443 上访问 *secret.encrypted-website.com*。

```
BEGIN
  DBMS_NETWORK_ACL_ADMIN.APPEND_HOST_ACE(
    host       => 'secret.encrypted-website.com',
    lower_port => 443,
    upper_port => 443,
    ace        => xs$ace_type(privilege_list => xs$name_list('http'),
                              principal_name => 'my-user',
                              principal_type => xs_acl.ptype_db));
                           -- If the program unit results in PLS-00201, set
                           -- the principal_type parameter to 2 as follows:
                           -- principal_type => 2));
END;
```

```
/
```

> **Important**
>
> 前面的程序单元可能会导致以下错误：`PLS-00201: identifier 'XS_ACL' must be declared`。如果返回此错误，请将为 `principal_type` 分配值的行替换为以下行，然后重新运行程序单元：
>
> ```
> principal_type => 2));
> ```
>
> 有关 PL/SQL 软件包 XS_ACL 中常量的更多信息，请参阅 Oracle Database 文档中的 [Real Application Security Administrator's and Developer's Guide]。

有关更多信息，请参阅 Oracle Database 文档中的[为外部网络服务配置访问控制]。

3. （可选）在标准端口上为用户和目标网站创建主机访问控制条目。

   如果是标准 Web 服务器端口 (80) 而不是安全端口 (443) 提供的网页，则可能需要使用标准端口。

```
BEGIN
  DBMS_NETWORK_ACL_ADMIN.APPEND_HOST_ACE(
    host       => 'secret.encrypted-website.com',
    lower_port => 80,
    upper_port => 80,
    ace        => xs$ace_type(privilege_list => xs$name_list('http'),
                              principal_name => 'my-user',
                              principal_type => xs_acl.ptype_db));
                           -- If the program unit results in PLS-00201, set
                           -- the principal_type parameter to 2 as follows:
                           -- principal_type => 2));
END;
/
```

4. 确认存在访问控制条目。

```
SET LINESIZE 150
COLUMN HOST FORMAT A40
COLUMN ACL FORMAT A50

SELECT HOST, LOWER_PORT, UPPER_PORT, ACL
```

```
  FROM DBA_NETWORK_ACLS
ORDER BY HOST;
```

5. 授予数据库用户对 UTL_HTTP 软件包的使用权限。

   以下 PL/SQL 计划授予用户 *my-user* 对 UTL_HTTP 的访问权限。

```
BEGIN
  rdsadmin.rdsadmin_util.grant_sys_object('UTL_HTTP', UPPER('my-user'));
  END;
/
```

6. 确认存在相关的访问控制列表。

```
SET LINESIZE 150
COLUMN ACL FORMAT A50
COLUMN PRINCIPAL FORMAT A20
COLUMN PRIVILEGE FORMAT A10

SELECT ACL, PRINCIPAL, PRIVILEGE, IS_GRANT,
       TO_CHAR(START_DATE, 'DD-MON-YYYY') AS START_DATE,
       TO_CHAR(END_DATE, 'DD-MON-YYYY') AS END_DATE
  FROM DBA_NETWORK_ACL_PRIVILEGES
ORDER BY ACL, PRINCIPAL, PRIVILEGE;
```

7. 授予数据库用户使用证书进行客户端身份验证和 Oracle Wallet 进行连接的权限。

   > **Note**
   >
   > 如果没有使用客户端证书进行身份验证，可以跳过此步骤。

```
DECLARE
  l_wallet_path all_directories.directory_path%type;
BEGIN
  SELECT DIRECTORY_PATH
    INTO l_wallet_path
    FROM ALL_DIRECTORIES
   WHERE UPPER(DIRECTORY_NAME)='WALLET_DIR';
  DBMS_NETWORK_ACL_ADMIN.APPEND_WALLET_ACE(
    wallet_path => 'file:/' || l_wallet_path,
```

```
    ace          =>  xs$ace_type(privilege_list => xs
$name_list('use_client_certificates'),
                                principal_name => 'my-user',
                                principal_type => xs_acl.ptype_db));
END;
/
```

步骤 6：测试从数据库实例到网站的连接

在此步骤中，您将配置数据库用户，以便其可以使用 UTL_HTTP、上传的 Oracle Wallet 和客户端证书连接到网站。

配置从 RDS for Oracle 数据库实例访问网站

1. 以具有 UTL_HTTP 权限的数据库用户身份登录 RDS for Oracle 数据库实例。
2. 确认与目标网站的连接可以解析主机地址。

   以下示例将从 *secret.encrypted-website.com* 获取主机地址。

   ```
   SELECT UTL_INADDR.GET_HOST_ADDRESS(host => 'secret.encrypted-website.com')
     FROM DUAL;
   ```
3. 测试失败连接。

   以下查询失败，因为 UTL_HTTP 需要带有证书的 Oracle Wallet 的位置。

   ```
   SELECT UTL_HTTP.REQUEST('secret.encrypted-website.com') FROM DUAL;
   ```
4. 使用 `UTL_HTTP.SET_WALLET` 并从 `DUAL` 中进行选择来测试网站访问。

   ```

   DECLARE
     l_wallet_path all_directories.directory_path%type;
   BEGIN
     SELECT DIRECTORY_PATH
       INTO l_wallet_path
       FROM ALL_DIRECTORIES
      WHERE UPPER(DIRECTORY_NAME)='WALLET_DIR';
     UTL_HTTP.SET_WALLET('file:/' || l_wallet_path);
   END;
   /
   ```

```
SELECT UTL_HTTP.REQUEST('secret.encrypted-website.com') FROM DUAL;
```

5. （可选）将查询存储在变量中并使用 `EXECUTE IMMEDIATE` 来测试网络访问。

```
DECLARE
  l_wallet_path all_directories.directory_path%type;
  v_webpage_sql VARCHAR2(1000);
  v_results     VARCHAR2(32767);
BEGIN
  SELECT DIRECTORY_PATH
    INTO l_wallet_path
    FROM ALL_DIRECTORIES
   WHERE UPPER(DIRECTORY_NAME)='WALLET_DIR';
  v_webpage_sql := 'SELECT UTL_HTTP.REQUEST(''secret.encrypted-website.com'', '''',
 ''file:/' ||l_wallet_path||''') FROM DUAL';
  DBMS_OUTPUT.PUT_LINE(v_webpage_sql);
  EXECUTE IMMEDIATE v_webpage_sql INTO v_results;
  DBMS_OUTPUT.PUT_LINE(v_results);
END;
/
```

6. （可选）查找 Oracle Wallet 目录的文件系统位置。

```
SELECT * FROM TABLE(rdsadmin.rds_file_util.listdir(p_directory => 'WALLET_DIR'));
```

使用上一个命令的输出来发出 HTTP 请求。例如，如果目录为 *rdsdbdata/userdirs/01*，则运行以下查询。

```
SELECT UTL_HTTP.REQUEST('https://secret.encrypted-website.com/', '',
 'file://rdsdbdata/userdirs/01')
FROM   DUAL;
```

在 RDS for Oracle 中使用 CDB

在 Oracle 多租户架构中，容器数据库（CDB）可以包括客户创建的可插拔数据库（PDB）。有关 CDB 的更多信息，请参阅 Oracle 数据库文档中的多租户架构简介。

主题

RDS for Oracle CDB 概述

运行 Oracle Database 19c 或更高版本时，您可以将 RDS for Oracle 数据库实例创建为容器数据库（CDB）。从 Oracle Database 21c 开始，所有数据库都是 CDB。CDB 不同于非 CDB，因为 CDB 可以包含可插拔数据库（PDB），在 RDS for Oracle 中称为租户数据库。PDB 是模式和对象的可移植集合，在应用程序看来是单独的数据库。

在创建 CDB 实例时创建初始租户数据库（PDB）。在 RDS for Oracle 中，您的客户端应用程序与 PDB 交互，而不是与 CDB 交互。使用 PDB 的体验基本上与使用非 CDB 的体验相同。

主题

CDB 架构的多租户配置

RDS for Oracle 支持 Oracle 多租户架构（也称为 CDB 架构）的多租户配置。在此配置中，您的 RDS for Oracle CDB 实例可以包含 1-30 个租户数据库，具体取决于数据库版本和任何所需的选项许可证。在 Oracle 数据库中，租户数据库是 PDB。您的数据库实例必须使用 Oracle 数据库版本 19.0.0.0.ru-2022-01.rur-2022.r1 或更高版本。

> **Note**
>
> Amazon RDS 配置之所以称为“多-租户”而不是“多租户”，是因为它是 RDS 平台的一项功能，而不仅仅是 Oracle 数据库引擎的功能。同样，RDS 术语“租户”是指 RDS 配置中的任何租户，而不仅仅是 Oracle PDB。在 RDS 文档中，未加连字号的术语“Oracle 多租户”仅指 Oracle 数据库 CDB 架构，该架构与本地部署和 RDS 部署均兼容。

您可以配置以下设置：

- 租户数据库名称
- 租户数据库主用户名
- 租户数据库主密码（可选与 Secrets Manager 集成）
- 租户数据库字符集
- 租户数据库国家字符集

租户数据库字符集可以不同于 CDB 字符集。这同样适用于国家字符集。创建初始租户数据库后，您可以使用 RDS API 创建、修改或删除租户数据库。CDB 名称原定设置为 `RDSCDB` 且无法更改。有关更多信息，请参阅[数据库实例的设置](#)和[修改 RDS for Oracle 租户数据库](#)。

CDB 架构的单租户配置

RDS for Oracle 支持 Oracle 多租户架构的旧配置，即单租户配置。在此配置中，RDS for Oracle CDB 实例只能包含一个租户（PDB）。之后无法创建更多 PDB。

CDB 的创建和转换选项

Oracle Database 21c 仅支持 CDB，而 Oracle Database 19c 同时支持 CDB 和非 CDB。所有 RDS for Oracle CDB 实例都支持多租户和单租户配置。

Oracle 数据库架构的创建、转换和升级选项

下表显示了用于创建和升级 RDS for Oracle 数据库的不同架构选项。

发布版本	数据库创建选项	架构转换选项	主要版本升级目标
Oracle Database 21c	仅限 CDB 架构	不适用	不适用
Oracle Database 19c	CDB 或非 CDB 架构	非 CDB 至 CDB 架构（2021 年 4 月 RU 或更高版本）	Oracle Database 21c CDB

如上表所示，在新的主要数据库版本中，您不能直接将非 CDB 升级到 CDB。但您可以将 Oracle Database 19c 非 CDB 转换为 Oracle Database 19c CDB，然后将 Oracle Database 19c CDB 升级为 Oracle Database 21c CDB。有关更多信息，请参阅 将 RDS for Oracle 非 CDB 转换为 CDB。

CDB 架构配置的转换选项

下表显示了用于转换 RDS for Oracle 数据库实例架构配置的不同选项。

当前架构和配置	转换为 CDB 架构的单租户配置	转换为 CDB 架构的多租户配置	转换为非 CDB 架构
非 CDB	支持	支持*	不适用
使用单租户配置的 CDB	不适用	支持	不支持
使用多租户配置的 CDB	不支持	不适用	不支持

* 您无法在单个操作中将非 CDB 转换为多租户配置。将非 CDB 转换为 CDB 后，CDB 将处于单租户配置。然后，您可以通过另一个操作将单租户配置转换为多租户配置。

CDB 中的用户账户和权限

在 Oracle 多租户架构中，所有用户账户都是普通用户或本地用户。CDB 普通用户是指在 CDB 根目录和每个现有和将来的 PDB 中已知单个身份和密码的数据库用户。相比之下，本地用户只存在于单个 PDB 中。

RDS 主用户是 PDB 中的本地用户账户，这是您在创建数据库实例时命名的。如果您创建新的用户账户，这些用户也将成为位于 PDB 中的本地用户。您不能使用任何用户账户创建新的 PDB 或修改现有 PDB 的状态。

`rdsadmin` 用户是普通用户账户。您可以运行此账户中存在的 RDS for Oracle 软件包，但无法以 `rdsadmin` 的身份登录。有关详细信息，请参阅 Oracle 文档中的关于普通用户和本地用户。

对于多租户和单租户配置中的主用户，您可以使用自行管理的凭证或由 Amazon Secrets Manager 管理的凭证。在单租户配置中，您可以使用实例级 CLI 命令，例如，对于托管式主密码使用 `create-db-instance`。在多租户配置中，您可以使用租户数据库命令，例如，对于托管式主密码使用 `create-tenant-database`。有关设置 Secrets Manager 集成的更多信息，请参阅使用 Secrets Manager 管理 RDS for Oracle 租户数据库的主用户密码。

CDB 中的参数组系列

CDB 有其自己的参数类系列和原定设置参数值。CDB 参数组系列如下所示：

- oracle-ee-cdb-21
- oracle-se2-cdb-21
- oracle-ee-cdb-19
- oracle-se2-cdb-19

RDS for Oracle CDB 的限制

RDS for Oracle 支持本地 CDB 中提供的一部分功能。

CDB 的限制

以下限制在 CDB 级别适用于 RDS for Oracle：

- 您无法连接到 CDB。您始终连接到租户数据库（PDB），而不是 CDB。就像为非 CDB 指定终端节点一样，应为 PDB 指定终端节点。唯一的区别是您指定 pdb_name 作为数据库名称，其中 pdb_name 是为 PDB 选择的名称。

- 您无法将使用多租户配置的 CDB 转换为使用单租户配置的 CDB。转换为多租户配置是单向不可逆的操作。
- 如果您的数据库实例使用低于 19.0.0.0.ru-2022-01.rur-2022.r1 的 Oracle 数据库版本，则您无法启用或者转换为多租户配置。
- 您不能将 RDS for Oracle CDB 与 ORDS 22 及更高版本结合使用。作为一种变通方法，您可以使用较低版本的 ORDS，也可以使用 Oracle Database 19c 非 CDB。
- 您不能在多租户配置中使用 Oracle Data Guard，但可以在单租户配置中使用 Oracle Data Guard。
- 您不能在 CDB 中使用数据库活动流。
- 您无法从 `CDB$ROOT` 中启用审计。您必须在每个 PDB 中单独启用审计。

租户数据库（PDB）限制

以下限制适用于 RDS for Oracle 多租户配置中的租户数据库：

- 您不能将租户数据库操作推迟到维护时段。所有更改都会立即生效。
- 您无法将租户数据库添加到使用单租户配置的 CDB。
- 您不能在单个操作中添加或修改多个租户数据库。一次只能添加或修改一个。
- 您不能将租户数据库名称修改为 `CDB$ROOT` 或 `PDB$SEED`。
- 如果租户数据库是 CDB 中唯一的租户，则无法将其删除。
- 并非所有数据库实例类类型都有足够的资源来支持 RDS for Oracle CDB 实例中的多个 PDB。PDB 数量的增加会影响较小实例类的性能和稳定性，并会增加大多数实例级操作（例如数据库升级）的时间。
- 不能使用多个 Amazon Web Services 账户在同一 CDB 中创建 PDB。PDB 必须归托管 PDB 的数据库实例的账户所有。
- CDB 中的所有 PDB 都使用相同的终端节点和数据库侦听器。
- PDB 级别不支持以下操作，但 CDB 级别支持以下操作：
  - 备份和恢复
  - 数据库升级
  - 维护操作
- PDB 级别不支持以下功能，但 CDB 级别支持以下功能：
  - 选项组（选项安装在您 CDB 实例上的所有 PDB 上）
  - 参数组（所有参数均来自与您的 CDB 实例关联的参数组）

- 本地 CDB 架构支持但 RDS for Oracle CDB 不支持的 PDB 级操作包括以下内容：

> **Note**
> 以下列表并非详尽内容。

  - 应用程序 PDB
  - 代理 PDB
  - 启动和停止 PDB
  - PDB 的拔出和插入

    要将数据移入或移出 CDB，请使用与非 CDB 相同的方法。有关迁移数据的更多信息，请参阅将数据导入到 Amazon RDS 上的 Oracle。
  - PDB 级别的设置选项

    PDB 将继承 CDB 选项组中的选项设置。有关设置选项的更多信息，请参阅Amazon RDS 的参数组。有关最佳实践，请参阅使用数据库参数组。
  - 在 PDB 中配置参数

    PDB 继承 CDB 的参数设置。有关设置选项的更多信息，请参阅向 Oracle 数据库实例添加选项。
  - 为同一 CDB 中的 PDB 配置不同的侦听器
  - Oracle 闪回功能

配置 RDS for Oracle CDB

配置 CDB 与配置非 CDB 类似。

主题

- 创建 RDS for Oracle CDB 实例
- 在 RDS for Oracle CDB 中连接到 PDB

创建 RDS for Oracle CDB 实例

在 RDS for Oracle 中，创建 CDB 实例与创建非 CDB 实例几乎完全相同。不同之处在于，您在创建数据库实例时选择了 Oracle 多租户架构，还选择了架构配置：多租户或单租户。如果您在创建使用多租

户配置的 CDB 时创建标签，则 RDS 会将标签传播到初始租户数据库。要创建 CDB，请使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API。

控制台

创建 CDB 实例

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在 Amazon RDS 控制台的右上角，选择要在其中创建 CDB 实例的 Amazon Web Services 区域。
3. 在导航窗格中，选择 Databases (数据库)。
4. 选择创建数据库。
5. 在选择数据库创建方法中，选择标准创建。
6. 在引擎选项下面，选择 Oracle。
7. 对于数据库管理类型，选择 Amazon RDS。
8. 对于架构设置，选择 Oracle 多租户架构。
9. 对于架构配置，请执行以下操作之一：
   - 选择多租户配置，继续执行下一步。
   - 选择单租户配置并跳至步骤 11。
10. （多租户配置）对于租户数据库设置，请进行以下更改：
    - 对于租户数据库名称，输入初始 PDB 的名称。PDB 名称必须与 CDB 名称不同，后者默认为 `RDSCDB`。
    - 对于租户数据库主用户名，输入 PDB 的主用户名。您不能使用租户数据库主用户名登录 CDB 本身。
    - 对于凭证管理，选择以下任一其它凭证管理选项：
      - 在 Amazon Secrets Manager 管理

        托管式密码适用于初始租户数据库，而不适用于实例。在选择加密密钥中，选择 Secrets Manager 创建的 KMS 密钥或您已创建的密钥。

> **Note**
>
> 我们建议将 Amazon Secrets Manager 作为管理凭证的最安全的方法。将收取额外费用。有关更多信息，请参阅 使用 Amazon RDS 和 Amazon Secrets Manager 管理密码。

- 自行管理

  要指定密码，请清除自动生成密码复选框（如果已选中）。在主密码和确认主密码中输入相同的密码。

- 在租户数据库字符集中，为 PDB 选择一个字符集。您可以选择与 CDB 字符集不同的租户数据库字符集。

  默认 PDB 字符集是 AL32UTF8。如果选择非默认 PDB 字符集，则创建 CDB 的速度可能会变慢。

> **Note**
>
> 您无法在创建操作中指定多个租户数据库。创建 CDB 时，它有一个 PDB。您可以在单独的操作中将 PDB 添加到现有的 CDB。

11. （单租户配置）根据数据库实例的设置中列出的选项选择所需的设置：

   - 在设置部分中，打开凭证设置。然后执行以下操作：

     i. 在主用户名中，输入 PDB 中本地用户的名称。您不能使用主用户名登录 CDB 根目录。

     ii. 对于凭证管理，选择以下任一其它凭证管理选项：

       - 在 Amazon Secrets Manager 管理

         在选择加密密钥中，选择 Secrets Manager 创建的 KMS 密钥或您已创建的密钥。

         > **Note**
         >
         > 我们建议将 Amazon Secrets Manager 作为管理凭证的最安全的方法。将收取额外费用。有关更多信息，请参阅 使用 Amazon RDS 和 Amazon Secrets Manager 管理密码。

- 自行管理

  要指定密码，请清除自动生成密码复选框（如果已选中）。在主密码和确认主密码中输入相同的密码。

12. 对于其余部分，请指定数据库实例设置。有关每个设置的信息，请参阅数据库实例的设置。
13. 选择创建数据库。

Amazon CLI

要使用多租户配置创建 CDB，请结合以下参数使用 create-db-instance 命令：

- `--db-instance-identifier`
- `--db-instance-class`
- `--engine { oracle-ee-cdb | oracle-se2-cdb }`
- `--master-username`
- `--master-user-password` 或 `--manage-master-user-password`
- `--multi-tenant`（对于单租户配置，不要指定 `multi-tenant` 或 `--no-multi-tenant`）
- `--allocated-storage`
- `--backup-retention-period`

有关每项设置的信息，请参阅 数据库实例的设置。

以下示例采用多租户配置创建了一个名为 *my-cdb-inst* 的 RDS for Oracle 数据库实例。如果您指定 `--no-multi-tenant` 或未指定 `--multi-tenant`，则默认 CDB 配置为单租户配置。引擎是 `oracle-ee-cdb`：一个指定 `oracle-ee` 和 `--multi-tenant` 但因错误失败的命令。初始租户数据库名为 *mypdb*。

Example

对于 Linux、macOS 或 Unix：

```
aws rds create-db-instance \
    --engine oracle-ee-cdb \
    --db-instance-identifier my-cdb-inst \
    --multi-tenant \
    --db-name mypdb \
    --allocated-storage 250 \
```

```
    --db-instance-class db.t3.large \
    --master-username pdb_admin \
    --manage-master-user-password \
    --backup-retention-period 3
```

对于 Windows：

```
aws rds create-db-instance ^
    --engine oracle-ee-cdb ^
    --db-instance-identifier my-cdb-inst ^
    --multi-tenant ^
    --db-name mypdb ^
    --allocated-storage 250 ^
    --db-instance-class db.t3.large ^
    --master-username pdb_admin ^
    --manage-master-user-password \ ^
    --backup-retention-period 3
```

> **Note**
>
> 作为安全最佳实践，请指定除此处所示提示以外的密码。

此命令生成类似于下述信息的输出。输出中不包括数据库名称、字符集、国家/地区字符集、主用户和主用户密钥。您可以使用 CLI 命令 `describe-tenant-databases` 查看此信息。

```
{
    "DBInstance": {
        "DBInstanceIdentifier": "my-cdb-inst",
        "DBInstanceClass": "db.t3.large",
        "MultiTenant": true,
        "Engine": "oracle-ee-cdb",
        "DBResourceId": "db-ABCDEFGJIJKLMNOPQRSTUVWXYZ",
        "DBInstanceStatus": "creating",
        "AllocatedStorage": 250,
        "PreferredBackupWindow": "04:59-05:29",
        "BackupRetentionPeriod": 3,
        "DBSecurityGroups": [],
        "VpcSecurityGroups": [
            {
                "VpcSecurityGroupId": "sg-0a1bcd2e",
                "Status": "active"
```

```
            }
        ],
        "DBParameterGroups": [
            {
                "DBParameterGroupName": "default.oracle-ee-cdb-19",
                "ParameterApplyStatus": "in-sync"
            }
        ],
        "DBSubnetGroup": {
            "DBSubnetGroupName": "default",
            "DBSubnetGroupDescription": "default",
            "VpcId": "vpc-1234567a",
            "SubnetGroupStatus": "Complete",
            ...
```

RDS API

要使用 Amazon RDS API 创建数据库实例，请调用 CreateDBInstance 操作。

有关每项设置的信息，请参阅 数据库实例的设置。

在 RDS for Oracle CDB 中连接到 PDB

您可以使用像 SQL*Plus 这样的实用程序连接到 PDB。要下载包含独立版本 SQL*Plus 的 Oracle 即时客户端，请参阅 Oracle 即时客户端下载。

要将 SQL*Plus 连接到 PDB，您需要以下信息：

- PDB 名称
- 数据库用户名和密码
- 您的数据库实例的端点
- 端口号

有关查找上述信息的信息，请参阅查找 RDS for Oracle 数据库实例的端点。

Example 使用 SQL*Plus 连接到您的 PDB

在以下示例中，用您的主用户替换 *master_user_name*。另外，请将端点替换为您的数据库实例，然后包含端口号和 Oracle SID。SID 值是您在创建数据库实例时指定的 PDB 的名称，而不是数据库实例标识符。

对于 Linux、macOS 或 Unix：

```
sqlplus 'master_user_name@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=endpoint)
(PORT=port))(CONNECT_DATA=(SID=pdb_name)))'
```

对于 Windows：

```
sqlplus master_user_name@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=endpoint)
(PORT=port))(CONNECT_DATA=(SID=pdb_name)))
```

您应该可以看到类似于如下所示的输出内容。

```
SQL*Plus: Release 19.0.0.0.0 Production on Mon Aug 21 09:42:20 2021
```

在您输入用户的密码之后，将出现 SQL 提示。

```
SQL>
```

Note

较短格式连接字符串 (Easy connect 或 EZCONNECT) (如 `sqlplus username/password@LONGER-THAN-63-CHARS-RDS-ENDPOINT-HERE:1521/database-identifier`) 可能会遇到最大字符限制，不应用于连接。

备份和还原 CDB

您可以使用 RDS 数据库快照或 Recovery Manager（RMAN）备份和还原 CDB。

使用数据库快照备份和还原 CDB

数据库快照在 CDB 和非 CDB 架构中的工作原理类似。主要区别如下所示：

- 还原 CDB 的数据库快照后，无法重命名 CDB。CDB 名为 `RDSCDB` 且无法更改。
- 还原 CDB 的数据库快照后，无法重命名 PDB。您可以使用 modify-tenant-database 命令修改 PDB 名称。
- 要在快照中查找租户数据库，请使用 CLI 命令 describe-db-snapshot-tenant-databases。

- 您无法直接与使用多租户架构配置的 CDB 快照中的租户数据库进行交互。如果还原数据库快照，则会还原其所有租户数据库。
- RDS for Oracle 将租户数据库上的标签隐式复制到数据库快照中的租户数据库。还原租户数据库后，标签会显示在还原后的数据库中。
- 如果您还原数据库快照并使用 `--tags` 参数指定新标签，则新标签将覆盖所有现有标签。
- 如果您为带有标签的 CDB 实例拍摄数据库快照，并且您指定了 `--copy-tags-to-snapshot`，则 RDS for Oracle 会将标签从租户数据库复制到快照中的租户数据库。

有关更多信息，请参阅 Oracle Database 注意事项。

使用 RMAN 备份和还原 CDB

要了解如何使用 RMAN 备份和还原 CDB 或个人租户数据库，请参阅执行 Oracle 数据库实例的常见 RMAN 任务。

将 RDS for Oracle 非 CDB 转换为 CDB

您可以使用 `modify-db-instance` 命令将 Oracle 数据库的架构从非 CDB 架构更改为 Oracle 多租户架构，也称为 CDB 架构。在大多数情况下，这种方法比创建新的 CDB 并导入数据更可取。转换操作会导致停机。

升级数据库引擎版本时，不能在同一操作中更改数据库架构。因此，要将 Oracle Database 19c 非 CDB 升级到 Oracle Database 21c CDB，您首先需要执行一个步骤来将非 CDB 转换为 CDB，然后再执行一个步骤来将 19c CDB 升级到 21c CDB。

非 CDB 转换操作有以下要求：

- 必须为数据库引擎类型指定 `oracle-ee-cdb` 或 `oracle-se2-cdb`。只支持这两个值。
- 您的数据库引擎必须使用具有 2021 年 4 月或更高版本的版本更新（RU）的 Oracle Database 19c。

该操作具有以下限制：

- 您无法将 CDB 转换为非 CDB。您只能将非 CDB 转换为 CDB。
- 您无法通过单次 `modify-db-instance` 调用将非 CDB 转换为多租户配置。将非 CDB 转换为 CDB 后，您的 CDB 将处于单租户配置。要将单租户配置转换为多租户配置，请再次运行 `modify-db-instance`。有关更多信息，请参阅 将单租户配置转换为多租户配置。

- 您无法转换启用了 Oracle Data Guard 的主数据库或副本数据库。要转换具有只读副本的非 CDB，请先删除所有只读副本。
- 您无法在同一操作中升级数据库引擎版本，并将非 CDB 转换为 CDB。

转换非 CDB 前，请考虑以下事项：

- 选项和参数组的注意事项与升级数据库引擎的注意事项相同。有关更多信息，请参阅 Oracle 数据库升级注意事项。
- 您可以在单个操作中将使用托管式主密码的现有非 CDB 实例转换为单租户实例。单租户实例继承托管式密码。
- 如果数据库实例安装了 OEMAGENT 选项，则最佳实践是在转换非 CDB 之前移除此选项。在非 CDB 转换为 CDB 之后，请重新安装该选项。有关更多信息，请参阅 适用于 Enterprise Manager Cloud Control 的 Oracle Management Agent。
- 在转换过程中，RDS 将联机重做日志大小重置为默认的 128M。

控制台

将非 CDB 转换为 CDB

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在 Amazon RDS 控制台的右上角，选择数据库实例所在的 Amazon Web Services 区域。
3. 在导航窗格中，选择数据库，然后选择要转换为 CDB 实例的非 CDB 实例。
4. 选择 Modify(修改)。
5. 对于架构设置，选择 Oracle 多租户架构。转换后，您的 CDB 将处于单租户配置。
6. （可选）对于数据库参数组，请为您的 CDB 实例选择一个新的参数组。转换数据库实例时和升级数据库实例时，适用的参数组注意事项相同。有关更多信息，请参阅 参数组注意事项。
7. (可选）对于选项组，为 CDB 实例选择一个新的选项组。转换数据库实例时和升级数据库实例时，适用的选项组注意事项相同。有关更多信息，请参阅 选项组注意事项。
8. （可选）对于凭证管理，请选择在 Amazon Secrets Manager 中管理或自行管理。有关更多信息，请参阅 使用 Secrets Manager 管理数据库实例的主用户密码。
9. 当所有更改都达到您的要求时，选择继续并查看修改摘要。
10. （可选）选择立即应用以立即应用更改。选择此选项在某些情况下可能导致停机。有关更多信息，请参阅“使用计划修改设置”。

11. 在确认页面上，检查您的更改。如果更改正确无误，请选择修改数据库实例。

    也可以选择 Back (返回) 编辑您的更改，或选择 Cancel (取消) 取消更改。

Amazon CLI

要将数据库实例上的非 CDB 转换为采用单租户配置的 CDB，请在 Amazon CLI 命令 modify-db-instance 中将 `--engine` 设置为 `oracle-ee-cdb` 或 `oracle-se2-cdb`。有关更多信息，请参阅 数据库实例的设置。

以下示例转换名为 *my-non-cdb* 的数据库实例，并指定自定义选项组和参数组。该命令还支持使用 Secrets Manager 管理密码。

Example

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier my-non-cdb \
    --engine oracle-ee-cdb \
    --option-group-name custom-option-group \
    --db-parameter-group-name custom-parameter-group \
    --manage-master-user-password
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier my-non-cdb ^
    --engine oracle-ee-cdb ^
    --option-group-name custom-option-group ^
    --db-parameter-group-name custom-parameter-group ^
    --manage-master-user-password
```

RDS API

要将非 CDB 转换为 CDB，请在 RDS API 操作 ModifyDBInstance 中指定 `Engine`。

将单租户配置转换为多租户配置

您可以将 RDS for Oracle CDB 的架构从单租户配置修改为多租户配置。转换前后，您的 CDB 包含单租户数据库（PDB）。数据库实例的标签会传播到在转换期间创建的初始租户数据库。

在开始之前，请确保您的 IAM 策略有权创建租户数据库。在转换过程中，RDS for Oracle 会将以下元数据迁移到新的租户数据库：

- 主用户名
- 托管式主密码（如果源 CDB 与 Secrets Manager 集成）
- 数据库名称
- 字符集
- 国家字符集

在转换之前，可以使用 `describe-db-instances` 命令查看上述信息。转换完成后，您可以使用 `describe-tenant-database` 命令查看信息。

从单租户到多租户的转换具有以下限制：

- 转换为多租户配置后，您以后无法转换回单租户配置。该转换是不可逆的。
- 您无法转换启用了 Oracle Data Guard 的主数据库或副本数据库。
- 您无法在同一操作中升级数据库引擎版本，并转换为多租户配置。
- 在转换过程中，您无法启用或禁用托管式主用户密码。

控制台

将使用单租户配置的 CDB 转换为多租户配置

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在 Amazon RDS 控制台的右上角，选择数据库实例所在的 Amazon Web Services 区域。
3. 在导航窗格中，选择数据库，然后选择要转换为 CDB 实例的非 CDB 实例。
4. 选择 Modify(修改)。
5. 对于架构设置，选择 Oracle 多租户架构。
6. 对于架构配置，请选择多租户配置。
7. （可选）对于数据库参数组，请为您的 CDB 实例选择一个新的参数组。转换数据库实例时和升级数据库实例时，适用的参数组注意事项相同。
8. (可选）对于选项组，为 CDB 实例选择一个新的选项组。转换数据库实例时和升级数据库实例时，适用的选项组注意事项相同。

9. 当所有更改都达到您的要求时，选择继续并查看修改摘要。
10. 选择立即应用。切换到多租户配置时需要此选项。注意，选择此选项在某些情况下可能导致停机。
11. 在确认页面上，检查您的更改。如果更改正确无误，请选择修改数据库实例。

    也可以选择 Back (返回) 编辑您的更改，或选择 Cancel (取消) 取消更改。

Amazon CLI

要将使用单租户配置的 CDB 转换为多租户配置，请在命令 modify-db-instance 中指定 Amazon CLI。

以下示例将名为 `my-st-cdb` 的数据库实例从单租户配置转换为多租户配置。`--apply-immediately` 选项是必需的。

Example

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance --region us-east-1\
    --db-instance-identifier my-st-cdb \
    --multi-tenant \
    --apply-immediately
```

对于 Windows：

```
aws rds modify-db-instance --region us-east-1 ^
    --db-instance-identifier my-st-cdb ^
    --multi-tenant ^
    --apply-immediately
```

输出如下所示。

```
{
    "DBInstance": {
        "DBInstanceIdentifier": "my-st-cdb",
        "DBInstanceClass": "db.r5.large",
        "MultiTenant": false,
        "Engine": "oracle-ee-cdb",
        "DBResourceId": "db-AB1CDE2FGHIJK34LMNOPRLXTXU",
        "DBInstanceStatus": "modifying",
        "MasterUsername": "admin",
```

```
        "DBName": "ORCL",
        ...
        "EngineVersion": "19.0.0.0.ru-2022-01.rur-2022-01.r1",
        "AutoMinorVersionUpgrade": true,
        "ReadReplicaDBInstanceIdentifiers": [],
        "LicenseModel": "bring-your-own-license",
        "OptionGroupMemberships": [
            {
                "OptionGroupName": "default:oracle-ee-cdb-19",
                "Status": "in-sync"
            }
        ],
        ...
        "PendingModifiedValues": {
            "MultiTenant": "true"
        }
    }
}
```

将 RDS for Oracle 租户数据库添加到您的 CDB 实例

在 RDS for Oracle 多租户配置中，租户数据库是 PDB。要添加租户数据库，请确保满足以下先决条件：

- 您的 CDB 已启用多租户配置。有关更多信息，请参阅 CDB 架构的多租户配置。
- 您具有创建租户数据库所需的 IAM 权限。

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 添加租户数据库。您不能通过单个操作添加多个租户数据库：必须一次添加一个。如果 CDB 启用了备份保留功能，则 Amazon RDS 会在数据库实例添加新的租户数据库之前和之后对其进行备份。

控制台

向数据库实例添加租户数据库

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在 Amazon RDS 控制台的右上角，选择要在其中创建租户数据库的 Amazon Web Services 区域。
3. 在导航窗格中，选择数据库。

4. 选择要向其中添加租户数据库的 CDB 实例。您的数据库实例必须使用 CDB 架构的多租户配置。
5. 选择操作，然后选择添加租户数据库。
6. 对于租户数据库设置，请执行以下操作：

   - 对于租户数据库名称，输入新 PDB 的名称。
   - 对于租户数据库主用户名，输入您 PDB 的主用户名。
   - 选择以下任一其它凭证管理选项：
     - 在 Amazon Secrets Manager 管理

       在选择加密密钥中，选择 Secrets Manager 创建的 KMS 密钥或您已创建的密钥。

       > **Note**
       >
       > 我们建议将 Amazon Secrets Manager 作为管理凭证的最安全的方法。将收取额外费用。有关更多信息，请参阅 使用 Amazon RDS 和 Amazon Secrets Manager 管理密码。

     - 自行管理

       要指定密码，请清除自动生成密码复选框（如果已选中）。在主密码和确认主密码中输入相同的密码。
     - 在其他配置下，在初始数据库名称中输入 PDB 的名称。您不能命名 CDB，它具有原定设置名称 `RDSCDB`。
   - 在租户数据库字符集中，为 PDB 选择一个字符集。默认值为 AL32UTF8。您可以选择不同于 CDB 字符集的 PDB 字符集。
   - 对于租户数据库国家字符集，请为 PDB 选择一个国家字符集。默认值为 AL32UTF8。国家字符集仅为使用 `NCHAR` 数据类型（`NCHAR`、`NVARCHAR2`、和 `NCLOB`）且不影响数据库元数据的列指定编码。

   有关上述设置的更多信息，请参阅数据库实例的设置。
7. 选择添加租户。

Amazon CLI

要使用 Amazon CLI 将租户数据库添加到您的 CDB，请结合以下必需参数使用 create-tenant-database 命令：

- `--db-instance-identifier`
- `--tenant-db-name`
- `--master-username`
- `--master-user-password`

以下示例在名为 *my-cdb-inst* 的 RDS for Oracle CDB 实例中创建一个名为 *mypdb2* 的租户数据库。PDB 的字符集是 UTF-16。

Example

对于 Linux、macOS 或 Unix：

```
aws rds create-tenant-database --region us-east-1 \
    --db-instance-identifier my-cdb-inst \
    --tenant-db-name mypdb2 \
    --master-username mypdb2-admin \
    --master-user-password mypdb2-pwd \
    --character-set-name UTF-16
```

对于 Windows：

```
aws rds create-tenant-database --region us-east-1 \
    --db-instance-identifier my-cdb-inst ^
    --tenant-db-name mypdb2 ^
    --master-username mypdb2-admin ^
    --master-user-password mypdb2-pwd ^
    --character-set-name UTF-16
```

该输出值看上去类似于以下内容。

```
...}
    "TenantDatabase" :
         {
            "DbiResourceId" : "db-abc123",
            "TenantDatabaseResourceId" : "tdb-bac567",
            "TenantDatabaseArn" : "arn:aws:rds:us-east-1:123456789012:db:my-cdb-
inst:mypdb2",
            "DBInstanceIdentifier" : "my-cdb-inst",
            "TenantDBName" : "mypdb2",
```

```
            "Status" : "creating",
            "MasterUsername" : "mypdb2",
            "CharacterSetName" : "UTF-16",
            ...
        }
}...
```

修改 RDS for Oracle 租户数据库

您只能修改 CDB 中租户数据库的 PDB 名称和主用户密码。请注意以下要求和限制：

- 要修改数据库实例中租户数据库的设置，租户数据库必须存在。
- 您无法在单个操作中修改多个租户数据库。一次只能修改一个租户数据库。
- 您不能将租户数据库的名称更改为 `CDB$ROOT` 或 `PDB$SEED`。

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 修改 PDB。

控制台

修改租户数据库的 PDB 名称或主密码

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在 Amazon RDS 控制台的右上角，选择要在其中创建租户数据库的 Amazon Web Services 区域。
3. 在导航窗格中，选择 Databases（数据库）。
4. 选择要修改其数据库名称或主用户密码的租户数据库。
5. 选择修改。
6. 对于租户数据库设置，请执行以下任意操作：
   - 对于租户数据库名称，输入新 PDB 的新名称。
   - 对于租户数据库主密码，输入新的密码。
7. 选择修改租户。

Amazon CLI

要使用 Amazon CLI 修改租户数据库，请结合以下参数调用 modify-tenant-database 命令：

- `--db-instance-identifier` *value*
- `--tenant-db-name` *value*
- `[--new-tenant-db-name` *value*`]`
- `[--master-user-password` *value*`]`

以下示例在数据库实例 my-cdb-inst 上将租户数据库 pdb1 重命名为 pdb-hr。

Example

对于 Linux、macOS 或 Unix：

```
aws rds modify-tenant-database --region us-east-1 \
    --db-instance-identifier my-cdb-inst \
    --tenant-db-name pdb1 \
    --new-tenant-db-name pdb-hr
```

对于 Windows：

```
aws rds modify-tenant-database --region us-east-1 ^
    --db-instance-identifier my-cdb-inst ^
    --tenant-db-name pdb1 ^
    --new-tenant-db-name pdb-hr
```

此命令生成类似于下述信息的输出。

```
{
    "TenantDatabase" : {
        "DbiResourceId" : "db-abc123",
        "TenantDatabaseResourceId" : "tdb-bac567",
        "TenantDatabaseArn" : "arn:aws:rds:us-east-1:123456789012:db:my-cdb-inst:pdb1",
        "DBInstanceIdentifier" : "my-cdb-inst",
        "TenantDBName" : "pdb1",
        "Status" : "modifying",
        "MasterUsername" : "tenant-admin-user"
        "Port" : "6555",
        "CharacterSetName" : "UTF-16",
        "MaxAllocatedStorage" : "1000",
        "ParameterGroups": [
            {
                "ParameterGroupName": "pdb1-params",
                "ParameterApplyStatus": "in-sync"
```

```
            }
        ],
        "OptionGroupMemberships": [
            {
                "OptionGroupName": "pdb1-options",
                "Status": "in-sync"
            }
        ],
        "PendingModifiedValues": {
            "TenantDBName": "pdb-hr"
        }
    }
}
```

从您的 CDB 中删除 RDS for Oracle 租户数据库

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 删除租户数据库（PDB）。请考虑以下先决条件和限制：

- 租户数据库和数据库实例必须存在。
- 要成功删除，必须存在以下情况之一：
  - 租户数据库和数据库实例可用。

    > **Note**
    > 您可以拍摄最终快照，但前提是租户数据库和数据库实例在发出 `delete-tenant-database` 命令之前处于可用状态。

  - 正在创建租户数据库。
  - 数据库实例正在修改租户数据库。
- 您无法在单个操作中删除多个租户数据库。
- 如果租户数据库是 CDB 中唯一的租户，则无法将其删除。

控制台

删除租户数据库

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。

2. 在导航窗格中，选择数据库，然后选择要删除的租户数据库。
3. 对于 Actions，选择 Delete。
4. 要为数据库实例创建最终数据库快照，请选择是否创建最终快照?。
5. 如果选择创建最终快照，请输入最终快照名称。
6. 在框中输入 **delete me**。
7. 选择删除。

Amazon CLI

要使用 Amazon CLI 删除租户数据库，请结合以下参数调用 delete-tenant-database 命令：

- `--db-instance-identifier value`
- `--tenant-db-name value`
- `[--skip-final-snapshot | --no-skip-final-snapshot]`
- `[--final-snapshot-identifier value]`

以下示例从名为 *my-cdb-inst* 的 CDB 中删除名为 *pdb-test* 的租户数据库。默认情况下，该操作会创建最终快照。

Example

对于 Linux、macOS 或 Unix：

```
aws rds delete-tenant-database --region us-east-1 \
    --db-instance-identifier my-cdb-inst \
    --tenant-db-name pdb-test \
    --final-snapshot-identifier final-snap-pdb-test
```

对于 Windows：

```
aws rds delete-tenant-database --region us-east-1 ^
    --db-instance-identifier my-cdb-inst ^
    --tenant-db-name pdb-test ^
    --final-snapshot-identifier final-snap-pdb-test
```

此命令生成类似于下述信息的输出。

```
{
    "TenantDatabase" : {
        "DbiResourceId" : "db-abc123",
        "TenantDatabaseResourceId" : "tdb-bac456",
        "TenantDatabaseArn" : "arn:aws:rds:us-east-1:123456789012:db:my-cdb-inst:pdb-
test",
        "DBInstanceIdentifier" : "my-cdb-inst",
        "TenantDBName" : "pdb-test",
        "Status" : "deleting",
        "MasterUsername" : "pdb-test-admin"
        "Port" : "6555",
        "CharacterSetName" : "UTF-16",
        "MaxAllocatedStorage" : "1000",
        "ParameterGroups": [
            {
                "ParameterGroupName": "tenant-1-params",
                "ParameterApplyStatus": "in-sync"
            }
        ],
        "OptionGroupMemberships": [
            {
                "OptionGroupName": "tenant-1-options",
                "Status": "in-sync"
            }
        ]
    }
}
```

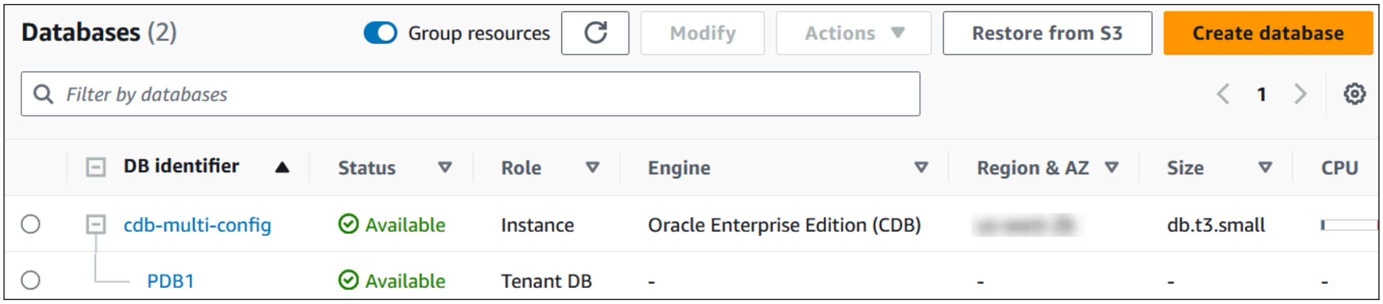
查看租户数据库详细信息

您可以像查看非 CDB 或 CDB 一样查看租户数据库的详细信息。

控制台

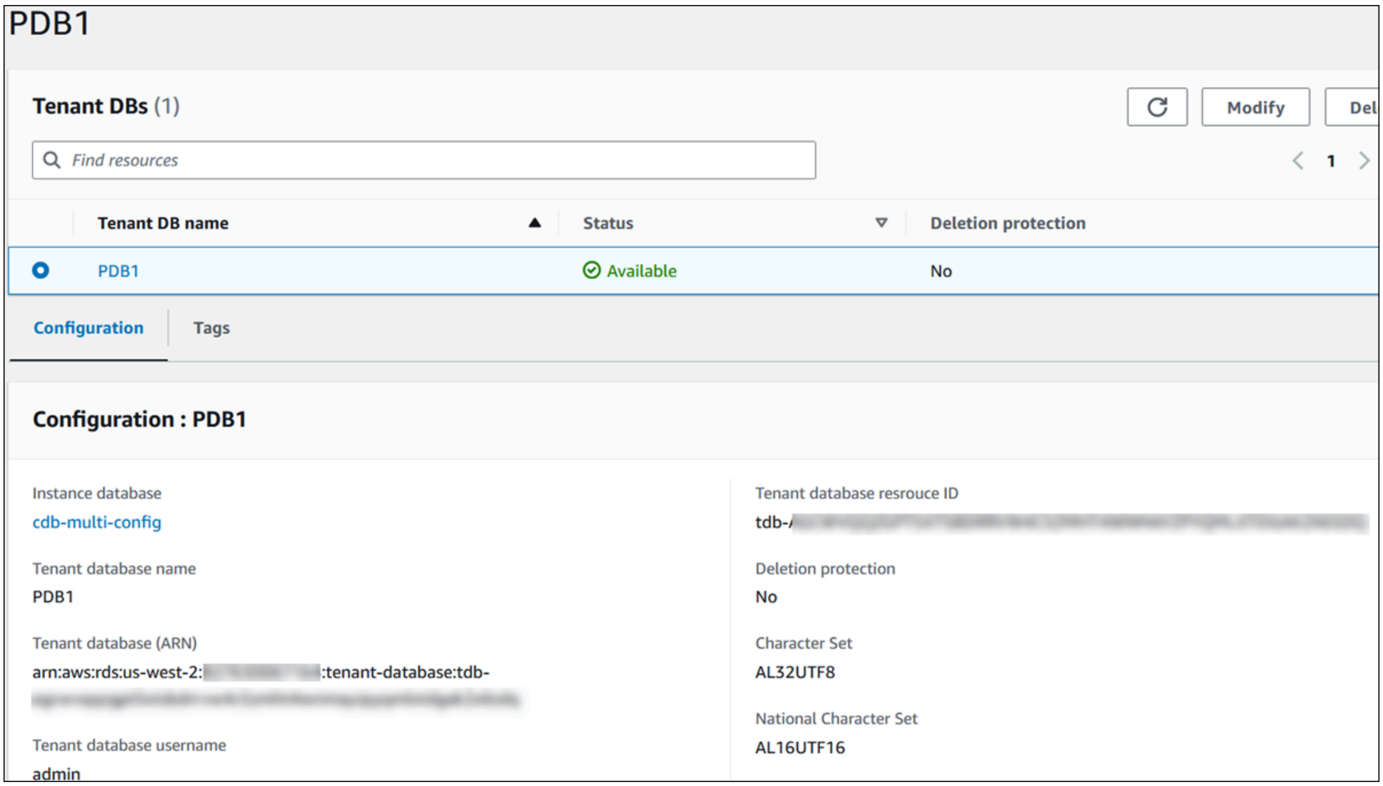
查看有关租户数据库的详细信息

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在 Amazon RDS 控制台的右上角，选择数据库实例所在的 Amazon Web Services 区域。
3. 在导航窗格中，选择 Databases（数据库）。



在上图中，唯一租户数据库（PDB）显示为数据库实例的子数据库。

4. 选择租户数据库的名称。



Amazon CLI

要查看有关 PDB 的详细信息，请使用 Amazon CLI 命令 describe-tenant-databases。

以下示例描述了指定区域中的所有租户数据库。

Example

对于 Linux、macOS 或 Unix：

```
aws rds describe-tenant-databases --region us-east-1
```

对于 Windows：

```
aws rds describe-tenant-databases --region us-east-1
```

此命令生成类似于下述信息的输出。

```
    "TenantDatabases" : [
         {
            "DBInstanceIdentifier" : "my-cdb-inst",
            "TenantDBName" : "pdb-test",
            "Status" : "available",
            "MasterUsername" : "pdb-test-admin",
            "DbiResourceId" : "db-abc123",
            "TenantDatabaseResourceId" : "tdb-bac456",
            "TenantDatabaseArn" : "arn:aws:rds:us-east-1:123456789012:db:my-cdb-
inst:pdb-test",
            "CharacterSetName": "AL32UTF8",
            "NcharCharacterSetName": "AL16UTF16",
            "DeletionProtection": false,
            "PendingModifiedValues": {
                 "MasterUserPassword": "****"
            },
            "TagList": []
         },
         {

            "DBInstanceIdentifier" : "my-cdb-inst2",
            "TenantDBName" : "pdb-dev",
            "Status" : "modifying",
            "MasterUsername" : "masterrdsuser"
            "DbiResourceId" : "db-xyz789",
            "TenantDatabaseResourceId" : "tdb-ghp890",
            "TenantDatabaseArn" : "arn:aws:rds:us-east-1:123456789012:db:my-cdb-
inst2:pdb-dev",
            "CharacterSetName": "AL32UTF8",
            "NcharCharacterSetName": "AL16UTF16",
            "DeletionProtection": false,
            "PendingModifiedValues": {
                 "MasterUserPassword": "****"
            },
            "TagList": []
         },
         ... other truncated data
```

以下示例描述了指定区域中数据库实例 my-cdb-inst 上的租户数据库。

Example

对于 Linux、macOS 或 Unix：

```
aws rds describe-tenant-databases --region us-east-1 \
    --db-instance-identifier my-cdb-inst
```

对于 Windows：

```
aws rds describe-tenant-databases --region us-east-1 ^
    --db-instance-identifier my-cdb-inst
```

此命令生成类似于下述信息的输出。


```
{
    "TenantDatabase": {
        "TenantDatabaseCreateTime": "2023-10-19T23:55:30.046Z",
        "DBInstanceIdentifier": "my-cdb-inst",
        "TenantDBName": "pdb-hr",
        "Status": "creating",
        "MasterUsername": "tenant-admin-user",
        "DbiResourceId": "db-abc123",
        "TenantDatabaseResourceId": "tdb-bac567",
        "TenantDatabaseARN": "arn:aws:rds:us-west-2:579508833180:pdb-hr:tdb-
abcdefghi1jklmno2p3qrst4uvw5xy6zabc7defghi8jklmn90op",
        "CharacterSetName": "AL32UTF8",
        "NcharCharacterSetName": "AL16UTF16",
        "DeletionProtection": false,
        "PendingModifiedValues": {
            "MasterUserPassword": "****"
        },
        "TagList": [
            {
                "Key": "TEST",
                "Value": "testValue"
            }
        ]
    }
}
```

以下示例描述了美国东部（弗吉尼亚州北部）区域中数据库实例 my-cdb-inst 上的租户数据库 pdb1。

Example

对于 Linux、macOS 或 Unix：

```
aws rds describe-tenant-databases --region us-east-1 \
--db-instance-identifier my-cdb-inst \
--tenant-db-name pdb1
```

对于 Windows：

```
aws rds describe-tenant-databases --region us-east-1 ^
--db-instance-identifier my-cdb-inst ^
--tenant-db-name pdb1
```

此命令生成类似于下述信息的输出。

```
{
    "TenantDatabases" : [
        {
            "DbiResourceId" : "db-abc123",
            "TenantDatabaseResourceId" : "tdb-bac567",
            "TenantDatabaseArn" : "arn:aws:rds:us-east-1:123456789012:db:my-cdb-
inst:pdb1"
            "DBInstanceIdentifier" : "my-cdb-inst",
            "TenantDBName" : "pdb1",
            "Status" : "ACTIVE",
            "MasterUsername" : "masterawsuser"
            "Port" : "1234",
            "CharacterSetName": "UTF-8",
            "ParameterGroups": [
                {
                    "ParameterGroupName": "tenant-custom-pg",
                    "ParameterApplyStatus": "in-sync"
                }
            ],
            {
            "OptionGroupMemberships": [
                {
                    "OptionGroupName": "tenant-custom-og",
```

```
                    "Status": "in-sync"
                }
            ]
        }
    ]
}
```

升级 CDB

您可以将 CDB 升级到其他 Oracle 数据库版本。例如，您可以将 Oracle Database 19c CDB 升级为 Oracle Database 21c CDB。在升级期间，您无法更改数据库架构。因此，您不能将非 CDB 升级为 CDB 或将 CDB 升级为非 CDB。

将 CDB 升级为 CDB 的过程与将非 CDB 升级为非 CDB 的过程相同。有关更多信息，请参阅 升级 RDS for Oracle 数据库引擎。

管理您的 RDS for Oracle 数据库实例

以下是您使用 RDS for Oracle 数据库实例执行的常见管理任务。对于所有 RDS 数据库实例，有些任务是相同的。其他任务是特定于 RDS for Oracle 的。

以下任务是所有 RDS 数据库的共同任务，但 Oracle 数据库具有特殊的注意事项。例如，您可以使用 Oracle 客户端 SQL*Plus 和 SQL Developer 连接到 Oracle 数据库。

任务区域	相关文档
实例类、存储和 PIOPS 如果您正在创建生产实例，请了解实例类、存储类型和预置 IOPS 在 Amazon RDS 中的工作原理。	RDS for Oracle 数据库实例类 Amazon RDS 存储类型
多可用区部署 生产数据库实例应使用多可用区部署。多可用区部署为数据库实例提供了更高的可用性、数据持久性和容错能力。	配置和管理 Amazon RDS 的多可用区部署
Amazon VPC 如果您的 Amazon 账户具有原定设置虚拟私有云（VPC），则自动在原定设置 VPC 中创建数据库实例。如果您的账户没有默认 VPC，并且您希望数据库实例位于 VPC 中，则必须在创建实例之前创建 VPC 和子网组。	在 VPC 中使用数据库实例
安全组 默认情况下，数据库实例使用阻止访问的防火墙。确保使用正确的 IP 地址和网络配置创建安全组来访问数据库实例。	使用安全组控制访问权限
参数组 如果数据库实例需要特定数据库参数，则在创建数据库实例前创建参数组。	Amazon RDS 的参数组
选项组	向 Oracle 数据库实例添加选项

任务区域	相关文档
如果数据库实例需要特定的数据库选项，则在创建数据库实例前创建选项组。	
连接到您的数据库实例 在创建安全组并将其与数据库实例关联后，您可以使用任何标准 SQL 客户端应用程序（如 Oracle SQL*Plus）连接到数据库实例。	连接到 Oracle 数据库实例
备份和还原 您可以配置数据库实例以采用自动备份或采用手动快照，然后从备份或快照还原实例。	备份、还原和导出数据
监控 您可以使用 CloudWatch Amazon RDS 指标、事件和增强监控来监控 Oracle 数据库实例。	在 Amazon RDS 控制台中查看指标 查看 Amazon RDS 事件
日志文件 您可以访问 Oracle 数据库实例的日志文件。	监控 Amazon RDS 日志文件

您在下面可以找到 RDS Oracle 常见 DBA 任务的特定于 Amazon RDS 实施的描述。为了提供托管服务体验，Amazon RDS 未提供数据库实例的 Shell 访问权限。此外，RDS 还限制了对需要高级权限的某些系统过程和表的访问。在许多任务中，您可以运行 `rdsadmin` 程序包，这是一个特定于 Amazon RDS 的工具，可用于管理您的数据库。

以下是运行 Oracle 的数据库实例的常见 DBA 任务：

- 系统任务

断开会话连接	Amazon RDS 方法：`rdsadmin.rdsadmin_util.disconnect` Oracle 方法：`alter system disconnect session`
终止会话	Amazon RDS 方法：`rdsadmin.rdsadmin_util.kill`

	Oracle 方法：`alter system kill session`
[在会话中取消 SQL 语句]()	Amazon RDS 方法：`rdsadmin.rdsadmin_util.cancel` Oracle 方法：`alter system cancel sql`
[启用和禁用受限制的会话]()	Amazon RDS 方法：`rdsadmin.rdsadmin_util.restricted_session` Oracle 方法：`alter system enable restricted session`
[刷新共享池]()	Amazon RDS 方法：`rdsadmin.rdsadmin_util.flush_shared_pool` Oracle 方法：`alter system flush shared_pool`
[刷新缓冲区缓存]()	Amazon RDS 方法：`rdsadmin.rdsadmin_util.flush_buffer_cache` Oracle 方法：`alter system flush buffer_cache`
[向 SYS 对象授予 SELECT 或 EXECUTE 权限]()	Amazon RDS 方法：`rdsadmin.rdsadmin_util.grant_sys_object` Oracle 方法：`grant`
[撤销 SYS 对象的 SELECT 或 EXECUTE 权限]()	Amazon RDS 方法：`rdsadmin.rdsadmin_util.revoke_sys_object` Oracle 方法：`revoke`
[管理 Oracle 数据库实例的 RDS_X$ 视图]()	Amazon RDS 方法：`rdsadmin.rdsadmin_util.create_sys_x$_view` Oracle 方法：`CREATE VIEW`
[向非主用户授予权限]()	Amazon RDS 方法：`grant`

创建自定义函数来验证密码	Amazon RDS 方法：`rdsadmin.rdsadmin_password_verify.create_verify_function` Amazon RDS 方法：`rdsadmin.rdsadmin_password_verify.create_passthrough_verify_fcn`
设置自定义 DNS 服务器	—
列出允许的系统诊断事件	Amazon RDS 方法：`rdsadmin.rdsadmin_util.list_allowed_system_events` Oracle 方法：—
设置系统诊断事件	Amazon RDS 方法：`rdsadmin.rdsadmin_util.set_allowed_system_events` Oracle 方法：`ALTER SYSTEM SET EVENTS 'set_event_clause'`
列出设置的系统诊断事件	Amazon RDS 方法：`rdsadmin.rdsadmin_util.list_set_system_events` Oracle 方法：`ALTER SESSION SET EVENTS 'IMMEDIATE EVENTDUMP(SYSTEM)'`
取消设置系统诊断事件	Amazon RDS 方法：`rdsadmin.rdsadmin_util.unset_system_event` Oracle 方法：`ALTER SYSTEM SET EVENTS 'unset_event_clause'`

- 数据库任务

更改数据库的全局名称	Amazon RDS 方法：`rdsadmin.rdsadmin_util.rename_global_name` Oracle 方法：`alter database rename`
创建表空间并配置其大小	Amazon RDS 方法：`create tablespace` Oracle 方法：`alter database`
设置默认表空间	Amazon RDS 方法：`rdsadmin.rdsadmin_util.alter_default_tablespace` Oracle 方法：`alter database default tablespace`
设置默认临时表空间	Amazon RDS 方法：`rdsadmin.rdsadmin_util.alter_default_temp_tablespace` Oracle 方法：`alter database default temporary tablespace`
在实例存储上创建临时表空间	Amazon RDS 方法：`rdsadmin.rdsadmin_util.create_inst_store_tmp_tblspace` Oracle 方法：`create temporary tablespace`
通过检查点检验数据库	Amazon RDS 方法：`rdsadmin.rdsadmin_util.checkpoint` Oracle 方法：`alter system checkpoint`
设置分布式恢复	Amazon RDS 方法：`rdsadmin.rdsadmin_util.enable_distr_recovery` Oracle 方法：`alter system enable distributed recovery`

设置数据库时区	Amazon RDS 方法：`rdsadmin.rdsadmin_util.alter_db_time_zone` Oracle 方法：`alter database set time_zone`
使用 Oracle 外部表	—
使用 Automatic Workload Repository (AWR) 生成性能报告	Amazon RDS 方法：`rdsadmin.rdsadmin_diagnostic_util` 过程 Oracle 方法：`dbms_workload_repository` 程序包
在 VPC 中调整用于数据库实例的数据库链接	—
设置数据库实例的默认版本	Amazon RDS 方法：`rdsadmin.rdsadmin_util.alter_default_edition` Oracle 方法：`alter database default edition`
为 SYS.AUD$ 表启用审核	Amazon RDS 方法：`rdsadmin.rdsadmin_master_util.audit_all_sys_aud_table` Oracle 方法：`audit`
为 SYS.AUD$ 表禁用审核	Amazon RDS 方法：`rdsadmin.rdsadmin_master_util.noaudit_all_sys_aud_table` Oracle 方法：`noaudit`
清理中断的在线索引构建	Amazon RDS 方法：`rdsadmin.rdsadmin_dbms_repair.online_index_clean` Oracle 方法：`dbms_repair.online_index_clean`
跳过损坏的数据块	Amazon RDS 方法：几个 `rdsadmin.rdsadmin_dbms_repair` 过程 Oracle 方法：`dbms_repair` 程序包

调整表空间、数据文件和临时文件的大小	Amazon RDS 方法：`rdsadmin.rdsadmin_util.resize_temp_tablespace`、`rdsadmin.rdsadmin_util.resize_tempfile` 或 `rdsadmin.rdsadmin_util.autoextend_tempfile` 过程 `rdsadmin.rdsadmin_util.resize_datafile` 或 `rdsadmin.rdsadmin_util.autoextend_datafile` 过程 Oracle 方法：—
清除回收站	Amazon RDS 方法：`EXEC rdsadmin.rdsadmin_util.purge_dba_recyclebin` Oracle 方法：`purge dba_recyclebin`
为全部编写设置原定设置显示值	Amazon RDS 方法：`EXEC rdsadmin.rdsadmin_util.dbms_redact_upd_full_rdct_val` Oracle 方法：`exec dbms_redact.UPDATE_FULL_REDACTION_VALUES`

- 日志任务

设置强制日志记录	Amazon RDS 方法：`rdsadmin.rdsadmin_util.force_logging` Oracle 方法：`alter database force logging`

设置补充日志记录	Amazon RDS 方法：`rdsadmin.rdsadmin_util.alter_supplemental_logging` Oracle 方法：`alter database add supplemental log`
切换联机日志文件	Amazon RDS 方法：`rdsadmin.rdsadmin_util.switch_logfile` Oracle 方法：`alter system switch logfile`
添加联机重做日志	Amazon RDS 方法：`rdsadmin.rdsadmin_util.add_logfile`
删除联机重做日志	Amazon RDS 方法：`rdsadmin.rdsadmin_util.drop_logfile`
调整联机重做日志的大小	—
保留存档重做日志	Amazon RDS 方法：`rdsadmin.rdsadmin_util.set_configuration`

从 Amazon S3 下载存档的重做日志	Amazon RDS 方法：`rdsadmin.rdsadmin_archive_log_download.download_log_with_seqnum` Amazon RDS 方法：`rdsadmin.rdsadmin_archive_log_download.download_logs_in_seqnum_range`
访问在线和存档的重做日志	Amazon RDS 方法：`rdsadmin.rdsadmin_master_util.create_archivelog_dir` Amazon RDS 方法：`rdsadmin.rdsadmin_master_util.create_onlinelog_dir`

- RMAN 任务

在 RDS for Oracle 中验证数据库文件	Amazon RDS 方法：`rdsadmin_rman_util.` *`procedure`* Oracle 方法：`RMAN VALIDATE`

启用和禁用数据块更改跟踪	Amazon RDS 方法：`rdsadmin_rman_util. procedure` Oracle 方法：`ALTER DATABASE`
反复检查存档重做日志	Amazon RDS 方法：`rdsadmin_rman_util.crosscheck_archivelog` Oracle 方法：`RMAN BACKUP`
备份归档的重做日志文件	Amazon RDS 方法：`rdsadmin_rman_util. procedure` Oracle 方法：`RMAN BACKUP`
执行完整数据库备份	Amazon RDS 方法：`rdsadmin_rman_util.backup_database_full` Oracle 方法：`RMAN BACKUP`
执行增量数据库备份	Amazon RDS 方法：`rdsadmin_rman_util.backup_database_incremental` Oracle 方法：`RMAN BACKUP`

备份表空间	Amazon RDS 方法：`rdsadmin_rman_util.backup_database_tablespace` Oracle 方法：`RMAN BACKUP`

- Oracle 计划程序任务

修改 DBMS_SCHEDULER 作业	Amazon RDS 方法：`dbms_scheduler.set_attribute` Oracle 方法：`dbms_scheduler.set_attribute`
修改自动任务维护时段	Amazon RDS 方法：`dbms_scheduler.set_attribute` Oracle 方法：`dbms_scheduler.set_attribute`
设置 Oracle Scheduler 作业的时区	Amazon RDS 方法：`dbms_scheduler.set_scheduler_attribute` Oracle 方法：`dbms_scheduler.set_scheduler_attribute`

关闭 SYS 拥有的 Oracle 计划程序任务	Amazon RDS 方法：`rdsadmin.rdsadmin_dbms_scheduler.disable` Oracle 方法：`dbms_scheduler.disable`
开启 SYS 拥有的 Oracle 计划程序任务	Amazon RDS 方法：`rdsadmin.rdsadmin_dbms_scheduler.enable` Oracle 方法：`dbms_scheduler.enable`
修改 CALENDAR 类型的任务的 Oracle 计划程序重复间隔	Amazon RDS 方法：`rdsadmin.rdsadmin_dbms_scheduler.set_attribute` Oracle 方法：`dbms_scheduler.set_attribute`
修改 NAMED 类型的任务的 Oracle 计划程序重复间隔	Amazon RDS 方法：`rdsadmin.rdsadmin_dbms_scheduler.set_attribute` Oracle 方法：`dbms_scheduler.set_attribute`

在创建 Oracle 计划程序任务时关闭自动提交	Amazon RDS 方法：`rdsadmin.rdsadmin_dbms_scheduler.set_no_commit_flag` Oracle 方法：`dbms_isched.set_no_commit_flag`

- 诊断问题

列出事件	Amazon RDS 方法：`rdsadmin.rdsadmin_adrci_util.list_adrci_incidents` Oracle 方法：ADRCI 命令 `show incident`
列出问题	Amazon RDS 方法：`rdsadmin.rdsadmin_adrci_util.list_adrci_problem` Oracle 方法：ADRCI 命令 `show problem`

[创建事件包]()	Amazon RDS 方法：`rdsadmin.rdsadmin_adrci_util.create_adrci_package` Oracle 方法：ADRCI 命令 `ips create package`
[显示跟踪文件]()	Amazon RDS 方法：`rdsadmin.rdsadmin_adrci_util.show_adrci_tracefile` Oracle 方法：ADRCI 命令 `show tracefile`

- [其他任务]()

[在主数据存储空间中创建和删除目录]()	Amazon RDS 方法：`rdsadmin.rdsadmin_util.create_directory` Oracle 方法：`CREATE DIRECTORY` Amazon RDS 方法：`rdsadmin.rdsadmin_util.drop_directory` Oracle 方法：`DROP DIRECTORY`

列出数据库实例目录中的文件	Amazon RDS 方法：`rdsadmin.rds_file_util.listdir` Oracle 方法：—
读取数据库实例目录中的文件	Amazon RDS 方法：`rdsadmin.rds_file_util.read_text_file` Oracle 方法：—
访问 Opatch 文件	Amazon RDS 方法：`rdsadmin.rds_file_util.read_text_file` 或 `rdsadmin.tracefile_listing` Oracle 方法：`opatch`
设置顾问任务参数	Amazon RDS 方法：`rdsadmin.rdsadmin_util.advisor_task_set_parameter` Oracle 方法：各种存储的程序包过程
禁用 AUTO_STATS_ADVISOR_TASK	Amazon RDS 方法：`rdsadmin.rdsadmin_util.advisor_task_drop` Oracle 方法：—

重新启用 AUTO_STATS_ADVISOR_TASK	Amazon RDS 方法：`rdsadmin.rdsadmin_util.dbms_stats_init` Oracle 方法：—

您还可以使用 Amazon RDS 过程将 Amazon S3 与 Oracle 集成在一起以及运行 OEM Management Agent 数据库任务。有关更多信息，请参阅Amazon S3 集成 和管理 Management Agent。

执行 Oracle 数据库实例的常见系统任务

下文中介绍了如何在运行 Oracle 的 Amazon RDS 数据库实例上，执行与系统相关的特定常见 DBA 任务。为了提供托管服务体验，Amazon RDS 不允许通过 shell 访问数据库实例，而仅限访问某些需要高级权限的系统过程和表。

主题

断开会话连接

要终止专用服务器进程以断开当前会话连接，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_util.disconnect`。`disconnect` 过程具有以下参数。

参数名称	数据类型	默认值	必需	描述
sid	number	—	是	会话标识符。
serial	number	—	是	会话的序列号。
method	varchar	'IMMEDIATE'	否	有效值为 'IMMEDIATE' 或 'POST_TRANSACTION' 。

以下示例断开一个会话。

```
begin
    rdsadmin.rdsadmin_util.disconnect(
        sid    => sid,
        serial => serial_number);
end;
/
```

要获取会话标识符和会话序列号，请查询 V$SESSION 视图。以下示例获取用户 AWSUSER 的所有会话。

```
SELECT SID, SERIAL#, STATUS FROM V$SESSION WHERE USERNAME = 'AWSUSER';
```

数据库必须处于打开状态，才能使用此方法。有关断开会话连接的更多信息，请参阅 Oracle 文档中的 ALTER SYSTEM。

终止会话

要终止会话，请使用 Amazon RDS 过程 rdsadmin.rdsadmin_util.kill。kill 过程具有以下参数。

参数名称	数据类型	默认值	必需	描述
sid	number	—	是	会话标识符。
serial	number	—	是	会话的序列号。

参数名称	数据类型	默认值	必需	描述
method	varchar	null	否	有效值为 'IMMEDIATE' 或 'PROCESS' 。如果您指定 IMMEDIATE ，其效果与运行以下语句相同： `ALTER SYSTEM KILL SESSION 'sid,serial#' IMMEDIATE` 如果您指定 PROCESS，将终止与会话关联的进程。仅当终止使用 IMMEDIATE 的会话失败时，才指定 PROCESS。

要获取会话标识符和会话序列号，请查询 V$SESSION 视图。以下示例获取用户 *AWSUSER* 的所有会话。

```
SELECT SID, SERIAL#, STATUS FROM V$SESSION WHERE USERNAME = 'AWSUSER';
```

以下示例终止会话。

```
BEGIN
    rdsadmin.rdsadmin_util.kill(
        sid    => sid,
        serial => serial_number,
        method => 'IMMEDIATE');
END;
/
```

以下示例终止与会话关联的进程。

```
BEGIN
    rdsadmin.rdsadmin_util.kill(
        sid    => sid,
        serial => serial_number,
```

```
        method => 'PROCESS');
END;
/
```

在会话中取消 SQL 语句

要在会话中取消 SQL 语句，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_util.cancel`。

> **Note**
>
> Oracle Database 19c (19.0.0) 和 RDS for Oracle 的所有更高主要和次要版本均支持此过程。

`cancel` 过程具有以下参数。

参数名称	数据类型	默认值	必需	描述
`sid`	number	—	是	会话标识符。
`serial`	number	—	是	会话的序列号。
`sql_id`	varchar2	null	否	SQL 语句的 SQL 标识符。

以下示例在会话中取消 SQL 语句。

```
begin
    rdsadmin.rdsadmin_util.cancel(
        sid    => sid,
        serial => serial_number,
        sql_id => sql_id);
end;
/
```

要获取会话标识符、会话序列号和 SQL 语句的 SQL 标识符，请查询 `V$SESSION` 视图。以下示例获取用户 `AWSUSER` 的所有会话和 SQL 标识符。

```
select SID, SERIAL#, SQL_ID, STATUS from V$SESSION where USERNAME = 'AWSUSER';
```

启用和禁用受限制的会话

要启用和禁用受限制的会话，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_util.restricted_session`。`restricted_session` 过程具有以下参数。

参数名称	数据类型	默认值	是	描述
`p_enable`	布尔值	true	否	设置为 `true` 则启用受限制的会话；设置为 `false` 则禁用受限制的会话。

以下示例介绍了如何启用和禁用受限制的会话。

```
/* Verify that the database is currently unrestricted. */

SELECT LOGINS FROM V$INSTANCE;

LOGINS
-------
ALLOWED

/* Enable restricted sessions */

EXEC rdsadmin.rdsadmin_util.restricted_session(p_enable => true);


/* Verify that the database is now restricted. */

SELECT LOGINS FROM V$INSTANCE;

LOGINS
----------
RESTRICTED


/* Disable restricted sessions */

EXEC rdsadmin.rdsadmin_util.restricted_session(p_enable => false);
```

```
/* Verify that the database is now unrestricted again. */

SELECT LOGINS FROM V$INSTANCE;

LOGINS
-------
ALLOWED
```

刷新共享池

要刷新共享池，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_util.flush_shared_pool`。`flush_shared_pool` 过程没有参数。

以下示例刷新共享池。

```
EXEC rdsadmin.rdsadmin_util.flush_shared_pool;
```

刷新缓冲区缓存

要刷新缓冲区缓存，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_util.flush_buffer_cache`。`flush_buffer_cache` 过程没有参数。

以下示例刷新缓冲区缓存。

```
EXEC rdsadmin.rdsadmin_util.flush_buffer_cache;
```

刷新数据库智能闪存缓存

要刷新数据库智能闪存缓存，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_util.flush_flash_cache`。`flush_flash_cache` 过程没有参数。以下示例刷新数据库智能闪存缓存。

```
EXEC rdsadmin.rdsadmin_util.flush_flash_cache;
```

有关将数据库智能闪存缓存与 RDS for Oracle 结合使用的更多信息，请参阅 在 RDS for Oracle 实例存储中存储临时数据。

向 SYS 对象授予 SELECT 或 EXECUTE 权限

通常，您使用可包含多个对象的角色来传输权限。要为单个对象授予权限，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_util.grant_sys_object`。此过程仅授予已通过角色或直接授权授予主用户的权限。

`grant_sys_object` 过程具有以下参数。

> **Important**
>
> 对于所有参数值，除非您使用区分大小写的标识符创建了用户，否则请使用大写字母。例如，如果您运行 `CREATE USER myuser` 或 `CREATE USER MYUSER`，则数据字典将存储 `MYUSER`。但是，如果您在 `CREATE USER "MyUser"` 中使用双引号，则数据字典将存储 `MyUser`。

参数名称	数据类型	默认值	必需	描述
`p_obj_name`	varchar2	—	是	要授予其权限的对象的名称。对象可以是目录、函数、包、过程、序列、表或视图。对象名称的拼写必须与其在 `DBA_OBJECTS` 中出现时的拼写完全相同。由于大多数系统对象都是用大写形式定义的，因此，建议您先尝试这样做。
`p_grantee`	varchar2	—	是	要向其授予权限的对象的名称。对象可以是架构或角色。
`p_privilege`	varchar2	null	是	—
`p_grant_option`	布尔值	false	否	设置为 `true`，则使用 with grant option。

以下示例将名为 V_$SESSION 的对象的 select 权限授予名为 USER1 的用户。

```
begin
    rdsadmin.rdsadmin_util.grant_sys_object(
        p_obj_name  => 'V_$SESSION',
        p_grantee   => 'USER1',
        p_privilege => 'SELECT');
end;
/
```

以下示例使用授予选项将名为 V_$SESSION 的对象的 select 权限授予名为 USER1 的用户。

```
begin
    rdsadmin.rdsadmin_util.grant_sys_object(
        p_obj_name     => 'V_$SESSION',
        p_grantee      => 'USER1',
        p_privilege    => 'SELECT',
        p_grant_option => true);
end;
/
```

要能够授予针对某个对象的权限，您的账户必须具有通过使用授权选项或通过使用 `with admin option` 授予的角色直接为其授予的这些权限。在最常见的情况下，您可能希望授予已授予给 SELECT 角色的针对 DBA 视图的 SELECT_CATALOG_ROLE 权限。如果尚未使用 `with admin option` 将该角色直接授予您的用户，则您将无法转让此权限。如果您具有 DBA 权限，则可直接将该角色授予其他用户。

以下示例将 SELECT_CATALOG_ROLE 和 EXECUTE_CATALOG_ROLE 授予 USER1。由于使用了 `with admin option`，因此，USER1 现在可向已获得 SELECT_CATALOG_ROLE 的 SYS 对象授予访问权。

```
GRANT SELECT_CATALOG_ROLE TO USER1 WITH ADMIN OPTION;
GRANT EXECUTE_CATALOG_ROLE to USER1 WITH ADMIN OPTION;
```

无需对已获得 PUBLIC 的对象重新授权。如果您使用 `grant_sys_object` 过程重新授予访问权，则此过程调用将成功。

撤销 SYS 对象的 SELECT 或 EXECUTE 权限

要撤销单个对象的特权，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_util.revoke_sys_object`。该过程仅撤消已通过角色或直接授权授予主账户的权限。

revoke_sys_object 过程具有以下参数。

参数名称	数据类型	默认值	必需	描述
p_obj_name	varchar2	—	是	要撤销其权限的对象的名称。对象可以是目录、函数、包、过程、序列、表或视图。对象名称的拼写必须与其在 DBA_OBJECTS 中出现时的拼写完全相同。由于大多数系统对象都是用大写形式定义的，因此，建议您先尝试这样做。
p_revokee	varchar2	—	是	要撤销其权限的对象的名称。对象可以是架构或角色。
p_privilege	varchar2	null	是	—

以下示例从名为 V_$SESSION 的用户撤销名为 USER1 的对象的 select 权限。

```
begin
    rdsadmin.rdsadmin_util.revoke_sys_object(
        p_obj_name  => 'V_$SESSION',
        p_revokee   => 'USER1',
        p_privilege => 'SELECT');
end;
/
```

管理 Oracle 数据库实例的 RDS_X$ 视图

您可能需要访问 SYS.X$ 固定表，这些表只能由 SYS 访问。要在符合条件的 X$ 表上创建 SYS.RDS_X$ 视图，请使用 rdsadmin.rdsadmin_util 包中的过程。您的主用户将自动获得对于 RDS_X$ 视图的权限 SELECT … WITH GRANT OPTION。

以下数据库引擎版本中提供了 rdsadmin.rdsadmin_util 过程：

- `21.0.0.0.ru-2023-10.rur-2023-10.r1` 以及更高的 Oracle Database 21c 版本
- `19.0.0.0.ru-2023-10.rur-2023-10.r1` 以及更高的 Oracle Database 19c 版本

> **Important**
>
> 在内部，`rdsadmin.rdsadmin_util` 包在 X$ 表上创建视图。X$ 表是内部系统对象，Oracle 数据库文档中未对此进行描述。我们建议您在非生产数据库中测试特定的视图，并且在 Oracle 支持的指导下仅在生产数据库中创建视图。

列出符合条件在 RDS_X$ 视图中使用的 X$ 固定表

要列出符合条件在 RDS_X$ 视图中使用的 X$ 表，请使用 RDS 过程 `rdsadmin.rdsadmin_util.list_allowed_sys_x$_views`。此过程不接受任何参数。以下语句列出了所有符合条件的 X$ 表（包括示例输出）。

```
SQL> SET SERVEROUTPUT ON
SQL> SELECT * FROM TABLE(rdsadmin.rdsadmin_util.list_allowed_sys_x$_views);

'X$BH'
'X$K2GTE'
'X$KCBWBPD'
'X$KCBWDS'
'X$KGLLK'
'X$KGLOB'
'X$KGLPN'
'X$KSLHOT'
'X$KSMSP'
'X$KSPPCV'
'X$KSPPI'
'X$KSPPSV'
'X$KSQEQ'
'X$KSQRS'
'X$KTUXE'
'X$KQRFP'
```

符合条件的 X$ 表的列表可能随时间推移而变化。要确保符合条件的 X$ 固定表的列表是最新的，请定期重新运行 `list_allowed_sys_x$_views`。

创建 SYS.RDS_X$ 视图

要在符合条件的 X$ 表上创建 RDS_X$ 视图，请使用 RDS 过程 `rdsadmin.rdsadmin_util.create_sys_x$_view`。您只能为 `rdsadmin.rdsadmin_util.list_allowed_sys_x$_views` 的输出中列出的表创建视图。`create_sys_x$_view` 过程接受以下参数。

参数名称	数据类型	默认值	必需	描述
`p_x$_tbl`	varchar2	Null	是	有效的 X$ 表名称。此值必须是 `list_allowed_sys_x$_views` 报告的 X$ 表之一。
`p_force_creation`	布尔值	`FALSE`	否	一个值，表示是否强制为 X$ 表创建已存在的 RDS_X$ 视图。默认情况下，如果视图已经存在，RDS 将不会创建该视图。要强制创建，请将此参数设置为 `TRUE`。

下面的示例基于表 `X$KGLOB` 创建 `SYS.RDS_X$KGLOB` 视图。视图名称的格式为 `RDS_X$tablename`。

```
SQL> SET SERVEROUTPUT ON
SQL> EXEC rdsadmin.rdsadmin_util.create_sys_x$_view('X$KGLOB');

PL/SQL procedure successfully completed.
```

以下数据字典查询列出视图 `SYS.RDS_X$KGLOB` 并显示其状态。您的主用户将自动获得对该视图的 `SELECT ... WITH GRANT OPTION` 权限。

```
SQL> SET SERVEROUTPUT ON
SQL> COL OWNER FORMAT A30
SQL> COL OBJECT_NAME FORMAT A30
SQL> COL STATUS FORMAT A30
SQL> SET LINESIZE 200
```

```
SQL> SELECT OWNER, OBJECT_NAME, STATUS
FROM DBA_OBJECTS
WHERE OWNER = 'SYS' AND OBJECT_NAME = 'RDS_X$KGLOB';

OWNER                          OBJECT_NAME                    STATUS
------------------------------ ------------------------------
 ------------------------------
SYS                            RDS_X$KGLOB                    VALID
```

> **Important**
>
> 不能保证 X$ 表在升级前后保持不变。在引擎升级期间，RDS for Oracle 会删除并重新创建 X$ 表上的 RDS_X$ 视图。然后它向主用户授予 `SELECT ... WITH GRANT OPTION` 权限。升级后，根据需要向数据库用户授予对相应 RDS_X$ 视图的权限。

列出 SYS.RDS_X$ 视图

要列出现有 RDS_X$ 视图，请使用 RDS 过程 `rdsadmin.rdsadmin_util.list_created_sys_x$_views`。该过程仅列出由过程 `create_sys_x$_view` 创建的视图。以下示例列出了具有相应 RDS_X$ 视图的 X$ 表（包括示例输出）。

```
SQL> SET SERVEROUTPUT ON
SQL> COL XD_TBL_NAME FORMAT A30
SQL> COL STATUS FORMAT A30
SQL> SET LINESIZE 200
SQL> SELECT * FROM TABLE(rdsadmin.rdsadmin_util.list_created_sys_x$_views);

XD_TBL_NAME                    STATUS
------------------------------ ------------------------------
X$BH                           VALID
X$K2GTE                        VALID
X$KCBWBPD                      VALID

3 rows selected.
```

删除 RDS_X$ 视图

要删除 SYS.RDS_X$ 视图，请使用 RDS 过程 `rdsadmin.rdsadmin_util.drop_sys_x$_view`。您只能删除 `rdsadmin.rdsadmin_util.list_allowed_sys_x$_views` 的输出中列出的视图。`drop_sys_x$_view` 过程接受以下参数。

参数名称	数据类型	默认值	必需	描述
p_x$_tbl	varchar2	Null	是	有效的 X$ 固定表名称。此值必须是 list_created_sys_x$_views 报告的 X$ 固定表之一。

以下示例删除了在表 X$KGLOB 上创建的 RDS_X$KGLOB 视图。

```
SQL> SET SERVEROUTPUT ON
SQL> EXEC rdsadmin.rdsadmin_util.drop_sys_x$_view('X$KGLOB');

PL/SQL procedure successfully completed.
```

以下示例显示视图 SYS.RDS_X$KGLOB 已被删除（包括示例输出）。

```
SQL> SET SERVEROUTPUT ON
SQL> COL OWNER FORMAT A30
SQL> COL OBJECT_NAME FORMAT A30
SQL> COL STATUS FORMAT A30
SQL> SET LINESIZE 200
SQL> SELECT OWNER, OBJECT_NAME, STATUS
FROM DBA_OBJECTS
WHERE OWNER = 'SYS' AND OBJECT_NAME = 'RDS_X$KGLOB';

no rows selected
```

向非主用户授予权限

您可以使用 SYS 角色向 SELECT_CATALOG_ROLE 架构中的多个对象授予 select 权限。SELECT_CATALOG_ROLE 角色向用户授予对数据字典视图的 SELECT 权限。以下示例向名为 SELECT_CATALOG_ROLE 的用户授予角色 user1。

```
GRANT SELECT_CATALOG_ROLE TO user1;
```

您可以使用 EXECUTE 角色向 SYS 架构中的多个对象授予 EXECUTE_CATALOG_ROLE 权限。EXECUTE_CATALOG_ROLE 角色向用户授予对数据字典中的包和过程的 EXECUTE 权限。以下示例向名为 user1 的用户授予角色 EXECUTE_CATALOG_ROLE。

```
GRANT EXECUTE_CATALOG_ROLE TO user1;
```

以下示例获取角色 SELECT_CATALOG_ROLE 和 EXECUTE_CATALOG_ROLE 允许的权限。

```
  SELECT *
    FROM ROLE_TAB_PRIVS
   WHERE ROLE IN ('SELECT_CATALOG_ROLE','EXECUTE_CATALOG_ROLE')
ORDER BY ROLE, TABLE_NAME ASC;
```

以下示例将创建一个名为 user1 的非主用户，授予 CREATE SESSION 权限并授予对名为 sh.sales 的数据库的 SELECT 权限。

```
CREATE USER user1 IDENTIFIED BY PASSWORD;
GRANT CREATE SESSION TO user1;
GRANT SELECT ON sh.sales TO user1;
```

创建自定义函数来验证密码

您可以通过以下方式创建自定义密码验证函数：

- 要使用标准验证逻辑，并将函数存储在 SYS 架构中，请使用 create_verify_function 过程。
- 要使用自定义验证逻辑，或者避免将函数存储在 SYS 架构中，请使用 create_passthrough_verify_fcn 过程。

create_verify_function 过程

您可以使用 Amazon RDS 过程 rdsadmin.rdsadmin_password_verify.create_verify_function 创建自定义函数来验证密码。所有版本的 RDS for Oracle 都支持 create_verify_function 过程。

create_verify_function 过程具有以下参数。

参数名称	数据类型	默认值	必需	描述
p_verify_function_name	varchar2	—	是	自定义函数的名称。将在 SYS 架构中为您创建此函数。为用户配置文件分配此函数。

参数名称	数据类型	默认值	必需	描述
p_min_length	number	8	否	要求的字符的最小数量。
p_max_length	number	256	否	允许的字符的最大数量。
p_min_letters	number	1	否	要求的字母的最小数量。
p_min_uppercase	number	0	否	要求的大写字母的最小数量。
p_min_lowercase	number	0	否	要求的小写字母的最小数量。
p_min_digits	number	1	否	要求的数字的最小数量。
p_min_special	number	0	否	要求的特殊字符的最小数量。
p_min_different_chars	number	3	否	旧密码和新密码之间需要的不同字符的最小数。
p_disallow_username	布尔值	true	否	设置为 true 可禁止在密码中使用用户名。
p_disallow_reverse	布尔值	true	否	设置为 true 则禁止在密码中使用反向用户名。
p_disallow_db_name	布尔值	true	否	设置为 true 则禁止在密码中使用数据库或服务器名称。
p_disallow_simple_strings	布尔值	true	否	设置为 true 则禁止将简单字符串用作密码。
p_disallow_whitespace	布尔值	false	否	设置为 true 则禁止在密码中使用空格字符。
p_disallow_at_sign	布尔值	false	否	设置为 true 以禁止在密码中使用 @ 字符。

可以创建多个密码验证函数。

自定义函数的名称受到一些限制。您的自定义函数不能具有与现有系统对象相同的名称。名称长度不能超过 30 个字符。另外，名称必须包括以下字符串之一：PASSWORD、VERIFY、COMPLEXITY、ENFORCE 或 STRENGTH。

下面的示例创建一个名为 CUSTOM_PASSWORD_FUNCTION 的函数。此函数要求密码具有至少 12 个字符、2 个大写字符、1 个数字和 1 个特殊字符，并且该密码禁止使用 @ 字符。

```
begin
    rdsadmin.rdsadmin_password_verify.create_verify_function(
        p_verify_function_name => 'CUSTOM_PASSWORD_FUNCTION',
        p_min_length           => 12,
        p_min_uppercase        => 2,
        p_min_digits           => 1,
        p_min_special          => 1,
        p_disallow_at_sign     => true);
end;
/
```

要查看验证函数的文本，请查询 DBA_SOURCE。以下示例获取名为 CUSTOM_PASSWORD_FUNCTION 的自定义密码函数的文本。

```
COL TEXT FORMAT a150

  SELECT TEXT
    FROM DBA_SOURCE
   WHERE OWNER = 'SYS'
     AND NAME = 'CUSTOM_PASSWORD_FUNCTION'
ORDER BY LINE;
```

要将您的验证函数与用户配置文件关联，请使用 `alter profile`。以下示例将验证函数与 DEFAULT 用户配置文件关联。

```
ALTER PROFILE DEFAULT LIMIT PASSWORD_VERIFY_FUNCTION CUSTOM_PASSWORD_FUNCTION;
```

要查看哪些用户配置文件与哪些验证函数关联，请查询 DBA_PROFILES。以下示例获取与名为 CUSTOM_PASSWORD_FUNCTION 的自定义验证函数关联的配置文件。

```
SELECT * FROM DBA_PROFILES WHERE RESOURCE_NAME = 'PASSWORD' AND LIMIT =
 'CUSTOM_PASSWORD_FUNCTION';
```

```
PROFILE                    RESOURCE_NAME                    RESOURCE  LIMIT
-------------------------  -------------------------------- --------
 ------------------------
DEFAULT                    PASSWORD_VERIFY_FUNCTION         PASSWORD
 CUSTOM_PASSWORD_FUNCTION
```

以下示例获取所有配置文件以及与之关联的密码验证函数。

```
SELECT * FROM DBA_PROFILES WHERE RESOURCE_NAME = 'PASSWORD_VERIFY_FUNCTION';

PROFILE                    RESOURCE_NAME                    RESOURCE  LIMIT
-------------------------  -------------------------------- --------
 ------------------------
DEFAULT                    PASSWORD_VERIFY_FUNCTION         PASSWORD
 CUSTOM_PASSWORD_FUNCTION
RDSADMIN                   PASSWORD_VERIFY_FUNCTION         PASSWORD  NULL
```

create_passthrough_verify_fcn 过程

所有版本的 RDS for Oracle 都支持 `create_passthrough_verify_fcn` 过程。

您可以使用 Amazon RDS 过程 `rdsadmin.rdsadmin_password_verify.create_passthrough_verify_fcn` 创建自定义函数来验证密码。`create_passthrough_verify_fcn` 过程具有以下参数。

参数名称	数据类型	默认值	必需	描述
`p_verify_function_name`	varchar2	—	是	自定义验证函数的名称。这是在 SYS 架构中为您创建的包装函数，它不包含任何验证逻辑。为用户配置文件分配此函数。
`p_target_owner`	varchar2	—	是	自定义验证函数的架构所有者。
`p_target_function_name`	varchar2	—	是	包含验证逻辑的现有自定义函数的名称。您的自定

参数名称	数据类型	默认值	必需	描述
				义函数必须返回布尔值。如果密码有效，您的函数应返回 `true`；如果密码无效应返回 `false`。

以下示例创建的密码验证函数使用名为 `PASSWORD_LOGIC_EXTRA_STRONG` 的函数的逻辑。

```
begin
    rdsadmin.rdsadmin_password_verify.create_passthrough_verify_fcn(
        p_verify_function_name => 'CUSTOM_PASSWORD_FUNCTION',
        p_target_owner         => 'TEST_USER',
        p_target_function_name => 'PASSWORD_LOGIC_EXTRA_STRONG');
end;
/
```

要将验证函数与用户配置文件关联，请使用 `alter profile`。以下示例将验证函数与 `DEFAULT` 用户配置文件关联。

```
ALTER PROFILE DEFAULT LIMIT PASSWORD_VERIFY_FUNCTION CUSTOM_PASSWORD_FUNCTION;
```

设置自定义 DNS 服务器

Amazon RDS 支持对运行 Oracle 的数据库实例进行出站网络访问。有关出站网络访问（包括先决条件）的更多信息，请参阅使用证书和 Oracle Wallet 配置 UTL_HTTP 访问。

Amazon RDS Oracle 支持通过客户拥有的自定义域名服务 (DNS) 服务器解析 DNS。通过自定义 DNS 服务器，您可以仅解析 Amazon RDS 数据库实例中的完全限定域名。

自定义 DNS 名称服务器设置完成后，它需要 30 分钟将更改传播到数据库实例中。将更改传播到数据库实例后，需要进行 DNS 查找的所有出站网络流量将通过端口 53 查询您的 DNS 服务器。

要为您的 Amazon RDS for Oracle 数据库实例设置自定义 DNS 服务器，请执行以下操作：

- 在附加到 Virtual Private Cloud (VPC) 的 DHCP 选项集中，将 `domain-name-servers` 选项设置为您的 DNS 名称服务器的 IP 地址。有关更多信息，请参阅 DHCP 选项集。

> **Note**
>
> `domain-name-servers` 选项最多可接受四个值，但您的 Amazon RDS 数据库实例只使用第一个值。

- 请确保您的 DNS 服务器可以解析所有查找查询，包括公有 DNS 名称、Amazon EC2 私有 DNS 名称和特定客户的 DNS 名称。如果出站网络流量中包含您的 DNS 服务器不能处理的 DNS 查找，您的 DNS 服务器必须已配置适当的上游 DNS 提供程序。
- 请配置您的 DNS 服务器，以生成等于或小于 512 字节的用户数据报协议 (UDP) 响应。
- 请配置您的 DNS 服务器，以生成等于或小于 1024 字节的传输控制协议 (TCP) 响应。
- 配置您的 DNS 服务器，允许流量通过 Amazon RDS 数据库实例的端口 53 入站。如果您的 DNS 服务器在 Amazon VPC 中，此 VPC 的安全组必须包含以下入站规则：允许端口 53 上的 UDP 和 TCP 流量。如果您的 DNS 服务器不在 Amazon VPC 中，它必须具有适当的防火墙允许名单，以允许端口 53 上的 UDP 和 TCP 入站流量。

  有关更多信息，请参阅您的 VPC 的安全组和添加和删除规则。
- 配置您的 Amazon RDS 数据库实例的 VPC，允许流量通过端口 53 出站。您的 VPC 安全组必须包含以下出站规则：允许端口 53 上的 UDP 和 TCP 流量。

  有关更多信息，请参阅您的 VPC 的安全组和添加和删除规则。
- 必须正确配置 Amazon RDS 数据库实例和 DNS 服务器之间的路由路径，允许 DNS 流量。
  - 如果 Amazon RDS 数据库实例和 DNS 服务器不在同一 VPC 中，必须在它们之间设置对等连接。有关更多信息，请参阅什么是 VPC 对等连接？

设置和取消设置系统诊断事件

要在会话级别设置和取消设置诊断事件，可以使用 Oracle SQL 语句 `ALTER SESSION SET EVENTS`。但是，要在系统级别设置事件，您不能使用 Oracle SQL。相反，请使用 `rdsadmin.rdsadmin_util` 软件包中的系统事件过程。以下引擎版本中提供系统事件过程：

- 所有 Oracle Database 21c 版本
- 19.0.0.0.ru-2020-10.rur-2020-10.r1 或更高的 Oracle Database 19c 版本

  有关更多信息，请参阅《Amazon RDS for Oracle 发布说明》中的版本 19.0.0.0.ru-2020-10.rur-2020-10.r1

Important

rdsadmin.rdsadmin_util 软件包内部使用 ALTER SYSTEM SET EVENTS 语句设置事件。Oracle 数据库文档中没有记录此 ALTER SYSTEM 语句。某些系统诊断事件可能会生成大量跟踪信息、引起争用或影响数据库可用性。我们建议您在非生产数据库中测试特定的诊断事件，并且在 Oracle 支持的指导下仅在生产数据库中设置事件。

列出允许的系统诊断事件

要列出可以设置的系统事件，请使用 Amazon RDS 过程 rdsadmin.rdsadmin_util.list_allowed_system_events。此过程不接受任何参数。

以下示例列出了您可以设置的所有系统事件。

```
SET SERVEROUTPUT ON
EXEC rdsadmin.rdsadmin_util.list_allowed_system_events;
```

以下示例输出列出了事件编号及其描述。使用 Amazon RDS 过程 set_system_event 来设置这些事件并使用 unset_system_event 来取消设置它们。

```
604   - error occurred at recursive SQL level
942   - table or view does not exist
1401  - inserted value too large for column
1403  - no data found
1410  - invalid ROWID
1422  - exact fetch returns more than requested number of rows
1426  - numeric overflow
1427  - single-row subquery returns more than one row
1476  - divisor is equal to zero
1483  - invalid length for DATE or NUMBER bind variable
1489  - result of string concatenation is too long
1652  - unable to extend temp segment by  in tablespace
1858  - a non-numeric character was found where a numeric was expected
4031  - unable to allocate  bytes of shared memory ("","","","")
6502  - PL/SQL: numeric or value error
10027 - Specify Deadlock Trace Information to be Dumped
10046 - enable SQL statement timing
10053 - CBO Enable optimizer trace
10173 - Dynamic Sampling time-out error
10442 - enable trace of kst for ORA-01555 diagnostics
```

```
12008 - error in materialized view refresh path
12012 - error on auto execute of job
12504 - TNS:listener was not given the SERVICE_NAME in CONNECT_DATA
14400 - inserted partition key does not map to any partition
31693 - Table data object  failed to load/unload and is being skipped due to error:
```

> **Note**
> 允许的系统事件列表可能会随着时间的推移而变化。要确保您拥有最新的符合条件的活动列表，请使用 `rdsadmin.rdsadmin_util.list_allowed_system_events`。

设置系统诊断事件

要设置系统事件，请使用Amazon RDS过程`rdsadmin.rdsadmin_util.set_system_event`。您只能设置 `rdsadmin.rdsadmin_util.list_allowed_system_events` 输出中列出的事件。`set_system_event` 过程接受以下参数。

参数名称	数据类型	默认值	必需	描述
`p_event`	number	—	是	系统事件编号。此值必须是 `list_allowed_system_events` 报告的事件编号之一。
`p_level`	number	—	是	活动级别。有关不同级别值的说明，请参阅 Oracle 数据库文档或 Oracle 支持。

过程 `set_system_event` 根据以下原则构造和运行所需的 `ALTER SYSTEM SET EVENTS` 语句：

- 事件类型（`context` 或 `errorstack`）是自动确定的。
- `ALTER SYSTEM SET EVENTS '`*`event`*` LEVEL `*`event_level`*`'` 表单中的语句设置上下文事件。此符号等同于 `ALTER SYSTEM SET EVENTS '`*`event`*` TRACE NAME CONTEXT FOREVER, LEVEL `*`event_level`*`'`。

- ALTER SYSTEM SET EVENTS '*event* ERRORSTACK (*event_level*)' 表单中的语句设置错误堆栈事件。此符号等同于 ALTER SYSTEM SET EVENTS '*event* TRACE NAME ERRORSTACK LEVEL *event_level*'。

以下示例将事件 942 设置为级别 3，将事件 10442 设置为级别 10。示例输出包括在内。

```
SQL> SET SERVEROUTPUT ON
SQL> EXEC rdsadmin.rdsadmin_util.set_system_event(942,3);
Setting system event 942 with: alter system set events '942 errorstack (3)'

PL/SQL procedure successfully completed.

SQL> EXEC rdsadmin.rdsadmin_util.set_system_event(10442,10);
Setting system event 10442 with: alter system set events '10442 level 10'

PL/SQL procedure successfully completed.
```

列出设置的系统诊断事件

要列出当前设置的系统事件，请使用 Amazon RDS 过程 rdsadmin.rdsadmin_util.list_set_system_events。此过程仅报告 set_system_event 在系统级别设置的事件。

以下示例列出了活动的系统事件。

```
SET SERVEROUTPUT ON
EXEC rdsadmin.rdsadmin_util.list_set_system_events;
```

以下示例输出显示了事件列表、事件类型、当前设置事件的级别以及设置事件的时间。

```
942 errorstack (3) - set at 2020-11-03 11:42:27
10442 level 10 - set at 2020-11-03 11:42:41

PL/SQL procedure successfully completed.
```

取消设置系统诊断事件

要取消设置系统事件，请使用 Amazon RDS 过程 rdsadmin.rdsadmin_util.unset_system_event。您只能取消设置

rdsadmin.rdsadmin_util.list_allowed_system_events 输出中列出的事件。unset_system_event 过程接受以下参数。

参数名称	数据类型	默认值	必需	描述
p_event	number	—	是	系统事件编号。此值必须是 list_allowed_system_events 报告的事件编号之一。

以下示例取消设置事件 942 和 10442。示例输出包括在内。

```
SQL> SET SERVEROUTPUT ON
SQL> EXEC rdsadmin.rdsadmin_util.unset_system_event(942);
Unsetting system event 942 with: alter system set events '942 off'

PL/SQL procedure successfully completed.

SQL> EXEC rdsadmin.rdsadmin_util.unset_system_event(10442);
Unsetting system event 10442 with: alter system set events '10442 off'

PL/SQL procedure successfully completed.
```

执行 Oracle 数据库实例的常见数据库任务

下文中介绍了如何在运行 Oracle 的 Amazon RDS 数据库实例上，执行与数据库相关的特定常见 DBA 任务。为了提供托管服务体验，Amazon RDS 未提供数据库实例的 Shell 访问权限。并且 Amazon RDS 还限制了对需要高级权限的某些系统程序和表的访问。

主题

- 使用 Automatic Workload Repository (AWR) 生成性能报告
- 在 VPC 中调整用于数据库实例的数据库链接
- 设置数据库实例的默认版本
- 为 SYS.AUD$ 表启用审核
- 为 SYS.AUD$ 表禁用审核
- 清理中断的在线索引构建
- 跳过损坏的数据块
- 调整表空间、数据文件和临时文件的大小
- 清除回收站
- 为全部编写设置原定设置显示值

更改数据库的全局名称

要更改数据库的全局名称，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_util.rename_global_name`。`rename_global_name` 过程具有以下参数。

参数名称	数据类型	默认值	必需	描述
`p_new_global_name`	varchar2	—	是	数据库的新全局名称。

数据库必须处于打开状态，才能更改名称。有关更改数据库的全局名称的更多信息，请参阅 Oracle 文档中的 ALTER DATABASE。

以下示例将数据库的全局名称更改为 `new_global_name`。

```
EXEC rdsadmin.rdsadmin_util.rename_global_name(p_new_global_name => 'new_global_name');
```

使用表空间

可以将表空间与 RDS for Oracle 结合使用，后者是一种逻辑存储单元，用于存储数据库的数据。

主题

- 创建表空间并配置其大小
- 设置默认表空间

- 设置默认临时表空间
- 在实例存储上创建临时表空间

创建表空间并配置其大小

Amazon RDS 仅支持数据文件、日志文件和控制文件的 Oracle Managed Files (OMF)。创建数据文件和日志文件时，无法指定物理文件名。

原定设置情况下，如果未指定数据文件大小，则创建的表空间原定设置为 AUTOEXTEND ON，且没有最大大小。在以下示例中，表空间 *users1* 是可自动扩展的。

```
CREATE TABLESPACE users1;
```

由于这些默认设置，表空间会增大以占用所有分配的存储空间。建议您在永久表空间和临时表空间上指定适当的最大大小，并仔细监视空间使用率。

以下示例创建一个名为 *users2* 的表空间，起始大小为 1GB。由于指定了数据文件大小，但未指定 AUTOEXTEND ON，因此表空间不可自动扩展。

```
CREATE TABLESPACE users2 DATAFILE SIZE 1G;
```

以下示例创建一个名为 *users3* 的表空间，起始大小为 1GB，开启了自动扩展，且最大大小为 10GB。

```
CREATE TABLESPACE users3 DATAFILE SIZE 1G AUTOEXTEND ON MAXSIZE 10G;
```

以下示例创建一个名为 *temp01* 的临时表空间。

```
CREATE TEMPORARY TABLESPACE temp01;
```

您可以使用 ALTER TABLESPACE 调整大文件表空间的大小。可以 KB (K)、MB (M)、GB (G) 或 TB (T) 为单位指定大小。以下示例将名为 *users_bf* 的大文件表空间的大小调整到 200MB。

```
ALTER TABLESPACE users_bf RESIZE 200M;
```

以下示例向名为 *users_sf* 的小文件表空间添加另一个数据文件。

```
ALTER TABLESPACE users_sf ADD DATAFILE SIZE 100000M AUTOEXTEND ON NEXT 250m
 MAXSIZE UNLIMITED;
```

设置默认表空间

要设置默认表空间，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_util.alter_default_tablespace`。`alter_default_tablespace` 过程具有以下参数。

参数名称	数据类型	默认值	必需	描述
`tablespace_name`	varchar	—	是	默认表空间的名称。

以下示例将默认表空间设置为 *users2*：

```
EXEC rdsadmin.rdsadmin_util.alter_default_tablespace(tablespace_name => 'users2');
```

设置默认临时表空间

要设置默认临时表空间，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_util.alter_default_temp_tablespace`。`alter_default_temp_tablespa`
过程具有以下参数。

参数名称	数据类型	默认值	必需	描述
`tablespace_name`	varchar	—	是	默认临时表空间的名称。

以下示例将默认临时表空间设置为 *temp01*。

```
EXEC rdsadmin.rdsadmin_util.alter_default_temp_tablespace(tablespace_name => 'temp01');
```

在实例存储上创建临时表空间

要在实例存储上创建临时表空间，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_util.create_inst_store_tmp_tblspace`。`create_inst_store_tmp_tbls`
过程具有以下参数。

参数名称	数据类型	默认值	必需	描述
`p_tablespace_name`	varchar	—	是	临时表空间的名称。

以下示例在实例存储中创建临时表空间 *temp01*。

```
EXEC rdsadmin.rdsadmin_util.create_inst_store_tmp_tblspace(p_tablespace_name =>
 'temp01');
```

> **Important**
>
> 运行 `rdsadmin_util.create_inst_store_tmp_tblspace` 时，新创建的临时表空间不会自动设置为原定设置临时表空间。要将其设置为原定设置，请参阅 设置默认临时表空间。

有关更多信息，请参阅 在 RDS for Oracle 实例存储中存储临时数据。

使用临时文件

在只读副本上向实例存储中添加临时文件

当您在主数据库实例上创建临时表空间时，只读副本不会创建临时文件。假设由于以下原因之一，只读副本上存在一个空的临时表空间：

- 您从只读副本上的表空间中删除了一个临时文件。有关更多信息，请参阅 删除只读副本上的临时文件。
- 您在主数据库实例上创建了一个新的临时表空间。在这种情况下，RDS for Oracle 会将元数据同步到只读副本。

您可以将临时文件添加到空的临时表空间，并将临时文件存储在实例存储中。要在实例存储中创建临时文件，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_util.add_inst_store_tempfile`。您只能在只读副本上使用此过程。 过程具有以下参数。

参数名称	数据类型	默认值	必需	描述
`p_tablespace_name`	varchar	—	是	只读副本上临时表空间的名称。

在以下示例中，您的只读副本上存在空的临时表空间 *temp01*。运行以下命令以创建此表空间的临时文件，并将其存储在实例存储中。

```
EXEC rdsadmin.rdsadmin_util.add_inst_store_tempfile(p_tablespace_name => 'temp01');
```

有关更多信息，请参阅 在 RDS for Oracle 实例存储中存储临时数据。

删除只读副本上的临时文件

您不能删除只读副本上的现有临时表空间。您可以将只读副本上的临时文件存储从 Amazon EBS 更改为实例存储，或者从实例存储更改为 Amazon EBS。为实现这些目标，请执行以下操作：

1. 删除只读副本上临时表空间中的当前临时文件。
2. 在不同的存储上创建新的临时文件。

要删除临时文件，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_util.drop_replica_tempfiles`。您只能在只读副本上使用此过程。`drop_replica_tempfiles` 过程具有以下参数。

参数名称	数据类型	默认值	必需	描述
`p_tablespace_name`	varchar	—	是	只读副本上临时表空间的名称。

假设一个名为 *temp01* 的临时表空间位于只读副本上的实例存储中。通过运行以下命令，删除此表空间中的所有临时文件。

```
EXEC rdsadmin.rdsadmin_util.drop_replica_tempfiles(p_tablespace_name => 'temp01');
```

有关更多信息，请参阅 在 RDS for Oracle 实例存储中存储临时数据。

通过检查点检验数据库

要通过检查点检验数据库，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_util.checkpoint`。`checkpoint` 过程没有参数。

以下示例为数据库创建检查点。

```
EXEC rdsadmin.rdsadmin_util.checkpoint;
```

设置分布式恢复

要设置分布式恢复，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_util.enable_distr_recovery` 和 `disable_distr_recovery`。这些过程没有参数。

以下示例启用分布式恢复。

```
EXEC rdsadmin.rdsadmin_util.enable_distr_recovery;
```

以下示例禁用分布式恢复。

```
EXEC rdsadmin.rdsadmin_util.disable_distr_recovery;
```

设置数据库时区

您可以通过以下方式设置 Amazon RDS Oracle 数据库的时区：

- `Timezone` 选项

  `Timezone` 选项在主机级别更改时区，并且会影响所有日期列和值（例如 `SYSDATE`）。有关更多信息，请参阅"Oracle 时区"。
- Amazon RDS 过程 `rdsadmin.rdsadmin_util.alter_db_time_zone`

  `alter_db_time_zone` 步骤仅更改某些数据类型的时区，而不更改 `SYSDATE`。在 Oracle 文档中列出了与设置时区有关的其他限制。

> **Note**
>
> 您还可以为 Oracle Scheduler 设置默认时区。有关更多信息，请参阅"设置 Oracle Scheduler 作业的时区"。

`alter_db_time_zone` 过程具有以下参数。

参数名称	数据类型	默认值	必需	描述
`p_new_tz`	varchar2	—	是	作为已命名的区域或与协调世界时 (UTC) 的绝对偏

参数名称	数据类型	默认值	必需	描述
				移量的时区。有效偏移范围介于 -12:00 到 +14:00 之间。

以下示例将时区更改为 UTC 加 3 小时。

```
EXEC rdsadmin.rdsadmin_util.alter_db_time_zone(p_new_tz => '+3:00');
```

以下示例将时区更改为非洲/阿尔及尔时区。

```
EXEC rdsadmin.rdsadmin_util.alter_db_time_zone(p_new_tz => 'Africa/Algiers');
```

使用 `alter_db_time_zone` 过程更改时区后，重启数据库实例以使更改生效。有关更多信息，请参阅“重启中的数据库实例”。有关升级时区的信息，请参阅 时区注意事项。

使用 Oracle 外部表

Oracle 外部表是具有不在数据库中数据的表。这些数据位于数据库可以访问的外部文件中。使用外部表，您可以访问数据而无需将其加载到数据库中。有关外部表的更多信息，请参阅 Oracle 文档中的管理外部表。

使用 Amazon RDS，您可以将外部表文件存储在目录对象中。您可以创建目录对象，或者使用 Oracle 数据库中预定义的目录对象，例如 DATA_PUMP_DIR 目录。有关创建目录对象的信息，请参阅在主数据存储空间中创建和删除目录。您可以查询 ALL_DIRECTORIES 视图来列出您的 Amazon RDS Oracle 数据库实例的目录对象。

> **Note**
> 目录对象指向由实例使用的主数据存储空间 (Amazon EBS 卷)。—包括数据文件、重做日志、审核、跟踪和其他文件在内—所用的空间均计入分配的存储中。

您可以使用 DBMS_FILE_TRANSFER 包或 UTL_FILE 包，将外部数据文件从一个 Oracle 数据库移动到另一个。外部数据文件从源数据库上的目录移至目标数据库上的指定目录中。有关使用 DBMS_FILE_TRANSFER 的信息，请参阅 使用 Oracle Data Pump 导入。

移动外部数据文件之后，您可以用它来创建外部表。以下示例创建使用 USER_DIR1 目录中 emp_xt_file1.txt 文件的外部表。

```
CREATE TABLE emp_xt (
  emp_id      NUMBER,
  first_name  VARCHAR2(50),
  last_name   VARCHAR2(50),
  user_name   VARCHAR2(20)
)
ORGANIZATION EXTERNAL (
  TYPE ORACLE_LOADER
  DEFAULT DIRECTORY USER_DIR1
  ACCESS PARAMETERS (
    RECORDS DELIMITED BY NEWLINE
    FIELDS TERMINATED BY ','
    MISSING FIELD VALUES ARE NULL
    (emp_id,first_name,last_name,user_name)
  )
  LOCATION ('emp_xt_file1.txt')
)
PARALLEL
REJECT LIMIT UNLIMITED;
```

假设您要将位于 Amazon RDS Oracle 数据库实例中的数据移动到外部数据文件中。在这种情况下，您可以通过创建外部表并从数据库中的表选择数据来填充外部数据文件。例如，以下 SQL 语句通过查询数据库中的 orders_xt 表创建 orders 外部表。

```
CREATE TABLE orders_xt
  ORGANIZATION EXTERNAL
   (
     TYPE ORACLE_DATAPUMP
     DEFAULT DIRECTORY DATA_PUMP_DIR
     LOCATION ('orders_xt.dmp')
   )
   AS SELECT * FROM orders;
```

在此示例中，数据填充到 DATA_PUMP_DIR 目录的 orders_xt.dmp 文件中。

使用 Automatic Workload Repository (AWR) 生成性能报告

要收集性能数据并生成报告，Oracle 建议使用 Automatic Workload Repository (AWR)。AWR 需要 Oracle Database Enterprise Edition 以及 Diagnostics and Tuning 包的许可证。要启用 AWR，

请将 CONTROL_MANAGEMENT_PACK_ACCESS 初始化参数设置为 DIAGNOSTIC 或 DIAGNOSTIC +TUNING。

在 RDS 中使用 AWR 报告

要生成 AWR 报告，您可以运行脚本，例如 awrrpt.sql。这些脚本安装在数据库主机服务器上。在 Amazon RDS 中，您无法直接访问主机。但是，您可以从其他 Oracle 数据库安装中获取 SQL 脚本的副本。

您还可以通过在 SYS.DBMS_WORKLOAD_REPOSITORY PL/SQL 软件包中运行过程来使用 AWR。您可以使用此软件包管理基准和快照，还可以显示 ASH 和 AWR 报告。例如，要生成文本格式的 AWR 报告，请运行 DBMS_WORKLOAD_REPOSITORY.AWR_REPORT_TEXT 过程。但是，您无法从Amazon Web Services Management Console访问这些 AWR 报告。

使用 AWR 时，我们建议使用 rdsadmin.rdsadmin_diagnostic_util 过程。您可以使用这些过程生成以下内容：

- AWR 报告
- Active Session History (ASH) 报告
- Automatic Database Diagnostic Monitor (ADDM) 报告
- AWR 数据的 Oracle Data Pump Export 转储文件

rdsadmin_diagnostic_util 过程将报告保存到数据库实例文件系统。您可以从控制台访问这些报告。您还可以使用 rdsadmin.rds_file_util 过程访问报告，并可以使用 S3 集成访问复制到 Amazon S3 的报告。有关更多信息，请参阅“读取数据库实例目录中的文件”和“Amazon S3 集成”。

您可在以下 Amazon RDS for Oracle 数据库引擎版本中使用 rdsadmin_diagnostic_util 过程：

- 所有 Oracle Database 21c 版本
- 19.0.0.0.ru-2020-04.rur-2020-04.r1 或更高的 Oracle Database 19c 版本

有关解释如何在复制场景中使用诊断报告的博客，请参阅为 Amazon RDS for Oracle 只读副本生成 AWR 报告。

诊断实用程序软件包的通用参数

通常，您在通过 rdsadmin_diagnostic_util 软件包管理 AWR 和 ADDM 时可以使用以下参数。

参数	数据类型	默认值	必需	描述
begin_snap_id	NUMBEF	—	是	开始快照的 ID。
end_snap_id	NUMBEF	—	是	结束快照的 ID。
dump_directory	VARCH/	BDUMF	否	将报告写入或将文件导出到的目录。如果您指定非默认目录，则运行 rdsadmin_diagnostic_util 过程的用户必须具有该目录的写入权限。
p_tag	VARCH/	—	否	可用于区分备份的字符串，以便指示备份的目的或用法，例如 incremental 或 daily。 您最多可指定 30 个字符。有效字符为 a-z、A-Z、0-9、下划线 (_)、短划线 (-) 以及点号 (.)。此标签不区分大小写。无论输入标签时使用的大小写如何，RMAN 始终以大写形式存储标签。 标签不一定是唯一的，因此多个备份可以使用相同的标签。如果未指定标签，则 RMAN 会使用 TAG*YYYYMMDDTHHMMSS* 格式自动分配默认标签，其中 *YYYY* 表示年份、*MM* 表示月份、*DD* 表示某天，*HH* 表示小时（采用 24 小时格式）、*MM* 表示分钟、*SS* 表示秒。日期和时间表示 RMAN 开始备份的时间。例如，备份的默认标签 TAG20190927T214517 表示备份开始于 2019 年 09 月 27 日 21:45:17。 以下 RDS for Oracle 数据库引擎版本支持 p_tag 参数： • Oracle Database 21c (21.0.0) • Oracle Database 19c (19.0.0)，使用 19.0.0.0.ru-2021-10.rur-2021-10.r1 及更高版本
report_type	VARCH/	HTML	否	报告的格式。有效值为 TEXT 和 HTML。

参数	数据类型	默认值	必需	描述
dbid	NUMBER	—	否	Oracle 的有效数据库标识符 (DBID)，显示在 DBA_HIST_DATABASE_INSTANCE 中。如果未指定此参数，RDS 将使用 V$DATABASE.DBID 视图中显示的当前 DBID。

使用 rdsadmin_diagnostic_util 软件包管理 ASH 时，通常使用以下参数。

参数	数据类型	默认值	必需	描述
begin_time	DATE	—	是	ASH 分析的开始时间。
end_time	DATE	—	是	ASH 分析的结束时间。
slot_width	NUMBER	0	否	ASH 报告的“Top Activity (热门活动)”部分中使用的槽的持续时间（以秒为单位）。如果未指定此参数，则 begin_time 和 end_time 之间的时间间隔不超过 10 个槽。
sid	NUMBER	Null	否	会话 ID。
sql_id	VARCHAR2	Null	否	SQL ID。
wait_class	VARCHAR2	Null	否	等待类名称。
service_hash	NUMBER	Null	否	服务名称哈希。
module_name	VARCHAR2	Null	否	模块名称。
action_name	VARCHAR2	Null	否	操作名称。

参数	数据类型	默认值	必需	描述
client_id	VARCHAR2	Null	否	数据库会话的特定于应用程序的 ID。
plsql_entry	VARCHAR2	Null	否	PL/SQL 入口点。

生成 AWR 报告

要生成 AWR 报告，请使用 rdsadmin.rdsadmin_diagnostic_util.awr_report 过程。

以下示例生成快照范围 101–106 的 AWR 报告。输出文本文件名为 awrrpt_101_106.txt。您可以通过Amazon Web Services Management Console访问此报告。

```
EXEC rdsadmin.rdsadmin_diagnostic_util.awr_report(101,106,'TEXT');
```

以下示例生成快照范围 63–65 的 HTML 报告。输出 HTML 文件名为 awrrpt_63_65.html。该过程将报告写入名为 AWR_RPT_DUMP 的非默认数据库目录。

```
EXEC rdsadmin.rdsadmin_diagnostic_util.awr_report(63,65,'HTML','AWR_RPT_DUMP');
```

将 AWR 数据提取到转储文件

要将 AWR 数据提取到转储文件中，请使用 rdsadmin.rdsadmin_diagnostic_util.awr_extract 过程。

以下示例提取范围 101–106 中的快照。输出转储文件名为 awrextract_101_106.dmp。您可以通过控制台访问此文件。

```
EXEC rdsadmin.rdsadmin_diagnostic_util.awr_extract(101,106);
```

以下示例提取范围在 63–65 中的快照。输出转储文件名为 awrextract_63_65.dmp。该文件存储在名为 AWR_RPT_DUMP 的非默认数据库目录中。

```
EXEC rdsadmin.rdsadmin_diagnostic_util.awr_extract(63,65,'AWR_RPT_DUMP');
```

生成 ADDM 报告

要生成 ADDM 报告，请使用 rdsadmin.rdsadmin_diagnostic_util.addm_report 过程。

以下示例生成快照范围 101–106 的 ADDM 报告。输出文本文件名为 addmrpt_101_106.txt。您可以通过控制台访问报告。

```
EXEC rdsadmin.rdsadmin_diagnostic_util.addm_report(101,106);
```

以下示例生成快照范围 63–65 的 ADDM 报告。输出文本文件名为 addmrpt_63_65.txt。该文件存储在名为 ADDM_RPT_DUMP 的非默认数据库目录中。

```
EXEC rdsadmin.rdsadmin_diagnostic_util.addm_report(63,65,'ADDM_RPT_DUMP');
```

生成 ASH 报告

要生成 ASH 报告，请使用 rdsadmin.rdsadmin_diagnostic_util.ash_report 过程。

以下示例生成一个 ASH 报告，其中包含当前时间 14 分钟前的数据。输出文件的名称使用格式 ashrpt*begin_timeend_time*.txt，其中 *begin_time* 和 *end_time* 使用格式 YYYYMMDDHH24MISS。您可以通过控制台访问文件。

```
BEGIN
    rdsadmin.rdsadmin_diagnostic_util.ash_report(
        begin_time     =>     SYSDATE-14/1440,
        end_time       =>     SYSDATE,
        report_type    =>     'TEXT');
END;
/
```

以下示例生成一个 ASH 报告，其中包括从 2019 年 11 月 18 日下午 6 点 07 分到 2019 年 11 月 18 日下午 6 点 15 分的数据。输出 HTML 报告名为 ashrpt_20190918180700_20190918181500.html。该报告存储在名为 AWR_RPT_DUMP 的非默认数据库目录中。

```
BEGIN
    rdsadmin.rdsadmin_diagnostic_util.ash_report(
        begin_time     =>    TO_DATE('2019-09-18 18:07:00', 'YYYY-MM-DD HH24:MI:SS'),
        end_time       =>    TO_DATE('2019-09-18 18:15:00', 'YYYY-MM-DD HH24:MI:SS'),
```

```
        report_type     =>    'html',
        dump_directory =>    'AWR_RPT_DUMP');
END;
/
```

从控制台或 CLI 访问 AWR 报告

要访问 AWR 报告或导出转储文件，您可以使用 Amazon Web Services Management Console 或 Amazon CLI。有关更多信息，请参阅 下载数据库日志文件。

在 VPC 中调整用于数据库实例的数据库链接

要在同一 Virtual Private Cloud (VPC) 或对等 VPC 中使用与 Amazon RDS 数据库实例的 Oracle 数据库链接，这两个数据库实例之间应具有有效路由。使用 VPC 路由表和网络访问控制列表 (ACL) 验证数据库实例之间的有效路由。

每个数据库实例的安全组必须允许另一个数据库实例的传入和传出。入站和出站规则可引用同一 VPC 或对等 VPC 中的安全组。有关更多信息，请参阅更新安全组以引用对等的 VPC 安全组。

如果您已使用 VPC 中的 DHCP 选项集配置自定义 DNS 服务器，则自定义 DNS 服务器必须能够解析数据库链接目标的名称。有关更多信息，请参阅"设置自定义 DNS 服务器"。

有关将数据库链接用于 Oracle Data Pump 的更多信息，请参阅 使用 Oracle Data Pump 导入。

设置数据库实例的默认版本

您可以在私有环境中重新定义称为版本的数据库对象。您可以使用基于版本的重新定义，以最少的停机来升级应用程序的数据库对象。

您可以使用 Amazon RDS 过程 `rdsadmin.rdsadmin_util.alter_default_edition` 设置 Amazon RDS Oracle 数据库实例的默认版本。

以下示例将 Amazon RDS Oracle 数据库实例的默认版本设置为 `RELEASE_V1`。

```
EXEC rdsadmin.rdsadmin_util.alter_default_edition('RELEASE_V1');
```

以下示例将 Amazon RDS Oracle 数据库实例的默认版本设置回 Oracle 默认值。

```
EXEC rdsadmin.rdsadmin_util.alter_default_edition('ORA$BASE');
```

有关基于 Oracle 版本的重新定义的更多信息，请参阅 Oracle 文档中的关于版本和基于版本的重新定义。

为 SYS.AUD$ 表启用审核

要在数据库审核跟踪表 SYS.AUD$ 上启用审核，请使用 Amazon RDS 过程 rdsadmin.rdsadmin_master_util.audit_all_sys_aud_table。唯一受支持的审核属性为 ALL。您无法审核或不审核单个语句或操作。

运行以下版本的 Oracle 数据库实例支持启用审核：

- Oracle Database 21c (21.0.0)
- Oracle Database 19c（19.0.0）

audit_all_sys_aud_table 过程具有以下参数。

参数名称	数据类型	默认值	必需	描述
p_by_access	布尔值	true	否	设置为 true 以审核 BY ACCESS。设置为 false 以审核 BY SESSION。

以下查询返回数据库的 SYS.AUD$ 的当前审核配置。

```
SELECT * FROM DBA_OBJ_AUDIT_OPTS WHERE OWNER='SYS' AND OBJECT_NAME='AUD$';
```

以下命令启用 ALL SYS.AUD$ 上的 BY ACCESS 的审核。

```
EXEC rdsadmin.rdsadmin_master_util.audit_all_sys_aud_table;

EXEC rdsadmin.rdsadmin_master_util.audit_all_sys_aud_table(p_by_access => true);
```

以下命令启用 ALL SYS.AUD$ 上的 BY SESSION 的审核。

```
EXEC rdsadmin.rdsadmin_master_util.audit_all_sys_aud_table(p_by_access => false);
```

有关更多信息，请参阅 Oracle 文档中的 AUDIT（传统审核）。

为 SYS.AUD$ 表禁用审核

要在数据库审核跟踪表 SYS.AUD$ 上禁用审核，请使用 Amazon RDS 过程 rdsadmin.rdsadmin_master_util.noaudit_all_sys_aud_table。此过程不需要任何参数。

以下查询返回数据库的 SYS.AUD$ 的当前审核配置：

```
SELECT * FROM DBA_OBJ_AUDIT_OPTS WHERE OWNER='SYS' AND OBJECT_NAME='AUD$';
```

以下命令禁用 ALL 上的 SYS.AUD$ 的审计。

```
EXEC rdsadmin.rdsadmin_master_util.noaudit_all_sys_aud_table;
```

有关更多信息，请参阅 Oracle 文档中的 NOAUDIT（传统审核）。

清理中断的在线索引构建

要清理失败的在线索引构建，请使用 Amazon RDS 过程 rdsadmin.rdsadmin_dbms_repair.online_index_clean。

online_index_clean 过程具有以下参数。

参数名称	数据类型	默认值	必需	描述
object_id	binary_integer	ALL_INDEX_ID	否	索引的对象 ID。通常，您可以使用 ORA-08104 错误文本中的对象 ID。
wait_for_lock	binary_integer	rdsadmin.rdsadmin_dbms_repair.lock_wait	否	指定 rdsadmin.rdsadmin_dbms_repair.lock_wait（默认值），尝试锁定底层对象，在锁定失败时重试，直至达到内部限制。 指定 rdsadmin.rdsadmin_dbms_repair.lock_nowait，

参数名称	数据类型	默认值	必需	描述
				尝试锁定底层对象，但在锁定失败时不重试。

下面的示例清理失败的在线索引构建：

```
declare
  is_clean boolean;
begin
  is_clean := rdsadmin.rdsadmin_dbms_repair.online_index_clean(
    object_id     => 1234567890,
    wait_for_lock => rdsadmin.rdsadmin_dbms_repair.lock_nowait
  );
end;
/
```

有关更多信息，请参阅 Oracle 文档中的 ONLINE_INDEX_CLEAN 函数。

跳过损坏的数据块

要在索引和表扫描期间跳过损坏的数据块，请使用 `rdsadmin.rdsadmin_dbms_repair` 程序包。

下面的过程封装了 `sys.dbms_repair.admin_table` 过程的功能，并且不带任何参数：

- `rdsadmin.rdsadmin_dbms_repair.create_repair_table`
- `rdsadmin.rdsadmin_dbms_repair.create_orphan_keys_table`
- `rdsadmin.rdsadmin_dbms_repair.drop_repair_table`
- `rdsadmin.rdsadmin_dbms_repair.drop_orphan_keys_table`
- `rdsadmin.rdsadmin_dbms_repair.purge_repair_table`
- `rdsadmin.rdsadmin_dbms_repair.purge_orphan_keys_table`

下面的过程采用与其 Oracle 数据库 `DBMS_REPAIR` 程序包中的对应项相同的参数。

- `rdsadmin.rdsadmin_dbms_repair.check_object`
- `rdsadmin.rdsadmin_dbms_repair.dump_orphan_keys`
- `rdsadmin.rdsadmin_dbms_repair.fix_corrupt_blocks`
- `rdsadmin.rdsadmin_dbms_repair.rebuild_freelists`

- rdsadmin.rdsadmin_dbms_repair.segment_fix_status
- rdsadmin.rdsadmin_dbms_repair.skip_corrupt_blocks

有关处理数据库损坏的更多信息，请参阅 Oracle 文档中的 DBMS_REPAIR。

Example 响应损坏的数据块

此示例显示了响应损坏的数据块的基本工作流程。您的步骤将取决于数据块损坏的位置和性质。

> **Important**
> 在尝试修复损坏的数据块之前，请仔细查看 DBMS_RECH 文档。

在索引和表扫描期间跳过损坏的数据块

1. 如果还没有修复表，则运行以下程序来创建修复表。

```
EXEC rdsadmin.rdsadmin_dbms_repair.create_repair_table;
EXEC rdsadmin.rdsadmin_dbms_repair.create_orphan_keys_table;
```

2. 运行以下程序以检查现有记录，适当时清除它们。

```
SELECT COUNT(*) FROM SYS.REPAIR_TABLE;
SELECT COUNT(*) FROM SYS.ORPHAN_KEY_TABLE;
SELECT COUNT(*) FROM SYS.DBA_REPAIR_TABLE;
SELECT COUNT(*) FROM SYS.DBA_ORPHAN_KEY_TABLE;

EXEC rdsadmin.rdsadmin_dbms_repair.purge_repair_table;
EXEC rdsadmin.rdsadmin_dbms_repair.purge_orphan_keys_table;
```

3. 运行以下程序检查损坏的数据块。

```
SET SERVEROUTPUT ON
DECLARE v_num_corrupt INT;
BEGIN
  v_num_corrupt := 0;
  rdsadmin.rdsadmin_dbms_repair.check_object (
    schema_name => '&corruptionOwner',
    object_name => '&corruptionTable',
    corrupt_count =>  v_num_corrupt
```

```
  );
  dbms_output.put_line('number corrupt: '||to_char(v_num_corrupt));
END;
/

COL CORRUPT_DESCRIPTION FORMAT a30
COL REPAIR_DESCRIPTION FORMAT a30

SELECT OBJECT_NAME, BLOCK_ID, CORRUPT_TYPE, MARKED_CORRUPT,
       CORRUPT_DESCRIPTION, REPAIR_DESCRIPTION
FROM   SYS.REPAIR_TABLE;

SELECT SKIP_CORRUPT
FROM   DBA_TABLES
WHERE  OWNER = '&corruptionOwner'
AND    TABLE_NAME = '&corruptionTable';
```

4. 使用 skip_corrupt_blocks 过程为受影响的表启用或禁用损坏跳过。根据具体情况，您可能还需要将数据提取到新表，然后删除包含损坏的数据块的表。

   运行以下程序为受影响的表启用损坏跳过。

```
begin
  rdsadmin.rdsadmin_dbms_repair.skip_corrupt_blocks (
    schema_name => '&corruptionOwner',
    object_name => '&corruptionTable',
    object_type => rdsadmin.rdsadmin_dbms_repair.table_object,
    flags => rdsadmin.rdsadmin_dbms_repair.skip_flag);
end;
/
select skip_corrupt from dba_tables where owner = '&corruptionOwner' and table_name
 = '&corruptionTable';
```

   运行以下程序禁用损坏跳过。

```
begin
  rdsadmin.rdsadmin_dbms_repair.skip_corrupt_blocks (
    schema_name => '&corruptionOwner',
    object_name => '&corruptionTable',
    object_type => rdsadmin.rdsadmin_dbms_repair.table_object,
    flags => rdsadmin.rdsadmin_dbms_repair.noskip_flag);
end;
/
```

```
select skip_corrupt from dba_tables where owner = '&corruptionOwner' and table_name
 = '&corruptionTable';
```

5. 完成所有修复工作后，运行以下过程删除修复表。

```
EXEC rdsadmin.rdsadmin_dbms_repair.drop_repair_table;
EXEC rdsadmin.rdsadmin_dbms_repair.drop_orphan_keys_table;
```

调整表空间、数据文件和临时文件的大小

原定设置情况下，在开启自动扩展的情况下创建 Oracle 表空间，并且没有最大大小。由于这些原定设置，表空间有时可能会变得过大。建议您在永久表空间和临时表空间上指定适当的最大大小，并仔细监视空间使用率。

调整永久表空间大小

要调整 RDS for Oracle 数据库实例中永久表空间的大小，请使用以下任何 Amazon RDS 过程：

- `rdsadmin.rdsadmin_util.resize_datafile`
- `rdsadmin.rdsadmin_util.autoextend_datafile`

`resize_datafile` 过程具有以下参数。

参数名称	数据类型	默认值	必需	描述
`p_data_file_id`	number	—	是	要调整大小的数据文件的标识符。
`p_size`	varchar2	—	是	数据文件的大小。以字节数（原定设置）、KB（K）、MB（M）或 GB（G）为单位指定大小。

`autoextend_datafile` 过程具有以下参数。

参数名称	数据类型	默认值	必需	描述
p_data_file_id	number	—	是	要调整大小的数据文件的标识符。
p_autoextend_state	varchar2	—	是	自动扩展功能的状态。指定 ON 可自动扩展数据文件，指定 OFF 可关闭自动扩展功能。
p_next	varchar2	—	否	下一个数据文件增量的大小。以字节数（原定设置）、KB（K）、MB（M）或 GB（G）为单位指定大小。
p_maxsize	varchar2	—	否	允许自动扩展的最大磁盘空间。以字节数（原定设置）、KB（K）、MB（M）或 GB（G）为单位指定大小。您可以指定 UNLIMITED 以删除文件大小限制。

以下示例将数据文件 4 的大小调整为 500MB。

```
EXEC rdsadmin.rdsadmin_util.resize_datafile(4,'500M');
```

以下示例对于数据文件 4 关闭自动扩展功能。该示例还为数据文件 5 开启自动扩展功能，增量为 128MB，且没有最大大小。

```
EXEC rdsadmin.rdsadmin_util.autoextend_datafile(4,'OFF');
EXEC rdsadmin.rdsadmin_util.autoextend_datafile(5,'ON','128M','UNLIMITED');
```

调整临时表空间大小

要调整 RDS for Oracle 数据库实例中临时表空间（包括只读副本）的大小，请使用以下任何 Amazon RDS 过程：

- `rdsadmin.rdsadmin_util.resize_temp_tablespace`
- `rdsadmin.rdsadmin_util.resize_tempfile`
- `rdsadmin.rdsadmin_util.autoextend_tempfile`

`resize_temp_tablespace` 过程具有以下参数。

参数名称	数据类型	默认值	必需	描述
`p_temp_tablespace_name`	varchar2	—	是	要调整大小的临时表空间的名称。
`p_size`	varchar2	—	是	表空间的大小。以字节数（原定设置）、KB（K）、MB（M）或 GB（G）为单位指定大小。

`resize_tempfile` 过程具有以下参数。

参数名称	数据类型	默认值	必需	描述
`p_temp_file_id`	number	—	是	要调整大小的临时文件的标识符。
`p_size`	varchar2	—	是	临时文件的大小。以字节数（原定设置）、KB（K）、MB（M）或 GB（G）为单位指定大小。

`autoextend_tempfile` 过程具有以下参数。

参数名称	数据类型	默认值	必需	描述
p_temp_file_id	number	—	是	要调整大小的临时文件的标识符。
p_autoextend_state	varchar2	—	是	自动扩展功能的状态。指定 ON 可自动扩展临时文件，指定 OFF 可关闭自动扩展功能。
p_next	varchar2	—	否	下一个临时文件增量的大小。以字节数（原定设置）、KB（K）、MB（M）或 GB（G）为单位指定大小。
p_maxsize	varchar2	—	否	允许自动扩展的最大磁盘空间。以字节数（原定设置）、KB（K）、MB（M）或 GB（G）为单位指定大小。您可以指定 UNLIMITED 以删除文件大小限制。

以下示例将名为 TEMP 的临时表空间的大小调整为 4GB。

```
EXEC rdsadmin.rdsadmin_util.resize_temp_tablespace('TEMP','4G');
```

```
EXEC rdsadmin.rdsadmin_util.resize_temp_tablespace('TEMP','4096000000');
```

以下示例将基于临时文件（文件标识符为 1）的临时表空间大小调整为 2MB。

```
EXEC rdsadmin.rdsadmin_util.resize_tempfile(1,'2M');
```

以下示例对于临时文件 1 关闭自动扩展功能。该示例还将临时文件 2 的最大自动扩展大小设置为 10GB，增量为 100MB。

```
EXEC rdsadmin.rdsadmin_util.autoextend_tempfile(1,'OFF');
EXEC rdsadmin.rdsadmin_util.autoextend_tempfile(2,'ON','100M','10G');
```

有关 Oracle 数据库实例的只读副本的更多信息，请参阅使用 Amazon RDS for Oracle 的只读副本。

清除回收站

删除表时，Oracle 数据库不会立即清除其存储空间。数据库将重命名表，并将其与所有关联的对象放在回收站中。清除回收站将删除这些项目并释放其存储空间。

要清除整个回收站，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_util.purge_dba_recyclebin`。但是，此过程无法清除回收站中的 SYS 和 RDSADMIN 对象。如果您需要清除这些对象，请联系 Amazon Support。

以下示例清除整个回收站。

```
EXEC rdsadmin.rdsadmin_util.purge_dba_recyclebin;
```

为全部编写设置原定设置显示值

要更改 Amazon RDS Oracle 实例上全部编写的原定设置显示值，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_util.dbms_redact_upd_full_rdct_val`。请注意，您使用 DBMS_REDACT PL/SQL 包创建了一个编写策略，如 Oracle 数据库文档中所述。`dbms_redact_upd_full_rdct_val` 过程为受现有策略影响的不同数据类型指定要显示的字符。

`dbms_redact_upd_full_rdct_val` 过程具有以下参数。

参数名称	数据类型	默认值	必需	描述
`p_number_val`	number	Null	否	修改 NUMBER 数据类型的列的原定设置值。
`p_binfloat_val`	binary_float	Null	否	修改 BINARY_FLOAT 数据类型的列的原定设置值。

参数名称	数据类型	默认值	必需	描述
p_bindouble_val	binary_double	Null	否	修改 BINARY_DOUBLE 数据类型的列的原定设置值。
p_char_val	char	Null	否	修改 CHAR 数据类型的列的原定设置值。
p_varchar_val	varchar2	Null	否	修改 VARCHAR2 数据类型的列的原定设置值。
p_nchar_val	nchar	Null	否	修改 NCHAR 数据类型的列的原定设置值。
p_nvarchar_val	nvarchar2	Null	否	修改 NVARCHAR2 数据类型的列的原定设置值。
p_date_val	date	Null	否	修改 DATE 数据类型的列的原定设置值。
p_ts_val	时间戳	Null	否	修改 TIMESTAMP 数据类型的列的原定设置值。
p_tswtz_val	timestamp with time zone	Null	否	修改 TIMESTAMP WITH TIME ZONE 数据类型的列的原定设置值。
p_blob_val	blob	Null	否	修改 BLOB 数据类型的列的原定设置值。
p_clob_val	clob	Null	否	修改 CLOB 数据类型的列的原定设置值。
p_nclob_val	nclob	Null	否	修改 NCLOB 数据类型的列的原定设置值。

以下示例将 CHAR 数据类型的原定设置编写值更改为 *：

```
EXEC rdsadmin.rdsadmin_util.dbms_redact_upd_full_rdct_val(p_char_val => '*');
```

以下示例更改 NUMBER、DATE 和 CHAR 数据类型的原定设置编写值：

```
BEGIN
rdsadmin.rdsadmin_util.dbms_redact_upd_full_rdct_val(
    p_number_val=>1,
    p_date_val=>to_date('1900-01-01','YYYY-MM-DD'),
    p_varchar_val=>'X');
END;
/
```

使用 dbms_redact_upd_full_rdct_val 过程更改全部编写的原定设置值后，请重启数据库实例以使更改生效。有关更多信息，请参阅 重启中的数据库实例。

执行 Oracle 数据库实例的常见日志相关任务

下文中介绍了如何在运行 Oracle 的 Amazon RDS 数据库实例上，执行与日志记录相关的特定常见 DBA 任务。为了提供托管服务体验，Amazon RDS 不允许通过 shell 访问数据库实例，而仅限访问某些需要高级权限的系统过程和表。

有关更多信息，请参阅“Amazon RDS for Oracle 数据库日志文件”。

主题

- 设置强制日志记录
- 设置补充日志记录
- 切换联机日志文件
- 添加联机重做日志
- 删除联机重做日志
- 调整联机重做日志的大小
- 保留存档重做日志
- 访问在线和存档的重做日志
- 从 Amazon S3 下载存档的重做日志

设置强制日志记录

在强制日志记录模式下，Oracle 记录对数据库进行的所有更改，但在临时表空间和临时段中进行的更改除外 (忽略 NOLOGGING 子句)。有关更多信息，请参阅 Oracle 文档中的指定“强制日志记录”模式。

要设置强制日志记录，请使用 Amazon RDS 过程 rdsadmin.rdsadmin_util.force_logging。force_logging 过程具有以下参数。

参数名称	数据类型	默认值	是	描述
p_enable	布尔值	true	否	设置为 true，数据库为强制日志记录模式；设置为 false 则取消数据库的强制日志记录模式。

以下示例将数据库置于强制日志记录模式下。

```
EXEC rdsadmin.rdsadmin_util.force_logging(p_enable => true);
```

设置补充日志记录

如果启用补充日志记录，则 LogMiner 将具有支持链接行和集群表的必要信息。有关更多信息，请参阅 Oracle 文档中的补充日志记录。

Oracle 数据库在默认情况下不启用补充日志记录。要启用和禁用补充日志记录，请使用 Amazon RDS 过程 rdsadmin.rdsadmin_util.alter_supplemental_logging。有关 Amazon RDS 如何管理 Oracle 数据库实例的存档重做日志的保留期的更多信息，请参阅 保留存档重做日志。

alter_supplemental_logging 过程具有以下参数。

参数名称	数据类型	默认值	必需	描述
p_action	varchar2	—	是	如果为 'ADD'，则添加补充日志记录；如果为 'DROP'，则删除补充日志记录。

参数名称	数据类型	默认值	必需	描述
p_type	varchar2	null	否	补充日志记录的类型。有效值为 'ALL'、'FOREIGN KEY'、'PRIMARY KEY'、'UNIQUE' 或 PROCEDURAL 。

以下示例启用补充日志记录。

```
begin
    rdsadmin.rdsadmin_util.alter_supplemental_logging(
        p_action => 'ADD');
end;
/
```

以下示例为所有具有固定长度的最大大小列启用补充日志记录。

```
begin
    rdsadmin.rdsadmin_util.alter_supplemental_logging(
        p_action => 'ADD',
        p_type   => 'ALL');
end;
/
```

以下示例为主键列启用补充日志记录。

```
begin
    rdsadmin.rdsadmin_util.alter_supplemental_logging(
        p_action => 'ADD',
        p_type   => 'PRIMARY KEY');
end;
/
```

切换联机日志文件

要切换日志文件，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_util.switch_logfile`。`switch_logfile` 过程没有参数。

以下示例切换日志文件。

```
EXEC rdsadmin.rdsadmin_util.switch_logfile;
```

添加联机重做日志

运行 Oracle 的 Amazon RDS 数据库实例从四个联机重做日志 (每个日志的大小为 128 MB) 开始。要添加其他重做日志，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_util.add_logfile`。

`add_logfile` 过程具有以下参数。

> **Note**
> 这些参数是互斥的。

参数名称	数据类型	默认值	必需	描述
`bytes`	positive	null	否	日志文件的大小 (以字节为单位)。 仅当日志大小小于 2147483648 字节（2 GiB）时才使用此参数。否则，RDS 会发出错误。对于大于此字节值的日志大小，请改用 `p_size` 参数。
`p_size`	varchar2	—	是	日志文件的大小，单位为 KB（K）、MB（M）或 GB（G）。

以下命令将添加一个 100MB 日志文件。

```
EXEC rdsadmin.rdsadmin_util.add_logfile(p_size => '100M');
```

删除联机重做日志

要删除重做日志，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_util.drop_logfile`。`drop_logfile` 过程具有以下参数。

参数名称	数据类型	默认值	必需	描述
`grp`	positive	—	是	日志的组编号。

以下示例删除组编号为 3 的日志。

```
EXEC rdsadmin.rdsadmin_util.drop_logfile(grp => 3);
```

您只能删除具有未使用或不活动状态的日志。以下示例获取日志的状态。

```
SELECT GROUP#, STATUS FROM V$LOG;

GROUP#     STATUS
---------- ----------------
1          CURRENT
2          INACTIVE
3          INACTIVE
4          UNUSED
```

调整联机重做日志的大小

运行 Oracle 的 Amazon RDS 数据库实例从四个联机重做日志 (每个日志的大小为 128 MB) 开始。以下示例说明如何使用 Amazon RDS 过程将每个日志的大小从 128 MB 调整到 512 MB。

```
/* Query V$LOG to see the logs.          */
/* You start with 4 logs of 128 MB each. */

SELECT GROUP#, BYTES, STATUS FROM V$LOG;

GROUP#     BYTES      STATUS
---------- ---------- ----------------
1          134217728  INACTIVE
2          134217728  CURRENT
3          134217728  INACTIVE
4          134217728  INACTIVE
```

```
/* Add four new logs that are each 512 MB */

EXEC rdsadmin.rdsadmin_util.add_logfile(bytes => 536870912);
EXEC rdsadmin.rdsadmin_util.add_logfile(bytes => 536870912);
EXEC rdsadmin.rdsadmin_util.add_logfile(bytes => 536870912);
EXEC rdsadmin.rdsadmin_util.add_logfile(bytes => 536870912);


/* Query V$LOG to see the logs. */
/* Now there are 8 logs.        */

SELECT GROUP#, BYTES, STATUS FROM V$LOG;

GROUP#     BYTES      STATUS
---------- ---------- ----------------
1          134217728  INACTIVE
2          134217728  CURRENT
3          134217728  INACTIVE
4          134217728  INACTIVE
5          536870912  UNUSED
6          536870912  UNUSED
7          536870912  UNUSED
8          536870912  UNUSED


/* Drop each inactive log using the group number. */

EXEC rdsadmin.rdsadmin_util.drop_logfile(grp => 1);
EXEC rdsadmin.rdsadmin_util.drop_logfile(grp => 3);
EXEC rdsadmin.rdsadmin_util.drop_logfile(grp => 4);


/* Query V$LOG to see the logs. */
/* Now there are 5 logs.        */

select GROUP#, BYTES, STATUS from V$LOG;

GROUP#     BYTES      STATUS
---------- ---------- ----------------
2          134217728  CURRENT
5          536870912  UNUSED
6          536870912  UNUSED
```

```
7          536870912  UNUSED
8          536870912  UNUSED


/* Switch logs so that group 2 is no longer current. */

EXEC rdsadmin.rdsadmin_util.switch_logfile;


/* Query V$LOG to see the logs.        */
/* Now one of the new logs is current. */

SQL>SELECT GROUP#, BYTES, STATUS FROM V$LOG;

GROUP#     BYTES      STATUS
---------- ---------- ----------------
2          134217728  ACTIVE
5          536870912  CURRENT
6          536870912  UNUSED
7          536870912  UNUSED
8          536870912  UNUSED


/* If the status of log 2 is still "ACTIVE", issue a checkpoint to clear it to
 "INACTIVE".  */

EXEC rdsadmin.rdsadmin_util.checkpoint;


/* Query V$LOG to see the logs.            */
/* Now the final original log is inactive. */

select GROUP#, BYTES, STATUS from V$LOG;

GROUP#     BYTES      STATUS
---------- ---------- ----------------
2          134217728  INACTIVE
5          536870912  CURRENT
6          536870912  UNUSED
7          536870912  UNUSED
8          536870912  UNUSED


# Drop the final inactive log.
```

```
EXEC rdsadmin.rdsadmin_util.drop_logfile(grp => 2);


/* Query V$LOG to see the logs.    */
/* Now there are four 512 MB logs. */

SELECT GROUP#, BYTES, STATUS FROM V$LOG;

GROUP#     BYTES      STATUS
---------- ---------- ----------------
5          536870912  CURRENT
6          536870912  UNUSED
7          536870912  UNUSED
8          536870912  UNUSED
```

保留存档重做日志

您可以在数据库实例上本地保留存档的重做日志，以用于 Oracle LogMiner（DBMS_LOGMNR）这类产品。保留重做日志后，您可以使用 LogMiner 分析日子。有关更多信息，请参阅 Oracle 文档中的使用 LogMiner 分析重做日志文件。

要保留存档的重做日志，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_util.set_configuration`。如果您在 Oracle Data Guard 中的主实例上使用此过程，则 RDS 会更改主实例和打开只读副本的存档日志保留设置，但不会更改已挂载副本上的设置。RDS 会短时间保留已挂载副本的最新存档重做日志。RDS 会自动删除下载到已挂载副本的较旧日志。

`set_configuration` 过程具有以下参数。

参数名称	数据类型	默认值	必需	描述
`name`	varchar	—	是	要更新的配置的名称。要更改已存档的重做日志的保留时间，请将名称设置为 `archivelog retention hours`。
`value`	varchar	—	是	该配置的值。设置日志要保留的小时数的值。

以下示例保留 24 小时的重做日志。

```
begin
    rdsadmin.rdsadmin_util.set_configuration(
        name  => 'archivelog retention hours',
        value => '24');
end;
/
commit;
```

> **Note**
>
> 要使更改生效，必须执行提交操作。

要查看存档的重做日志为数据库实例保留多长时间，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_util.show_configuration`。

以下示例显示日志保留时间。

```
set serveroutput on
EXEC rdsadmin.rdsadmin_util.show_configuration;
```

输出显示 `archivelog retention hours` 的当前设置。以下输出显示存档的重做日志保留 48 小时。

```
NAME:archivelog retention hours
VALUE:48
DESCRIPTION:ArchiveLog expiration specifies the duration in hours before archive/redo
 log files are automatically deleted.
```

由于存档的重做日志保留在数据库实例中，因此，请确保数据库实例已为保留的日志分配足量的存储空间。要确定数据库实例上在最近 X 小时内使用的空间量，可以运行以下查询（将 X 替换为小时数）。

```
SELECT SUM(BLOCKS * BLOCK_SIZE) bytes
  FROM V$ARCHIVED_LOG
 WHERE FIRST_TIME >= SYSDATE-(X/24) AND DEST_ID=1;
```

仅在数据库实例的备份保留期大于零时，RDS for Oracle 才会生成存档的重做日志。默认情况下，备份保留期大于零。

当存档的日志保留期过期时，RDS for Oracle 将从数据库实例中删除归档的重做日志。为支持将数据库实例还原到某个时间点，Amazon RDS 会根据备份保留期在数据库实例之外保留存档的重做日志。要修改备份保留期，请参阅 修改 Amazon RDS 数据库实例。

> **Note**
> 在一些情况下，您可以在 Linux 上使用 JDBC 下载存档重做日志并体验长延迟时间和连接重置。在此类情况下，您的 Java 客户端上的默认随机数字生成器设置可能会引发问题。我们建议您将 JDBC 驱动程序设置为使用非阻塞的随机数字生成器。

访问在线和存档的重做日志

您可能需要访问在线和存档重做日志文件，以供 GoldenGate、Attunity、Informatica 等外部工具进行挖掘。要访问这些文件，请执行以下操作：

1. 创建目录对象，对物理文件路径提供只读访问权限。

   使用 `rdsadmin.rdsadmin_master_util.create_archivelog_dir` 和 `rdsadmin.rdsadmin_master_util.create_onlinelog_dir`。
2. 使用 PL/SQL 读取文件。

   您可以使用 PL/SQL 读取文件。有关从目录对象读取文件的更多信息，请参阅 列出数据库实例目录中的文件 和 读取数据库实例目录中的文件。

以下版本支持访问事务日志：

- Oracle Database 21c
- Oracle Database 19c

以下代码用于创建目录，为您的在线和存档重做日志文件提供只读访问权限。

> **Important**
> 此代码还会撤销 `DROP ANY DIRECTORY` 权限。

```
EXEC rdsadmin.rdsadmin_master_util.create_archivelog_dir;
```

```
EXEC rdsadmin.rdsadmin_master_util.create_onlinelog_dir;
```

以下代码会删掉在线和存档重做日志文件的目录。

```
EXEC rdsadmin.rdsadmin_master_util.drop_archivelog_dir;
EXEC rdsadmin.rdsadmin_master_util.drop_onlinelog_dir;
```

以下代码用于授予和撤销 `DROP ANY DIRECTORY` 权限。

```
EXEC rdsadmin.rdsadmin_master_util.revoke_drop_any_directory;
EXEC rdsadmin.rdsadmin_master_util.grant_drop_any_directory;
```

从 Amazon S3 下载存档的重做日志

您可以使用 `rdsadmin.rdsadmin_archive_log_download` 软件包在数据库实例上下载存档的重做日志。如果存档的重做日志不再存在于数据库实例上，您可能需要从 Amazon S3 重新下载它们。然后，您可以挖掘日志或使用日志恢复或复制数据库。

> **Note**
>
> 您无法在只读副本实例上下载存档的重做日志。

下载归档的重做日志：基本步骤

归档的重做日志的可用性取决于以下保留策略：

- 备份保留策略 – Amazon S3 提供了此策略内的日志。此策略之外的日志将被删除。
- 归档的日志保留策略 – 此策略内的日志在数据库实例上可用。此策略之外的日志将被删除。

如果日志不在您的实例上，但受到备份保留期的保护，请使用 `rdsadmin.rdsadmin_archive_log_download` 以重新下载。RDS for Oracle 会将日志保存到数据库实例的 `/rdsdbdata/log/arch` 目录。

从 Amazon S3 下载归档的重做日志

1. 配置您的保留期，以确保您下载的归档重做日志在您需要的时间内得到保留。请确保 `COMMIT` 您的更改。

RDS 根据归档日志保留策略保留您下载的日志，从下载日志之时开始。要了解如何设置保留策略，请参阅保留存档重做日志。

2. 最多等待 5 分钟，使归档的日志保留策略更改生效。
3. 使用 `rdsadmin.rdsadmin_archive_log_download` 从 Amazon S3 下载归档的重做日志。

   有关更多信息，请参阅“下载单个归档的重做日志”和“下载一系列归档的重做日志”。

   > **Note**
   >
   > RDS 在下载之前会自动检查可用的存储空间。如果请求的日志占用大量空间，您将收到警报。

4. 确认日志已成功从 Amazon S3 下载。

   您可以查看 bdump 文件中的下载任务的状态。bdump 文件的路径名称为 `/rdsdbdata/log/trace/dbtask-`*`task-id`*`.log`。在前面的下载步骤中，您可以运行 `SELECT` 语句，该语句可返回 `VARCHAR2` 数据类型中的任务 ID。有关更多信息，请参阅监控文件传输的状态中的类似示例。

下载单个归档的重做日志

要将单个归档的重做日志下载到 `/rdsdbdata/log/arch` 目录中，请使用 `rdsadmin.rdsadmin_archive_log_download.download_log_with_seqnum`。此过程具有以下参数。

参数名称	数据类型	默认值	必需	描述
`seqnum`	number	—	是	归档的重做日志的序列号。

以下示例展示的是下载序列号为 20 的日志。

```
SELECT rdsadmin.rdsadmin_archive_log_download.download_log_with_seqnum(seqnum => 20)
       AS TASK_ID
FROM   DUAL;
```

下载一系列归档的重做日志

要将一系列归档的重做日志下载到 /rdsdbdata/log/arch 目录中，请使用 download_logs_in_seqnum_range。每个请求的下载限制为 300 个日志。download_logs_in_seqnum_range 过程具有以下参数。

参数名称	数据类型	默认值	必需	描述
start_seq	number	—	是	系列的起始序列号。
end_seq	number	—	是	系列的结束序列号。

以下示例展示的是下载序列号为 50 至 100 的日志。

```
SELECT rdsadmin.rdsadmin_archive_log_download.download_logs_in_seqnum_range(start_seq
 => 50, end_seq => 100)
       AS TASK_ID
FROM   DUAL;
```

执行 Oracle 数据库实例的常见 RMAN 任务

在下一节中，您可以了解如何在运行 Oracle 的 Amazon RDS 数据库实例上执行 Oracle Recovery Manager (RMAN) DBA 任务。为了提供托管服务体验，Amazon RDS 未提供数据库实例的 Shell 访问权限。并且它还限制了对需要高级权限的某些系统程序和表的访问。

使用 Amazon RDS 包 rdsadmin.rdsadmin_rman_util 执行 Amazon RDS for Oracle 数据库到磁盘的 RMAN 备份。rdsadmin.rdsadmin_rman_util 包支持完整和增量数据库文件备份、表空间备份和归档重做日志备份。

在 RMAN 备份完成后，您可以从 Amazon RDS for Oracle 数据库实例主机复制备份文件。您可能出于还原到非 RDS 主机目的或出于长期存储备份目的而执行此操作。例如，您可以将备份文件复制到 Amazon S3 存储桶。有关更多信息，请参阅使用 Amazon S3 集成。

RMAN 备份的备份文件保留在 Amazon RDS 数据库实例主机上，直到您手动删除它们。您可以使用 UTL_FILE.FREMOVE Oracle 过程从目录中删除文件。有关更多信息，请参阅 Oracle 数据库文档中的 FREMOVE 过程。

您不能使用 RMAN 还原 RDS for Oracle 数据库实例。但是，您可以使用 RMAN 将备份还原到本地或 Amazon EC2 实例。有关更多信息，请参阅博客文章将 Amazon RDS for Oracle 实例还原到自行管理的实例。

> **Note**
>
> 要备份和还原到另一个 Amazon RDS for Oracle 数据库实例，您可以继续使用 Amazon RDS 备份和还原特征。有关更多信息，请参阅 备份、还原和导出数据。

主题

RMAN 备份的先决条件

在使用 `rdsadmin.rdsadmin_rman_util` 包备份数据库之前，请确保满足以下先决条件：

- 确保您的 RDS for Oracle 数据库处于 `ARCHIVELOG` 模式。要启用此模式，请将备份保留期设置为非零值。
- 在备份归档重做日志或执行包括归档重做日志的完整或增量备份时，以及在备份数据库时，请确保将重做日志保留期设置为非零值。需要归档重做日志才能在恢复期间保持数据库文件的一致性。有关更多信息，请参阅 保留存档重做日志。

- 确保您的数据库实例有足够的可用空间来保存备份。备份数据库时，请将 Oracle 目录对象指定为过程调用中的一个参数。RMAN 将文件放在指定的目录中。您可以使用默认目录（如 DATA_PUMP_DIR），或创建新目录。有关更多信息，请参阅 在主数据存储空间中创建和删除目录。

  您可以使用 CloudWatch 指标 FreeStorageSpace 监控 RDS for Oracle 实例中的当前可用空间。尽管 RMAN 仅备份格式化的数据块并支持压缩，但我们建议您的可用空间超过数据库的当前大小。

RMAN 过程的常见参数

您可以使用 Amazon RDS 包 rdsadmin.rdsadmin_rman_util 中的过程来通过 RMAN 执行任务。有几个参数是包中的过程通用的。该包具有以下常见参数。

参数名称	数据类型	有效值	默认值	必需	描述
p_directory_name	varchar	有效的数据库目录名称。	—	是	包含备份文件的目录的名称。
p_label	varchar	a-z, A-Z, 0-9, '_', '-', '.'	—	否	包括在备份文件名中的唯一字符串。 Note 限制为 30 个字符。
p_owner	varchar	p_directory_name 中指定的目录的有效拥有者。	—	是	包含备份文件的目录的拥有者。
p_tag	varchar	a-z, A-Z, 0-9, '_', '-', '.'	NULL	否	可用于区分备份的字符串，以便指示备份的目的或用法，例如每日备份、每周备份或增量级备份。

参数名称	数据类型	有效值	默认值	必需	描述
					限制为 30 个字符。此标签不区分大小写。无论输入标签时使用的大小写如何，标签始终以大写形式存储。 标签不一定是唯一的，因此多个备份可以使用相同的标签。 如果未指定标签，则 RMAN 会使用 `TAG`*`YYYYMMDDTHHMMSS`* 格式自动分配默认标签，其中 *`YYYY`* 表示年份、*`MM`* 表示月份、*`DD`* 表示某天，*`HH`* 表示小时（采用 24 小时格式）、*`MM`* 表示分钟、*`SS`* 表示秒。日期和时间表示 RMAN 开始备份的时间。 例如，如果备份开始于 2019 年 9 月 27 日 21:45:17，则收到的标签可能为 `TAG20190927T214517` 。 以下 Amazon RDS for Oracle 数据库引擎版本支持 `p_tag` 参数： • Oracle Database 21c (21.0.0) • Oracle Database 19c (19.0.0)，使用 19.0.0.0.ru-2021-10.rur-2021-10.r1 或更高版本
`p_compress`	布尔值	`TRUE`, `FALSE`	`FALSE`	否	指定 `TRUE` 以启用基本备份压缩。 指定 `FALSE` 以禁用基本备份压缩。

参数名称	数据类型	有效值	默认值	必需	描述
p_include_archive_logs	布尔值	TRUE, FALSE	FALSE	否	指定 TRUE 以在备份中包括存档重做日志。 指定 FALSE 以从备份中排除存档重做日志。 如果您在备份中包括存档重做日志，请使用 rdsadmin.rdsadmin_util.set_configuration 过程将保留设置为一小时或更长时间。另外，就在要运行备份之前调用 rdsadmin.rdsadmin_rman_util.crosscheck_archivelog 过程。否则，由于缺少已被 Amazon RDS 管理过程删除的存档重做日志文件，备份可能会失败。
p_include_controlfile	布尔值	TRUE, FALSE	FALSE	否	指定 TRUE 以在备份中包括控制文件。 指定 FALSE 以从备份中排除控制文件。
p_optimize	布尔值	TRUE, FALSE	TRUE	否	指定 TRUE 以启用备份优化（如果包括存档重做日志）以减小备份大小。 指定 FALSE 以禁用备份优化。

参数名称	数据类型	有效值	默认值	必需	描述
p_parallel	number	对于 Oracle Database 企业版 (EE)，为介于 1 和 254 之间的有效整数。 1对于其他 Oracle Database 版本为 。	1	否	通道数。
p_rman_to_dbms_output	布尔值	TRUE, FALSE	FALSE	否	为 TRUE 时，在 DBMS_OUTPUT 目录中的文件之外，RMAN 输出还发送到 BDUMP 包。在 SQL*Plus 中，使用 SET SERVEROUTPUT ON 查看输出。 为 FALSE 时，RMAN 输出只发送到 BDUMP 目录中的文件。
p_section_size_mb	number	有效的整数	NULL	否	扇区大小，以兆字节 (MB) 为单位。 通过将每个文件拆分为指定扇区大小来并行验证。 为 NULL 时将忽略参数。

参数名称	数据类型	有效值	默认值	必需	描述
p_validation_type	varchar	'PHYSICAL', 'PHYSICAL+LOGICAL'	'PHYSICAL'	否	检测损坏的程度。 指定 'PHYSICAL' 检查物理损坏。物理损坏的例子包括数据块在标头和脚注中不匹配。 指定 'PHYSICAL+LOGICAL' 在物理损坏之外，还检查逻辑不一致的情况。逻辑损坏的例子包括数据块损坏。

在 RDS for Oracle 中验证数据库文件

您可以使用 Amazon RDS 包 `rdsadmin.rdsadmin_rman_util` 验证 Amazon RDS for Oracle 数据库文件，例如数据文件、表空间、控制文件以及服务器参数文件（SPFILE）。

有关 RMAN 验证的更多信息，请参阅[验证数据库文件和备份](#)以及 Oracle 文档中的 [VALIDATE](#)。

主题

验证数据库

要验证 RDS for Oracle 中的 Oracle 数据库使用的所有相关文件，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_rman_util.validate_database`。

该过程使用 RMAN 任务的以下常见参数：

- `p_validation_type`
- `p_parallel`

- p_section_size_mb
- p_rman_to_dbms_output

有关更多信息，请参阅 RMAN 过程的常见参数。

以下示例使用参数的默认值来验证数据库。

```
EXEC rdsadmin.rdsadmin_rman_util.validate_database;
```

以下示例使用参数的指定值来验证数据库。

```
BEGIN
    rdsadmin.rdsadmin_rman_util.validate_database(
        p_validation_type     => 'PHYSICAL+LOGICAL',
        p_parallel            => 4,
        p_section_size_mb     => 10,
        p_rman_to_dbms_output => FALSE);
END;
/
```

当 p_rman_to_dbms_output 参数设置为 FALSE 时，RMAN 输出写入到 BDUMP 目录中的文件。

要查看 BDUMP 目录中的文件，请运行以下 SELECT 语句。

```
SELECT * FROM table(rdsadmin.rds_file_util.listdir('BDUMP')) order by mtime;
```

要查看 BDUMP 目录中文件的内容，请运行以下 SELECT 语句。

```
SELECT text FROM table(rdsadmin.rds_file_util.read_text_file('BDUMP','rds-rman-validate-nnn.txt'));
```

将文件名替换为您要查看的文件的名称。

验证租户数据库

要验证容器数据库（CDB）中租户数据库的数据文件，请使用 Amazon RDS 过程 rdsadmin.rdsadmin_rman_util.validate_tenant。

该过程仅适用于当前租户数据库，并使用 RMAN 任务的以下常见参数：

- p_validation_type
- p_parallel
- p_section_size_mb
- p_rman_to_dbms_output

有关更多信息，请参阅 RMAN 过程的常见参数。以下数据库引擎版本支持此过程：

- Oracle Database 21c（21.0.0）CDB
- Oracle Database 19c（19.0.0）CDB

以下示例使用参数的默认值来验证当前的租户数据库。

```
EXEC rdsadmin.rdsadmin_rman_util.validate_tenant;
```

以下示例使用参数的指定值来验证当前的租户数据库。

```
BEGIN
    rdsadmin.rdsadmin_rman_util.validate_tenant(
        p_validation_type     => 'PHYSICAL+LOGICAL',
        p_parallel            => 4,
        p_section_size_mb     => 10,
        p_rman_to_dbms_output => FALSE);
END;
/
```

当 p_rman_to_dbms_output 参数设置为 FALSE 时，RMAN 输出写入到 BDUMP 目录中的文件。

要查看 BDUMP 目录中的文件，请运行以下 SELECT 语句。

```
SELECT * FROM table(rdsadmin.rds_file_util.listdir('BDUMP')) order by mtime;
```

要查看 BDUMP 目录中文件的内容，请运行以下 SELECT 语句。

```
SELECT text FROM table(rdsadmin.rds_file_util.read_text_file('BDUMP','rds-rman-
validate-nnn.txt'));
```

将文件名替换为您要查看的文件的名称。

验证表空间

要验证与表空间关联的文件，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_rman_util.validate_tablespace`。

该过程使用 RMAN 任务的以下常见参数：

- `p_validation_type`
- `p_parallel`
- `p_section_size_mb`
- `p_rman_to_dbms_output`

有关更多信息，请参阅“RMAN 过程的常见参数”。

此过程还使用以下其他参数。

参数名称	数据类型	有效值	默认值	必需	描述
`p_tablespace_name`	varchar2	有效的空间表名称	—	是	表空间的名称。

验证控制文件

要仅验证 Amazon RDS Oracle 数据库实例使用的控制文件，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_rman_util.validate_current_controlfile`。

该过程使用 RMAN 任务的以下常见参数：

- `p_validation_type`
- `p_rman_to_dbms_output`

有关更多信息，请参阅“RMAN 过程的常见参数”。

验证 SPFILE

要仅验证 Amazon RDS Oracle 数据库实例使用的服务器参数文件 (SPFILE)，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_rman_util.validate_spfile`。

该过程使用 RMAN 任务的以下常见参数：

- p_validation_type
- p_rman_to_dbms_output

有关更多信息，请参阅 RMAN 过程的常见参数。

验证 Oracle 数据文件

要验证数据文件，请使用 Amazon RDS 过程 rdsadmin.rdsadmin_rman_util.validate_datafile。

该过程使用 RMAN 任务的以下常见参数：

- p_validation_type
- p_parallel
- p_section_size_mb
- p_rman_to_dbms_output

有关更多信息，请参阅"RMAN 过程的常见参数"。

此过程还使用以下其他参数。

参数名称	数据类型	有效值	默认值	必需	描述
p_datafile	varchar2	有效的数据文件 ID 编号或包含完整路径的有效数据文件名称	—	是	数据文件 ID 编号（来自 v$datafile.file# ）或包含路径的完整数据文件名（来自 v$datafile.name ）。
p_from_block	number	有效的整数	NULL	否	在数据文件中开始验证的块的编号。当这为 NULL 时，将使用 1。
p_to_block	number	有效的整数	NULL	否	在数据文件中结束验证的块的编号。当这

参数名称	数据类型	有效值	默认值	必需	描述
					为 NULL 时，将使用数据文件中的最大块。

启用和禁用数据块更改跟踪

数据库更改跟踪记录更改了跟踪文件中的数据块。此技术可以提高 RMAN 增量备份的性能。有关详细信息，请参阅 Oracle 数据库文档中的使用数据块更改跟踪来提高增量备份性能。

只读副本不支持 RMAN 特征。但是，作为高可用性策略的一部分，您可以选择使用过程 `rdsadmin.rdsadmin_rman_util.enable_block_change_tracking` 在只读副本中启用数据块跟踪。如果您将此只读副本提升为源数据库实例，则会为新的源实例启用数据块更改跟踪。因此，您的实例可以受益于快速的增量备份。

以下数据库引擎版本仅在企业版中支持数据块更改跟踪过程：

- Oracle Database 21c (21.0.0)
- Oracle Database 19c（19.0.0）

> **Note**
>
> 在单租户 CDB 中，以下操作有效，但没有客户可见的机制可以检测操作的当前状态。另请参阅 RDS for Oracle CDB 的限制。

要为数据库实例启用数据块更改跟踪，您可以使用 Amazon RDS 过程 `rdsadmin.rdsadmin_rman_util.enable_block_change_tracking`。要禁用数据块更改跟踪，请使用 `disable_block_change_tracking`。这些过程不需要任何参数。

要确定是否为您的数据库实例启用了块更改跟踪，请运行以下查询。

```
SELECT STATUS, FILENAME FROM V$BLOCK_CHANGE_TRACKING;
```

以下示例为数据库实例启用块更改跟踪。

```
EXEC rdsadmin.rdsadmin_rman_util.enable_block_change_tracking;
```

以下示例为数据库实例禁用块更改跟踪。

```
EXEC rdsadmin.rdsadmin_rman_util.disable_block_change_tracking;
```

反复检查存档重做日志

您可以使用 Amazon RDS 过程 `rdsadmin.rdsadmin_rman_util.crosscheck_archivelog` 反复检查存档重做日志。

您可以使用此过程反复检查控制文件中注册的存档重做日志并可以选择删除过期的日志记录。当 RMAN 进行备份时，它会在控制文件中创建一条记录。随着时间的推移，这些记录会增加控制文件的大小。我们建议您定期删除过期的记录。

> **Note**
> 标准 Amazon RDS 备份不使用 RMAN，因此不会在控制文件中创建记录。

该过程使用 RMAN 任务的常见参数 `p_rman_to_dbms_output`。

有关更多信息，请参阅"RMAN 过程的常见参数"。

此过程还使用以下其他参数。

参数名称	数据类型	有效值	默认值	必需	描述
`p_delete_expired`	布尔值	`TRUE`, `FALSE`	`TRUE`	否	当为 `TRUE` 时，从控制文件中删除过期的存档重做日志记录。 当为 `FALSE` 时，保留控制文件中过期的存档重做日志记录。

以下 Amazon RDS for Oracle 数据库引擎版本支持此过程：

- Oracle Database 21c (21.0.0)
- Oracle Database 19c（19.0.0）

以下示例将控制文件中存档的重做日志记录标记为已过期，但不删除这些记录。

```
BEGIN
    rdsadmin.rdsadmin_rman_util.crosscheck_archivelog(
        p_delete_expired      => FALSE,
        p_rman_to_dbms_output => FALSE);
END;
/
```

以下示例从控制文件中删除过期的存档重做日志记录。

```
BEGIN
    rdsadmin.rdsadmin_rman_util.crosscheck_archivelog(
        p_delete_expired      => TRUE,
        p_rman_to_dbms_output => FALSE);
END;
/
```

备份归档的重做日志文件

您可以使用 Amazon RDS 包 `rdsadmin.rdsadmin_rman_util` 备份 Amazon RDS Oracle 数据库实例的存档重做日志。

以下 Amazon RDS for Oracle 数据库引擎版本支持备份存档重做日志的过程：

- Oracle Database 21c (21.0.0)
- Oracle Database 19c（19.0.0）

主题

- 备份所有存档重做日志
- 备份日期范围中的存档重做日志
- 备份 SCN 范围中的存档重做日志
- 备份序列号范围中的存档重做日志

备份所有存档重做日志

要备份 Amazon RDS Oracle 数据库实例的所有存档重做日志，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_rman_util.backup_archivelog_all`。

该过程使用 RMAN 任务的以下常见参数：

- p_owner
- p_directory_name
- p_label
- p_parallel
- p_compress
- p_rman_to_dbms_output
- p_tag

有关更多信息，请参阅“RMAN 过程的常见参数”。

以下示例备份数据库实例的所有存档重做日志。

```
BEGIN
    rdsadmin.rdsadmin_rman_util.backup_archivelog_all(
        p_owner              => 'SYS',
        p_directory_name     => 'MYDIRECTORY',
        p_parallel           => 4,
        p_tag                => 'MY_LOG_BACKUP',
        p_rman_to_dbms_output => FALSE);
END;
/
```

备份日期范围中的存档重做日志

要指定日期范围以备份 Amazon RDS Oracle 数据库实例的特定存档重做日志，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_rman_util.backup_archivelog_date`。日期范围指定要备份哪些存档重做日志。

该过程使用 RMAN 任务的以下常见参数：

- p_owner
- p_directory_name
- p_label
- p_parallel

- p_compress
- p_rman_to_dbms_output
- p_tag

有关更多信息，请参阅“RMAN 过程的常见参数”。

此过程还使用以下其他参数。

参数名称	数据类型	有效值	默认值	必需	描述
p_from_date	date	介于磁盘上存在的存档重做日志的 start_date 和 next_date 之间的日期。该值必须小于或等于为 p_to_date 指定的值。	—	是	存档日志备份的起始日期。
p_to_date	date	介于磁盘上存在的存档重做日志的 start_date 和 next_date 之间的日期。该值必须大于或	—	是	存档日志备份的结束日期。

参数名称	数据类型	有效值	默认值	必需	描述
		等于为 p_from_date 指定的值。			

以下示例备份数据库实例的日期范围中的存档重做日志。

```
BEGIN
    rdsadmin.rdsadmin_rman_util.backup_archivelog_date(
        p_owner               => 'SYS',
        p_directory_name      => 'MYDIRECTORY',
        p_from_date           => '03/01/2019 00:00:00',
        p_to_date             => '03/02/2019 00:00:00',
        p_parallel            => 4,
        p_tag                 => 'MY_LOG_BACKUP',
        p_rman_to_dbms_output => FALSE);
END;
/
```

备份 SCN 范围中的存档重做日志

要指定系统更改号 (SCN) 范围以备份 Amazon RDS Oracle 数据库实例的特定存档重做日志，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_rman_util.backup_archivelog_scn`。SCN 范围指定要备份哪些存档重做日志。

该过程使用 RMAN 任务的以下常见参数：

- `p_owner`
- `p_directory_name`
- `p_label`
- `p_parallel`
- `p_compress`
- `p_rman_to_dbms_output`
- `p_tag`

有关更多信息，请参阅“RMAN 过程的常见参数”。

此过程还使用以下其他参数。

参数名称	数据类型	有效值	默认值	必需	描述
`p_from_scn`	number	磁盘上存在的存档重做日志的 SCN。该值必须小于或等于为 `p_to_scn` 指定的值。	—	是	存档日志备份的起始 SCN。
`p_to_scn`	number	磁盘上存在的存档重做日志的 SCN。该值必须大于或等于为 `p_from_scn` 指定的值。	—	是	存档日志备份的结束 SCN。

以下示例备份数据库实例的 SCN 范围中的存档重做日志。

```
BEGIN
    rdsadmin.rdsadmin_rman_util.backup_archivelog_scn(
        p_owner               => 'SYS',
        p_directory_name      => 'MYDIRECTORY',
        p_from_scn            => 1533835,
```

```
        p_to_scn              => 1892447,
        p_parallel            => 4,
        p_tag                 => 'MY_LOG_BACKUP',
        p_rman_to_dbms_output => FALSE);
END;
/
```

备份序列号范围中的存档重做日志

要指定序列号范围以备份 Amazon RDS Oracle 数据库实例的特定存档重做日志，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_rman_util.backup_archivelog_sequence`。序列号范围指定要备份哪些存档重做日志。

该过程使用 RMAN 任务的以下常见参数：

- `p_owner`
- `p_directory_name`
- `p_label`
- `p_parallel`
- `p_compress`
- `p_rman_to_dbms_output`
- `p_tag`

有关更多信息，请参阅“RMAN 过程的常见参数”。

此过程还使用以下其他参数。

参数名称	数据类型	有效值	默认值	必需	描述
`p_from_sequence`	number	磁盘上存在的存档重做日志的序列号。该值必须小于或等于为 `p_to_seq`	—	是	存档日志备份的起始序列号。

参数名称	数据类型	有效值	默认值	必需	描述
		ence 指定的值。			
p_to_sequence	number	磁盘上存在的存档重做日志的序列号。该值必须大于或等于为 p_from_sequence 指定的值。	—	是	存档日志备份的结束序列号。

以下示例备份数据库实例的序列号范围中的存档重做日志。

```
BEGIN
    rdsadmin.rdsadmin_rman_util.backup_archivelog_sequence(
        p_owner               => 'SYS',
        p_directory_name      => 'MYDIRECTORY',
        p_from_sequence       => 11160,
        p_to_sequence         => 11160,
        p_parallel            => 4,
        p_tag                 => 'MY_LOG_BACKUP',
        p_rman_to_dbms_output => FALSE);
END;
/
```

执行完整数据库备份

您可以使用 Amazon RDS 过程 `rdsadmin.rdsadmin_rman_util.backup_database_full` 执行备份中包括的所有数据块文件的备份。

该过程使用 RMAN 任务的以下常见参数：

- `p_owner`

- p_directory_name
- p_label
- p_parallel
- p_section_size_mb
- p_include_archive_logs
- p_optimize
- p_compress
- p_rman_to_dbms_output
- p_tag

有关更多信息，请参阅“RMAN 过程的常见参数”。

以下 Amazon RDS for Oracle 数据库引擎版本支持此过程：

- Oracle Database 21c (21.0.0)
- Oracle Database 19c（19.0.0）

以下示例使用参数的指定值来执行数据库实例的完整备份。

```
BEGIN
    rdsadmin.rdsadmin_rman_util.backup_database_full(
        p_owner               => 'SYS',
        p_directory_name      => 'MYDIRECTORY',
        p_parallel            => 4,
        p_section_size_mb     => 10,
        p_tag                 => 'FULL_DB_BACKUP',
        p_rman_to_dbms_output => FALSE);
END;
/
```

执行租户数据库的完整备份

您可以执行在容器数据库（CDB）中包括租户数据库的所有数据块的备份。适用 Amazon RDS 过程 rdsadmin.rdsadmin_rman_util.backup_tenant_full。该过程仅适用于当前数据库备份，并使用 RMAN 任务的以下常见参数：

- p_owner

- p_directory_name
- p_label
- p_parallel
- p_section_size_mb
- p_include_archive_logs
- p_optimize
- p_compress
- p_rman_to_dbms_output
- p_tag

有关更多信息，请参阅 RMAN 过程的常见参数。

以下 RDS for Oracle 数据库引擎版本支持 rdsadmin_rman_util.backup_tenant_full 过程：

- Oracle Database 21c（21.0.0）CDB
- Oracle Database 19c（19.0.0）CDB

以下示例使用参数的指定值来执行当前租户数据库的完整备份。

```
BEGIN
    rdsadmin.rdsadmin_rman_util.backup_tenant_full(
        p_owner               => 'SYS',
        p_directory_name      => 'MYDIRECTORY',
        p_parallel            => 4,
        p_section_size_mb     => 10,
        p_tag                 => 'FULL_TENANT_DB_BACKUP',
        p_rman_to_dbms_output => FALSE);
END;
/
```

执行增量数据库备份

您可以使用 Amazon RDS 过程 rdsadmin.rdsadmin_rman_util.backup_database_incremental 执行数据库实例的增量备份。

有关增量备份的更多信息，请参阅 Oracle 文档中的增量备份。

该过程使用 RMAN 任务的以下常见参数：

- p_owner
- p_directory_name
- p_label
- p_parallel
- p_section_size_mb
- p_include_archive_logs
- p_include_controlfile
- p_optimize
- p_compress
- p_rman_to_dbms_output
- p_tag

有关更多信息，请参阅“RMAN 过程的常见参数”。

以下 Amazon RDS for Oracle 数据库引擎版本支持此过程：

- Oracle Database 21c (21.0.0)
- Oracle Database 19c（19.0.0）

此过程还使用以下其他参数。

参数名称	数据类型	有效值	默认值	必需	描述
p_level	number	0, 1	0	否	指定 0 以启用完整增量备份。 指定 1 以启用非累积增量备份。

以下示例使用参数的指定值来执行数据库实例的增量备份。

```
BEGIN
```

```
    rdsadmin.rdsadmin_rman_util.backup_database_incremental(
        p_owner               => 'SYS',
        p_directory_name      => 'MYDIRECTORY',
        p_level               => 1,
        p_parallel            => 4,
        p_section_size_mb     => 10,
        p_tag                 => 'MY_INCREMENTAL_BACKUP',
        p_rman_to_dbms_output => FALSE);
END;
/
```

执行租户数据库的增量备份

您可以执行 CDB 中当前租户数据库的增量备份。适用 Amazon RDS 过程 `rdsadmin.rdsadmin_rman_util.backup_tenant_incremental`。

有关增量备份的更多信息，请参阅 Oracle 数据库文档中的增量备份。

该过程仅适用于当前租户数据库，并使用 RMAN 任务的以下常见参数：

- `p_owner`
- `p_directory_name`
- `p_label`
- `p_parallel`
- `p_section_size_mb`
- `p_include_archive_logs`
- `p_include_controlfile`
- `p_optimize`
- `p_compress`
- `p_rman_to_dbms_output`
- `p_tag`

有关更多信息，请参阅 RMAN 过程的常见参数。

以下 Amazon RDS for Oracle 数据库引擎版本支持此过程：

- Oracle Database 21c（21.0.0）CDB

- Oracle Database 19c（19.0.0）CDB

此过程还使用以下其他参数。

参数名称	数据类型	有效值	默认值	必需	描述
`p_level`	number	`0`, `1`	`0`	否	指定 `0` 以启用完整增量备份。 指定 `1` 以启用非累积增量备份。

以下示例使用参数的指定值来执行当前租户数据库的增量备份。

```
BEGIN
    rdsadmin.rdsadmin_rman_util.backup_tenant_incremental(
        p_owner               => 'SYS',
        p_directory_name      => 'MYDIRECTORY',
        p_level               => 1,
        p_parallel            => 4,
        p_section_size_mb     => 10,
        p_tag                 => 'MY_INCREMENTAL_BACKUP',
        p_rman_to_dbms_output => FALSE);
END;
/
```

备份表空间

您可以使用 Amazon RDS 过程 `rdsadmin.rdsadmin_rman_util.backup_tablespace` 备份表空间。

该过程使用 RMAN 任务的以下常见参数：

- `p_owner`
- `p_directory_name`
- `p_label`
- `p_parallel`
- `p_section_size_mb`

- p_include_archive_logs
- p_include_controlfile
- p_optimize
- p_compress
- p_rman_to_dbms_output
- p_tag

有关更多信息，请参阅“RMAN 过程的常见参数”。

此过程还使用以下其他参数。

参数名称	数据类型	有效值	默认值	必需	描述
p_tablespace_name	varchar2	有效的空间表名称。	—	是	要备份的表空间的名称。

以下 Amazon RDS for Oracle 数据库引擎版本支持此过程：

- Oracle Database 21c (21.0.0)
- Oracle Database 19c（19.0.0）

以下示例使用参数的指定值来执行表空间备份。

```
BEGIN
    rdsadmin.rdsadmin_rman_util.backup_tablespace(
        p_owner               => 'SYS',
        p_directory_name      => 'MYDIRECTORY',
        p_tablespace_name     => 'MYTABLESPACE',
        p_parallel            => 4,
        p_section_size_mb     => 10,
        p_tag                 => 'MYTABLESPACE_BACKUP',
        p_rman_to_dbms_output => FALSE);
END;
/
```

备份控制文件

您可以使用 Amazon RDS 过程 `rdsadmin.rdsadmin_rman_util.backup_current_controlfile` 备份控制文件。

该过程使用 RMAN 任务的以下常见参数：

- `p_owner`
- `p_directory_name`
- `p_label`
- `p_compress`
- `p_rman_to_dbms_output`
- `p_tag`

有关更多信息，请参阅“RMAN 过程的常见参数”。

以下 Amazon RDS for Oracle 数据库引擎版本支持此过程：

- Oracle Database 21c (21.0.0)
- Oracle Database 19c（19.0.0）

以下示例使用参数的指定值来备份控制文件。

```
BEGIN
    rdsadmin.rdsadmin_rman_util.backup_current_controlfile(
        p_owner               => 'SYS',
        p_directory_name      => 'MYDIRECTORY',
        p_tag                 => 'CONTROL_FILE_BACKUP',
        p_rman_to_dbms_output => FALSE);
END;
/
```

执行块介质恢复

您可以使用 Amazon RDS 过程 `rdsadmin.rdsadmin_rman_util.recover_datafile_block` 恢复各个数据块，即块介质恢复。您可以使用此重载过程来恢复单个数据块或一系列数据块。

该过程使用 RMAN 任务的以下常见参数：

- p_rman_to_dbms_output

有关更多信息，请参阅 RMAN 过程的常见参数。

此过程使用以下其它参数。

参数名称	数据类型	有效值	默认值	必需	描述
p_datafile	NUMBER	有效的数据文件 ID 号。	—	是	包含损坏块的数据文件。通过以下任一方式指定数据文件： • 数据文件 ID 号，位于 V$DATAFILE.FILE# • 完整数据文件名（包括路径），位于 V$DATAFILE.NAME
p_block	NUMBER	有效的整数。	—	是	要恢复的单个块的数量。 以下参数是互斥的： • p_block • p_from_block 和 p_to_block
p_from_block	NUMBER	有效的整数。	—	是	要恢复的一系列块中的第一个块编号。 以下参数是互斥的： • p_block • p_from_block 和 p_to_block
p_to_block	NUMBER	有效的整数。	—	是	要恢复的一系列块中的最后一个块编号。 以下参数是互斥的：

参数名称	数据类型	有效值	默认值	必需	描述
					• p_block • p_from_block 和 p_to_block

以下 Amazon RDS for Oracle 数据库引擎版本支持此过程：

- Oracle Database 21c (21.0.0)
- Oracle Database 19c（19.0.0）

以下示例恢复数据文件 5 中的块 100。

```
BEGIN
    rdsadmin.rdsadmin_rman_util.recover_datafile_block(
        p_datafile            => 5,
        p_block               => 100,
        p_rman_to_dbms_output => TRUE);
END;
/
```

以下示例将数据文件 5 中的块 100 恢复为 150。

```
BEGIN
    rdsadmin.rdsadmin_rman_util.recover_datafile_block(
        p_datafile            => 5,
        p_from_block          => 100,
        p_to_block            => 150,
        p_rman_to_dbms_output => TRUE);
END;
/
```

执行 Oracle 数据库实例的常见计划任务

某些 SYS 拥有的计划程序作业可能会影响正常的数据库操作。在这种情况下，Oracle 支持人员建议您修改计划。如果您需要启用或禁用 SYS 作业，请先在测试环境中测试对计划作业的操作，然后再在生产环境中实施它。要执行 SYS 拥有的 Oracle 计划程序作业的任务，请使用 Amazon RDS 包 `rdsadmin.rdsadmin_dbms_scheduler`。

下表所示的 Amazon RDS for Oracle 数据库引擎版本支持 `rdsadmin.rdsadmin_dbms_scheduler` 过程。使用此软件包时，您可以指定表中列出的 SYS 作业。

数据库版本	默认启用的作业	默认禁用的作业
Oracle Database 19c	BSLN_MAINTAIN_STATS_JOB CLEANUP_NON_EXIST_OBJ CLEANUP_ONLINE_IND_BUILD CLEANUP_ONLINE_PMO CLEANUP_TAB_IOT_PMO CLEANUP_TRANSIENT_PKG CLEANUP_TRANSIENT_TYPE DRA_REEVALUATE_OPEN_FAILURES FILE_SIZE_UPD ORA$AUTOTASK_CLEAN PMO_DEFERRED_GIDX_MAINT_JOB PURGE_LOG RSE$CLEAN_RECOVERABLE_SCRIPT SM$CLEAN_AUTO_SPLIT_MERGE	FGR$AUTOPURGE_JOB FILE_WATCHER HM_CREATE_OFFLINE_DICTIONARY LOAD_OPATCH_INVENTORY ORA$PREPLUGIN_BACKUP_JOB XMLDB_NFS_CLEANUP_JOB
Oracle Database 21c	BSLN_MAINTAIN_STATS_JOB CLEANUP_NON_EXIST_OBJ CLEANUP_ONLINE_IND_BUILD CLEANUP_ONLINE_PMO CLEANUP_TAB_IOT_PMO CLEANUP_TRANSIENT_PKG CLEANUP_TRANSIENT_TYPE DRA_REEVALUATE_OPEN_FAILURES FILE_SIZE_UPD ORA$AUTOTASK_CLEAN PMO_DEFERRED_GIDX_MAINT_JOB PURGE_LOG	FGR$AUTOPURGE_JOB FILE_WATCHER HM_CREATE_OFFLINE_DICTIONARY LOAD_OPATCH_INVENTORY ORA$PREPLUGIN_BACKUP_JOB ORA$_ATSK_AUTOSTS XMLDB_NFS_CLEANUP_JOB

Oracle 计划程序过程的常见参数

要使用 Oracle 计划程序执行任务，请使用 Amazon RDS 包 `rdsadmin.rdsadmin_dbms_scheduler` 中的过程。有几个参数是包中的过程通用的。该包具有以下常见参数。

参数名称	数据类型	有效值	默认值	必需	描述
`name`	varchar2	执行 Oracle 数据库实例的常见计划任务 的表中列出的步骤	—	是	要修改的作业的名称。
`attribute`	varchar2	`'REPEAT_INTERVAL'` `_NAME'`	–	是	要修改的属性。 要修改作业的重复间隔，请指定 `'REPEAT_INTERVAL'` 。 要修改作业的计划名称，请指定 `'SCHEDULE_NAME'` 。
`value`	varchar2	有效的计划间隔或计划名称，具体取决于使用的属性。	–	是	新的属性值。

修改 DBMS_SCHEDULER 作业

要修改 Oracle Scheduler 的某些组件，使用 Oracle 过程 dbms_scheduler.set_attribute。有关更多信息，请参阅 Oracle 文档中的 DBMS_SCHEDULER 和 SET_ATTRIBUTE 过程。

在使用 Amazon RDS 数据库实例时，在对象名称前面加上架构名称 SYS。以下示例设置星期一窗口对象的资源计划属性。

```
BEGIN
    DBMS_SCHEDULER.SET_ATTRIBUTE(
        name      => 'SYS.MONDAY_WINDOW',
        attribute => 'RESOURCE_PLAN',
        value     => 'resource_plan_1');
END;
/
```

修改自动任务维护时段

Amazon RDS for Oracle 实例使用维护时段的默认设置创建。自动维护任务（如优化程序统计信息收集）在这些窗口中运行。默认情况下，维护时段会打开 Oracle 数据库资源管理器。

要修改时段，请使用 DBMS_SCHEDULER 程序包。出于以下原因，您可能需要修改维护时段设置：

- 您希望维护作业在不同的时间运行，具有不同的设置，或根本不运行。例如，可能需要修改时段持续时间，或更改重复时间和间隔。
- 您希望避免在维护期间启用资源管理器造成性能影响。例如，如果指定了默认维护计划，并且在数据库处于加载状态时打开维护时段，则可能会看到等待事件，例如 resmgr:cpu quantum。此等待事件与数据库资源管理器相关。您有以下选项：
  - 确保您的数据库实例的非高峰时段处于活动状态。
  - 禁用默认维护计划，方法是将 resource_plan 属性设置为空字符串。
  - 在参数组中将 resource_manager_plan 参数设置为 FORCE:。如果您的实例使用企业版，则此设置将阻止激活数据库资源管理器计划。

修改维护时段设置

1. 使用 Oracle SQL 客户端 Connect 到数据库。
2. 查询调度程序时段的当前配置。

   以下示例查询 MONDAY_WINDOW 的配置。

```
SELECT ENABLED, RESOURCE_PLAN, DURATION, REPEAT_INTERVAL
FROM   DBA_SCHEDULER_WINDOWS
WHERE  WINDOW_NAME='MONDAY_WINDOW';
```

以下输出显示该时段正在使用默认值。

```
ENABLED         RESOURCE_PLAN                  DURATION         REPEAT_INTERVAL
--------------- ------------------------------ ----------------
 ------------------------------
TRUE            DEFAULT_MAINTENANCE_PLAN       +000 04:00:00
 freq=daily;byday=MON;byhour=22
                                                                ;byminute=0;
 bysecond=0
```

3. 使用 DBMS_SCHEDULER 程序包修改时段。

   以下示例将资源计划设置为 null，以便资源管理器不会在维护时段期间运行。

```
BEGIN
  -- disable the window to make changes
  DBMS_SCHEDULER.DISABLE(name=>'"SYS"."MONDAY_WINDOW"',force=>TRUE);

  -- specify the empty string to use no plan
  DBMS_SCHEDULER.SET_ATTRIBUTE(name=>'"SYS"."MONDAY_WINDOW"',
 attribute=>'RESOURCE_PLAN', value=>'');

  -- re-enable the window
  DBMS_SCHEDULER.ENABLE(name=>'"SYS"."MONDAY_WINDOW"');
END;
/
```

   以下示例将时段的最大持续时段设置为 2 小时。

```
BEGIN
  DBMS_SCHEDULER.DISABLE(name=>'"SYS"."MONDAY_WINDOW"',force=>TRUE);
  DBMS_SCHEDULER.SET_ATTRIBUTE(name=>'"SYS"."MONDAY_WINDOW"',
 attribute=>'DURATION', value=>'0 2:00:00');
  DBMS_SCHEDULER.ENABLE(name=>'"SYS"."MONDAY_WINDOW"');
END;
/
```

以下示例将重复间隔设置为每个星期一上午 10 点。

```
BEGIN
  DBMS_SCHEDULER.DISABLE(name=>'"SYS"."MONDAY_WINDOW"',force=>TRUE);
  DBMS_SCHEDULER.SET_ATTRIBUTE(name=>'"SYS"."MONDAY_WINDOW"',
 attribute=>'REPEAT_INTERVAL',
 value=>'freq=daily;byday=MON;byhour=10;byminute=0;bysecond=0');
  DBMS_SCHEDULER.ENABLE(name=>'"SYS"."MONDAY_WINDOW"');
END;
/
```

设置 Oracle Scheduler 作业的时区

要修改 Oracle Scheduler 的时区，您可以使用 Oracle 过程 `dbms_scheduler.set_scheduler_attribute`。有关 `dbms_scheduler` 程序包的更多信息，请参阅 Oracle 文档中的 DBMS_SCHEDULER 和 SET_SCHEDULER_ATTRIBUTE。

修改当前时区设置

1. 使用客户端（例如 SQL Developer）连接到数据库。有关更多信息，请参阅“使用 Oracle SQL Developer 连接到数据库实例”。
2. 按如下所示设置默认时区，并将时区替换为 *time_zone_name*。

```
BEGIN
  DBMS_SCHEDULER.SET_SCHEDULER_ATTRIBUTE(
    attribute => 'default_timezone',
    value => 'time_zone_name'
  );
END;
/
```

在以下示例中，您将时区更改为亚洲/上海。

首先查询当前时区，如下所示。

```
SELECT VALUE FROM DBA_SCHEDULER_GLOBAL_ATTRIBUTE WHERE
 ATTRIBUTE_NAME='DEFAULT_TIMEZONE';
```

输出显示当前时区为 ETC/UTC。

```
VALUE
-------
Etc/UTC
```

然后，将时区设置为亚洲/上海。

```
BEGIN
  DBMS_SCHEDULER.SET_SCHEDULER_ATTRIBUTE(
    attribute => 'default_timezone',
    value => 'Asia/Shanghai'
  );
END;
/
```

有关更改系统时区的更多信息，请参阅 Oracle 时区。

关闭 SYS 拥有的 Oracle 计划程序任务

要禁用 SYS 用户拥有的 Oracle 计划程序任务，请使用 `rdsadmin.rdsadmin_dbms_scheduler.disable` 过程。

该过程在 Oracle 计划程序任务中使用 name 常见参数。有关更多信息，请参阅“Oracle 计划程序过程的常见参数”。

以下示例禁用 `SYS.CLEANUP_ONLINE_IND_BUILD` Oracle 计划程序作业。

```
BEGIN
   rdsadmin.rdsadmin_dbms_scheduler.disable('SYS.CLEANUP_ONLINE_IND_BUILD');
END;
/
```

开启 SYS 拥有的 Oracle 计划程序任务

要开启 SYS 拥有的 Oracle 计划程序任务，请使用 `rdsadmin.rdsadmin_dbms_scheduler.enable` 过程。

该过程在 Oracle 计划程序任务中使用 name 常见参数。有关更多信息，请参阅“Oracle 计划程序过程的常见参数”。

以下示例启用 SYS.CLEANUP_ONLINE_IND_BUILD Oracle 计划程序作业。

```
BEGIN
   rdsadmin.rdsadmin_dbms_scheduler.enable('SYS.CLEANUP_ONLINE_IND_BUILD');
END;
/
```

修改 CALENDAR 类型的任务的 Oracle 计划程序重复间隔

要修改重复间隔以修改 SYS 拥有的 CALENDAR 类型的 Oracle 计划程序作业，请使用 rdsadmin.rdsadmin_dbms_scheduler.disable 过程。

该过程在 Oracle 计划程序任务中使用以下常见参数：

- name
- attribute
- value

有关更多信息，请参阅"Oracle 计划程序过程的常见参数"。

以下示例修改 SYS.CLEANUP_ONLINE_IND_BUILD Oracle 计划程序作业的重复间隔。

```
BEGIN
     rdsadmin.rdsadmin_dbms_scheduler.set_attribute(
          name      => 'SYS.CLEANUP_ONLINE_IND_BUILD',
          attribute => 'repeat_interval',
          value     => 'freq=daily;byday=FRI,SAT;byhour=20;byminute=0;bysecond=0');
END;
/
```

修改 NAMED 类型的任务的 Oracle 计划程序重复间隔

某些 Oracle 计划程序作业使用计划名称而不是间隔。对于此类型的作业，您必须在主用户架构中创建新的命名计划。请使用标准 Oracle sys.dbms_scheduler.create_schedule 过程执行该操作。此外，还要使用 rdsadmin.rdsadmin_dbms_scheduler.set_attribute procedure 将新的命名计划分配给作业。

该过程在 Oracle 计划程序任务中使用以下常见参数：

- name

- `attribute`
- `value`

有关更多信息，请参阅“Oracle 计划程序过程的常见参数”。

以下示例修改 `SYS.BSLN_MAINTAIN_STATS_JOB` Oracle 计划程序作业的重复间隔。

```
BEGIN
     DBMS_SCHEDULER.CREATE_SCHEDULE (
          schedule_name   => 'rds_master_user.new_schedule',
          start_date      => SYSTIMESTAMP,
          repeat_interval =>
 'freq=daily;byday=MON,TUE,WED,THU,FRI;byhour=0;byminute=0;bysecond=0',
          end_date        => NULL,
          comments        => 'Repeats daily forever');
END;
/

BEGIN
     rdsadmin.rdsadmin_dbms_scheduler.set_attribute (
          name      => 'SYS.BSLN_MAINTAIN_STATS_JOB',
          attribute => 'schedule_name',
          value     => 'rds_master_user.new_schedule');
END;
/
```

在创建 Oracle 计划程序任务时关闭自动提交

当 `DBMS_SCHEDULER.CREATE_JOB` 创建 Oracle 计划程序任务时，它会立即创建任务并提交更改。要执行以下操作，可能需要将 Oracle 计划程序任务的创建合并到用户事务中：

- 回滚用户事务时，回滚 Oracle 计划程序任务。
- 提交主用户事务时，创建 Oracle 计划程序任务。

可以使用过程 `rdsadmin.rdsadmin_dbms_scheduler.set_no_commit_flag` 开启此行为。此过程不需要任何参数。您可以在以下 RDS for Oracle 版本中使用此过程：

- 21.0.0.0.ru-2022-07.rur-2022-07.r1 及更高版本
- 19.0.0.0.ru-2022-07.rur-2022-07.r1 及更高版本

以下示例关闭 Oracle 计划程序的自动提交，创建 Oracle 计划程序任务，然后回滚事务。由于自动提交已关闭，因此数据库还会回滚 Oracle 计划程序任务的创建。

```
BEGIN
  rdsadmin.rdsadmin_dbms_scheduler.set_no_commit_flag;
  DBMS_SCHEDULER.CREATE_JOB(job_name   => 'EMPTY_JOB',
                            job_type   => 'PLSQL_BLOCK',
                            job_action => 'begin null; end;',
                            auto_drop  => false);
  ROLLBACK;
END;
/

PL/SQL procedure successfully completed.

SELECT * FROM DBA_SCHEDULER_JOBS WHERE JOB_NAME='EMPTY_JOB';

no rows selected
```

诊断 RDS for Oracle 数据库实例的问题

Oracle 数据库包括一个故障诊断基础设施，可用于调查数据库问题。在 Oracle 术语中，问题 是严重错误，如代码错误或数据损坏。事件 是指问题的发生。如果同一错误发生三次，则基础设施会显示此问题的三个事件。有关更多信息，请参阅 Oracle 数据库文档中的诊断和解决问题 。

自动诊断存储库命令解释器 (ADRCI) 实用程序是用于管理诊断数据的 Oracle 命令行工具。例如，您可以使用此工具调查问题并打包诊断数据。事件包 包括引用特定问题的一个或所有事件的诊断数据。您可以将事件包（作为 .zip 文件实现）上传到 Oracle 支持部门。

为了提供托管服务体验，Amazon RDS 未向 Shell 提供对于 ADRCI 的访问权限。要为 RDS for Oracle 实例执行诊断任务，请使用 Amazon RDS 程序包 `rdsadmin.rdsadmin_adrci_util`。

通过使用 `rdsadmin_adrci_util` 中的函数，您可以列出和打包问题和事件，还可以显示跟踪文件。所有函数都返回一个任务 ID。此 ID 构成包含 ADRCI 输出的日志文件名称的一部分，如 `dbtask-`*`task_id`*`.log` 中所示。日志文件位于 BDUMP 目录中。您可以按照下载数据库日志文件中描述的过程下载日志文件。

诊断过程的常见参数

要执行诊断任务，请使用 Amazon RDS 程序包 `rdsadmin.rdsadmin_adrci_util` 中的函数。该包具有以下常见参数。

参数名称	数据类型	有效值	默认值	必需	描述
`incident_id`	number	有效的事件 ID 或 Null	Null	否	如果值为 Null，函数将显示所有事件。如果值不为 Null 并且表示有效的事件 ID，则函数将显示指定的事件。
`problem_id`	number	有效的问题 ID 或 Null	Null	否	如果值为 Null，则函数显示所有问题。如果该值不为 Null 并表示有效的问题 ID，则函数将显示指定的问题。
`last`	number	大于 0 或 Null 的有效整数	Null	否	如果值为 Null，则函数最多显示 50 个项目。如果值不为 Null，则函数显示指定的数字。

列出事件

要列出 Oracle 的诊断事件，请使用 Amazon RDS 函数 `rdsadmin.rdsadmin_adrci_util.list_adrci_incidents`。您可以在基本模式或详细模式下列出事件。默认情况下，函数会列出 50 个最近的事件。

此函数使用以下常见参数：

- `incident_id`
- `problem_id`
- `last`

如果您指定 `incident_id` 和 `problem_id`，则 `incident_id` 覆盖 `problem_id`。有关更多信息，请参阅 诊断过程的常见参数。

此函数使用以下附加参数。

参数名称	数据类型	有效值	默认值	必需	描述
detail	布尔值	TRUE 或者 FALSE	FALSE	否	如果为 TRUE，函数将以详细模式列出事件。如果为 FALSE，函数将以基本模式列出事件。

要列出所有事件，请不带任何参数查询 `rdsadmin.rdsadmin_adrci_util.list_adrci_incidents` 函数。查询将返回任务 ID。

```
SQL> SELECT rdsadmin.rdsadmin_adrci_util.list_adrci_incidents AS task_id FROM DUAL;

TASK_ID
------------------
1590786706158-3126
```

或者，不带任何参数调用 `rdsadmin.rdsadmin_adrci_util.list_adrci_incidents` 函数，并将输出存储在 SQL 客户端变量中。您可以在其他语句中使用该变量。

```
SQL> VAR task_id VARCHAR2(80);
SQL> EXEC :task_id := rdsadmin.rdsadmin_adrci_util.list_adrci_incidents;

PL/SQL procedure successfully completed.
```

要读取日志文件，请调用 Amazon RDS 过程 `rdsadmin.rds_file_util.read_text_file`。提供任务 ID 作为文件名的一部分。下面的输出显示三个事件：53523、53522 和 53521。

```
SQL> SELECT * FROM TABLE(rdsadmin.rds_file_util.read_text_file('BDUMP',
 'dbtask-'||:task_id||'.log'));

TEXT
-------------------------------------------------------------------------------------------------------
2020-05-29 21:11:46.193 UTC [INFO ] Listing ADRCI incidents.
2020-05-29 21:11:46.256 UTC [INFO ]
ADR Home = /rdsdbdata/log/diag/rdbms/orcl_a/ORCL:
*************************************************************************
```

```
INCIDENT_ID PROBLEM_KEY                                                     CREATE_TIME
----------- -----------------------------------------------------------
 ----------------------------------------
53523       ORA 700 [EVENT_CREATED_INCIDENT] [942] [SIMULATED_ERROR_003 2020-05-29
 20:15:20.928000 +00:00
53522       ORA 700 [EVENT_CREATED_INCIDENT] [942] [SIMULATED_ERROR_002 2020-05-29
 20:15:15.247000 +00:00
53521       ORA 700 [EVENT_CREATED_INCIDENT] [942] [SIMULATED_ERROR_001 2020-05-29
 20:15:06.047000 +00:00
3 rows fetched


2020-05-29 21:11:46.256 UTC [INFO ] The ADRCI incidents were successfully listed.
2020-05-29 21:11:46.256 UTC [INFO ] The task finished successfully.

14 rows selected.
```

要列出特定事件，请使用 incident_id 参数指定事件 ID。在以下示例中，您只查询日志文件以搜索事件 53523。

```
SQL> EXEC :task_id :=
 rdsadmin.rdsadmin_adrci_util.list_adrci_incidents(incident_id=>53523);

PL/SQL procedure successfully completed.

SQL> SELECT * FROM TABLE(rdsadmin.rds_file_util.read_text_file('BDUMP',
 'dbtask-'||:task_id||'.log'));

TEXT
------------------------------------------------------------------------------------------------------------------------
2020-05-29 21:15:25.358 UTC [INFO ] Listing ADRCI incidents.
2020-05-29 21:15:25.426 UTC [INFO ]
ADR Home = /rdsdbdata/log/diag/rdbms/orcl_a/ORCL:
*************************************************************************
INCIDENT_ID          PROBLEM_KEY
 CREATE_TIME
-------------------- -----------------------------------------------------------
 ---------------------------------
53523                ORA 700 [EVENT_CREATED_INCIDENT] [942] [SIMULATED_ERROR_003
 2020-05-29 20:15:20.928000 +00:00
1 rows fetched
```

```
2020-05-29 21:15:25.427 UTC [INFO ] The ADRCI incidents were successfully listed.
2020-05-29 21:15:25.427 UTC [INFO ] The task finished successfully.

12 rows selected.
```

列出问题

要列出 Oracle 的诊断问题，请使用 Amazon RDS 函数 `rdsadmin.rdsadmin_adrci_util.list_adrci_problems`。

默认情况下，函数列出 50 个最近的问题。

此函数使用公用参数 `problem_id` 和 `last`。有关更多信息，请参阅 诊断过程的常见参数。

要获取所有问题的任务 ID，请不带任何参数调用 `rdsadmin.rdsadmin_adrci_util.list_adrci_problems` 函数，并将输出存储在 SQL 客户端变量中。

```
SQL> EXEC :task_id := rdsadmin.rdsadmin_adrci_util.list_adrci_problems;

PL/SQL procedure successfully completed.
```

要读取日志文件，请调用 `rdsadmin.rds_file_util.read_text_file` 函数，同时提供任务 ID 作为文件名的一部分。在以下输出中，日志文件显示三个问题：1、2 和 3。

```
SQL> SELECT * FROM TABLE(rdsadmin.rds_file_util.read_text_file('BDUMP',
 'dbtask-'||:task_id||'.log'));

TEXT
-------------------------------------------------------------------------------------------------
2020-05-29 21:18:50.764 UTC [INFO ] Listing ADRCI problems.
2020-05-29 21:18:50.829 UTC [INFO ]
ADR Home = /rdsdbdata/log/diag/rdbms/orcl_a/ORCL:
*************************************************************************
PROBLEM_ID   PROBLEM_KEY                                                 LAST_INCIDENT
       LASTINC_TIME
---------- ----------------------------------------------------------- -------------
 ---------------------------------
2          ORA 700 [EVENT_CREATED_INCIDENT] [942] [SIMULATED_ERROR_003 53523
 2020-05-29 20:15:20.928000 +00:00
3          ORA 700 [EVENT_CREATED_INCIDENT] [942] [SIMULATED_ERROR_002 53522
 2020-05-29 20:15:15.247000 +00:00
```

```
1          ORA 700 [EVENT_CREATED_INCIDENT] [942] [SIMULATED_ERROR_001 53521
 2020-05-29 20:15:06.047000 +00:00
3 rows fetched


2020-05-29 21:18:50.829 UTC [INFO ] The ADRCI problems were successfully listed.
2020-05-29 21:18:50.829 UTC [INFO ] The task finished successfully.

14 rows selected.
```

在下面的示例中，您只列出问题 3。

```
SQL> EXEC :task_id := rdsadmin.rdsadmin_adrci_util.list_adrci_problems(problem_id=>3);

PL/SQL procedure successfully completed.
```

要读取问题 3 的日志文件，请调用 `rdsadmin.rds_file_util.read_text_file`。提供任务 ID 作为文件名的一部分。

```
SQL> SELECT * FROM TABLE(rdsadmin.rds_file_util.read_text_file('BDUMP',
 'dbtask-'||:task_id||'.log'));

TEXT
------------------------------------------------------------------------
2020-05-29 21:19:42.533 UTC [INFO ] Listing ADRCI problems.
2020-05-29 21:19:42.599 UTC [INFO ]
ADR Home = /rdsdbdata/log/diag/rdbms/orcl_a/ORCL:
*************************************************************************
PROBLEM_ID PROBLEM_KEY                                                   LAST_INCIDENT
 LASTINC_TIME
---------- ----------------------------------------------------------- -------------
 --------------------------------
3          ORA 700 [EVENT_CREATED_INCIDENT] [942] [SIMULATED_ERROR_002 53522
 2020-05-29 20:15:15.247000 +00:00
1 rows fetched


2020-05-29 21:19:42.599 UTC [INFO ] The ADRCI problems were successfully listed.
2020-05-29 21:19:42.599 UTC [INFO ] The task finished successfully.

12 rows selected.
```

创建事件包

您可以使用 Amazon RDS 函数创建事件包 `rdsadmin.rdsadmin_adrci_util.create_adrci_package`。输出是一个 .zip 文件，您可以将其提供给 Oracle 支持部门。

此函数使用以下常见参数：

- `problem_id`
- `incident_id`

确保指定前面的参数之一。如果指定了两个参数，则 `incident_id` 覆盖 `problem_id`。有关更多信息，请参阅"诊断过程的常见参数"。

要为特定事件创建包，请使用 `rdsadmin.rdsadmin_adrci_util.create_adrci_package` 参数调用 Amazon RDS 函数 `incident_id`。以下示例为事件 53523 创建一个包。

```
SQL> EXEC :task_id :=
 rdsadmin.rdsadmin_adrci_util.create_adrci_package(incident_id=>53523);

PL/SQL procedure successfully completed.
```

要读取日志文件，请调用 `rdsadmin.rds_file_util.read_text_file`。您可以将任务 ID 作为文件名的一部分提供。输出显示您生成了事件包 `ORA700EVE_20200529212043_COM_1.zip`。

```
SQL> SELECT * FROM TABLE(rdsadmin.rds_file_util.read_text_file('BDUMP',
 'dbtask-'||:task_id||'.log'));

TEXT
--------------------------------------------------------------------------------------------------------
2020-05-29 21:20:43.031 UTC [INFO ] The ADRCI package is being created.
2020-05-29 21:20:47.641 UTC [INFO ] Generated package 1 in file /rdsdbdata/log/trace/
ORA700EVE_20200529212043_COM_1.zip, mode complete
2020-05-29 21:20:47.642 UTC [INFO ] The ADRCI package was successfully created.
2020-05-29 21:20:47.642 UTC [INFO ] The task finished successfully.
```

要打包特定问题的诊断数据，请使用 `problem_id` 参数指定其 ID。在以下示例中，您仅打包问题 3 的数据。

```
SQL> EXEC :task_id := rdsadmin.rdsadmin_adrci_util.create_adrci_package(problem_id=>3);
```

```
PL/SQL procedure successfully completed.
```

要读取任务输出，请调用 rdsadmin.rds_file_util.read_text_file，同时提供任务 ID 作为文件名的一部分。输出显示您生成了事件包 ORA700EVE_20200529212111_COM_1.zip。

```
SQL> SELECT * FROM TABLE(rdsadmin.rds_file_util.read_text_file('BDUMP',
 'dbtask-'||:task_id||'.log'));

TEXT
---------------------------------------------------------------------------------------------------
2020-05-29 21:21:11.050 UTC [INFO ] The ADRCI package is being created.
2020-05-29 21:21:15.646 UTC [INFO ] Generated package 2 in file /rdsdbdata/log/trace/
ORA700EVE_20200529212111_COM_1.zip, mode complete
2020-05-29 21:21:15.646 UTC [INFO ] The ADRCI package was successfully created.
2020-05-29 21:21:15.646 UTC [INFO ] The task finished successfully.
```

还可以下载日志文件。有关更多信息，请参阅 下载数据库日志文件。

显示跟踪文件

您可以使用 Amazon RDS 函数 rdsadmin.rdsadmin_adrci_util.show_adrci_tracefile 列出跟踪目录下的跟踪文件以及当前 ADR 主目录下的所有事件目录。还可以显示跟踪文件和事件跟踪文件的内容。

此函数使用以下参数。

参数名称	数据类型	有效值	默认值	必需	描述
filename	varchar2	有效的跟踪文件名	Null	否	如果值为 Null，函数将显示所有跟踪文件。如果不为 Null，则函数显示指定的文件。

要显示跟踪文件，请调用 Amazon RDS 函数 rdsadmin.rdsadmin_adrci_util.show_adrci_tracefile。

```
SQL> EXEC :task_id := rdsadmin.rdsadmin_adrci_util.show_adrci_tracefile;
```

```
PL/SQL procedure successfully completed.
```

要列出跟踪文件名，请调用 Amazon RDS 过程 `rdsadmin.rds_file_util.read_text_file`，同时提供任务 ID 作为文件名的一部分。

```
SQL> SELECT * FROM TABLE(rdsadmin.rds_file_util.read_text_file('BDUMP',
 'dbtask-'||:task_id||'.log')) WHERE TEXT LIKE '%/alert_%';

TEXT
---------------------------------------------------------------
     diag/rdbms/orcl_a/ORCL/trace/alert_ORCL.log.2020-05-28
     diag/rdbms/orcl_a/ORCL/trace/alert_ORCL.log.2020-05-27
     diag/rdbms/orcl_a/ORCL/trace/alert_ORCL.log.2020-05-26
     diag/rdbms/orcl_a/ORCL/trace/alert_ORCL.log.2020-05-25
     diag/rdbms/orcl_a/ORCL/trace/alert_ORCL.log.2020-05-24
     diag/rdbms/orcl_a/ORCL/trace/alert_ORCL.log.2020-05-23
     diag/rdbms/orcl_a/ORCL/trace/alert_ORCL.log.2020-05-22
     diag/rdbms/orcl_a/ORCL/trace/alert_ORCL.log.2020-05-21
     diag/rdbms/orcl_a/ORCL/trace/alert_ORCL.log

9 rows selected.
```

在以下示例中，您将为 `alert_ORCL.log` 生成输出。

```
SQL> EXEC :task_id := rdsadmin.rdsadmin_adrci_util.show_adrci_tracefile('diag/rdbms/
orcl_a/ORCL/trace/alert_ORCL.log');

PL/SQL procedure successfully completed.
```

要读取日志文件，请调用 `rdsadmin.rds_file_util.read_text_file`。提供任务 ID 作为文件名的一部分。输出显示 alert_ORCL.log 的前 10 行。

```
SQL> SELECT * FROM TABLE(rdsadmin.rds_file_util.read_text_file('BDUMP',
 'dbtask-'||:task_id||'.log')) WHERE ROWNUM <= 10;

TEXT
-----------------------------------------------------------------------------------------
2020-05-29 21:24:02.083 UTC [INFO ] The trace files are being displayed.
2020-05-29 21:24:02.128 UTC [INFO ] Thu May 28 23:59:10 2020
Thread 1 advanced to log sequence 2048 (LGWR switch)
  Current log# 3 seq# 2048 mem# 0: /rdsdbdata/db/ORCL_A/onlinelog/o1_mf_3_hbl2p8xs_.log
Thu May 28 23:59:10 2020
```

```
Archived Log entry 2037 added for thread 1 sequence 2047 ID 0x5d62ce43 dest 1:
Fri May 29 00:04:10 2020
Thread 1 advanced to log sequence 2049 (LGWR switch)
  Current log# 4 seq# 2049 mem# 0: /rdsdbdata/db/ORCL_A/onlinelog/o1_mf_4_hbl2qgmh_.log
Fri May 29 00:04:10 2020

10 rows selected.
```

还可以下载日志文件。有关更多信息，请参阅 下载数据库日志文件。

执行 Oracle 数据库实例的其他任务

下文中介绍了如何在运行 Oracle 的 Amazon RDS 数据库实例上执行其他 DBA 任务。为了提供托管服务体验，Amazon RDS 不允许通过 shell 访问数据库实例，而仅限访问某些需要高级权限的系统过程和表。

主题

- 在主数据存储空间中创建和删除目录
- 列出数据库实例目录中的文件
- 读取数据库实例目录中的文件
- 访问 Opatch 文件
- 管理顾问任务
- 传输表空间

在主数据存储空间中创建和删除目录

要创建目录，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_util.create_directory`。您可以创建最多 10000 个目录 (全部位于主数据存储空间中)。要删除目录，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_util.drop_directory`。

`create_directory` 和 `drop_directory` 过程具有以下必需参数。

参数名称	数据类型	默认值	必需	描述
`p_directory_name`	VARCHAR2	—	是	目录的名称。

以下示例创建一个名为 `PRODUCT_DESCRIPTIONS` 的新目录。

```
EXEC rdsadmin.rdsadmin_util.create_directory(p_directory_name =>
 'product_descriptions');
```

数据字典以大写形式存储目录名称。您可以通过查询 DBA_DIRECTORIES 列出目录。系统将自动选择实际主机路径名称。以下示例获取名为 PRODUCT_DESCRIPTIONS 的目录的目录路径：

```
SELECT DIRECTORY_PATH
  FROM DBA_DIRECTORIES
 WHERE DIRECTORY_NAME='PRODUCT_DESCRIPTIONS';

DIRECTORY_PATH
----------------------------------------
/rdsdbdata/userdirs/01
```

数据库实例的主用户名在新目录中拥有读取和写入权限，可以向其他用户授予访问权限。EXECUTE 权限不可用于数据库实例上的目录。目录在您的主数据存储空间中创建，将占用空间和 I/O 带宽。

以下示例删除名为 PRODUCT_DESCRIPTIONS 的目录。

```
EXEC rdsadmin.rdsadmin_util.drop_directory(p_directory_name => 'product_descriptions');
```

> **Note**
>
> 还可以使用 Oracle SQL 命令 DROP DIRECTORY 删除目录。

删除目录不会删除其内容。由于 rdsadmin.rdsadmin_util.create_directory 过程可重用路径名称，因此，删除的目录中的文件会显示在新创建的目录中。在您删除目录之前，我们建议您使用 UTL_FILE.FREMOVE 从目录中删除文件。有关更多信息，请参阅 Oracle 文档中的 FREMOVE 过程。

列出数据库实例目录中的文件

要列出目录中的文件，请使用 Amazon RDS 过程 rdsadmin.rds_file_util.listdir。Oracle 副本不支持此过程。listdir 过程具有以下参数。

参数名称	数据类型	默认值	必需	描述
p_directory	varchar2	—	是	要列出的目录的名称。

以下示例将目录 PRODUCT_DESCRIPTIONS 的读/写权限授予用户 rdsadmin，然后列出此目录中的文件。

```
GRANT READ,WRITE ON DIRECTORY PRODUCT_DESCRIPTIONS TO rdsadmin;
SELECT * FROM TABLE(rdsadmin.rds_file_util.listdir(p_directory =>
 'PRODUCT_DESCRIPTIONS'));
```

读取数据库实例目录中的文件

要读取文本文件，请使用 Amazon RDS 过程 rdsadmin.rds_file_util.read_text_file。read_text_file 过程具有以下参数。

参数名称	数据类型	默认值	必需	描述
p_directory	varchar2	—	是	包含文件的目录的名称。
p_filename	varchar2	—	是	要读取的文件的名称。

以下示例在 rice.txt 目录中创建 PRODUCT_DESCRIPTIONS 文件。

```
declare
  fh sys.utl_file.file_type;
begin
  fh := utl_file.fopen(location=>'PRODUCT_DESCRIPTIONS', filename=>'rice.txt',
 open_mode=>'w');
  utl_file.put(file=>fh, buffer=>'AnyCompany brown rice, 15 lbs');
  utl_file.fclose(file=>fh);
end;
/
```

以下示例读取目录 rice.txt 中的文件 PRODUCT_DESCRIPTIONS。

```
SELECT * FROM TABLE
    (rdsadmin.rds_file_util.read_text_file(
        p_directory => 'PRODUCT_DESCRIPTIONS',
        p_filename  => 'rice.txt'));
```

访问 Opatch 文件

Opatch 是一个 Oracle 实用程序，可用于将修补程序应用和回滚到 Oracle 软件。用于确定哪些修补程序已应用于数据库的 Oracle 机制是 `opatch lsinventory` 命令。为了向自带许可 (BYOL) 客户打开服务请求，Oracle 支持部门会请求 `lsinventory` 文件，有时还请求由 Opatch 生成的 `lsinventory_detail` 文件。

为提供托管服务体验，Amazon RDS 未提供对 Opatch 的 Shell 访问权限。相反，`lsinventory-*dbv*.txt` 包含与当前引擎版本相关的修补程序信息。当您执行次要或主要升级时，Amazon RDS 会在应用补丁后的一小时内更新 `lsinventory-*dbv*.txt`。要验证应用的补丁，请参阅`lsinventory-*dbv*.txt`。该操作类似于运行 `opatch lsinventory` 命令。

> **Note**
>
> 本节中的示例假定 BDUMP 目录名为 BDUMP。在只读副本上，BDUMP 目录名称不同。要了解如何通过对只读副本查询 `V$DATABASE.DB_UNIQUE_NAME` 来获取 BDUMP 名称，请参阅列出文件。

清单文件使用 Amazon RDS 命名约定 `lsinventory-*dbv*.txt` 和 `lsinventory_detail-*dbv*.txt`，其中 *dbv* 是数据库版本的全名。`lsinventory-*dbv*.txt` 文件在所有数据库版本上均可用。对应的 `lsinventory_detail-*dbv*.txt` 在 19.0.0.0、ru-2020-01.rur-2020-01.r1 或更高版本上可用。

例如，如果您的数据库版本为 19.0.0.0.ru-2021-07.rur-2021-07.r1，则您的清单文件具有以下名称。

```
lsinventory-19.0.0.0.ru-2021-07.rur-2021-07.r1.txt
lsinventory_detail-19.0.0.0.ru-2021-07.rur-2021-07.r1.txt
```

确保下载与当前版本的数据库引擎匹配的文件。

控制台

使用控制台下载清单文件

1. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。
3. 选择要查看其日志文件的数据库实例的名称。

4. 选择 Logs & events (日志和事件) 选项卡。
5. 向下滚动到日志部分。
6. 在日志部分中，搜索 lsinventory。
7. 选择要访问的文件，然后选择下载。

SQL

要在 SQL 客户端中读取 lsinventory-*dbv*.txt，您可以使用 SELECT 语句。对于此技术，请使用以下任一 rdsadmin 函数：rdsadmin.rds_file_util.read_text_file 或 rdsadmin.tracefile_listing。

在下面的示例查询中，将 *dbv* 替换为您的 Oracle 数据库版本。例如，您的数据库版本可能是 19.0.0.0.ru-2020-04.rur-2020-04.r1。

```
SELECT text
FROM   TABLE(rdsadmin.rds_file_util.read_text_file('BDUMP', 'lsinventory-dbv.txt'));
```

PL/SQL

要在 SQL 客户端中读取 lsinventory-*dbv*.txt，您可以编写 PL/SQL 程序。此程序使用 utl_file 读取文件，并使用 dbms_output 打印文件。这些是 Oracle 提供的程序包。

在下面的示例程序中，将 *dbv* 替换为您的 Oracle 数据库版本。例如，您的数据库版本可能是 19.0.0.0.ru-2020-04.rur-2020-04.r1。

```
SET SERVEROUTPUT ON
DECLARE
  v_file               SYS.UTL_FILE.FILE_TYPE;
  v_line               VARCHAR2(1000);
  v_oracle_home_type   VARCHAR2(1000);
  c_directory          VARCHAR2(30) := 'BDUMP';
  c_output_file        VARCHAR2(30) := 'lsinventory-dbv.txt';
BEGIN
  v_file := SYS.UTL_FILE.FOPEN(c_directory, c_output_file, 'r');
  LOOP
    BEGIN
      SYS.UTL_FILE.GET_LINE(v_file, v_line,1000);
      DBMS_OUTPUT.PUT_LINE(v_line);
    EXCEPTION
      WHEN no_data_found THEN
```

```
        EXIT;
    END;
  END LOOP;
END;
/
```

或者，查询 `rdsadmin.tracefile_listing`，并将输出后台打印到一个文件。以下示例将输出后台打印到 `/tmp/tracefile.txt`。

```
SPOOL /tmp/tracefile.txt
SELECT *
FROM   rdsadmin.tracefile_listing
WHERE  FILENAME LIKE 'lsinventory%';
SPOOL OFF;
```

管理顾问任务

Oracle 数据库中包含许多顾问。每个顾问都支持自动化任务和手动任务。您可以使用 `rdsadmin.rdsadmin_util` 程序包中的过程来管理一些顾问任务。

以下引擎版本中提供了顾问任务过程：

- Oracle Database 21c (21.0.0)
- 版本 19.0.0.0.ru-2021-01.rur-2021-01.r1 及更高 Oracle Database 19c 版本

  有关更多信息，请参阅《Amazon RDS for Oracle 发布说明》中的[版本 19.0.0.0.ru-2021-01.rur-2021-01.r1]。

主题

- [设置顾问任务参数]
- [禁用 AUTO_STATS_ADVISOR_TASK]
- [重新启用 AUTO_STATS_ADVISOR_TASK]

设置顾问任务参数

要为某些顾问任务设置参数，请使用 Amazon RDS 程序 `rdsadmin.rdsadmin_util.advisor_task_set_parameter`。`advisor_task_set_parameter` 过程具有以下参数。

<table>
<tr><th>参数名称</th><th>数据类型</th><th>默认值</th><th>必需</th><th>描述</th></tr>
<tr><td>p_task_name</td><td>varchar2</td><td>—</td><td>是</td><td>要更改其参数的顾问任务的名称。有效值有：
AUTO_STATS_ADVISOR_TASK
INDIVIDUAL_STATS_ADVISOR_TASK
SYS_AUTO_SPM_EVOLVE_TASK
SYS_AUTO_SQL_TUNING_TASK</td></tr>
<tr><td>p_parameter</td><td>varchar2</td><td>—</td><td>是</td><td>任务参数的名称。要查找顾问任务的有效参数，请运行以下查询。将 p_task_name 替换为 p_task_name 的有效值：<pre>COL PARAMETER_NAME FORMAT a30
COL PARAMETER_VALUE FORMAT a30
SELECT PARAMETER_NAME, PARAMETER
_VALUE
FROM DBA_ADVISOR_PARAMETERS
WHERE TASK_NAME=' p_task_name '
AND PARAMETER_VALUE != 'UNUSED'
ORDER BY PARAMETER_NAME;</pre></td></tr>
<tr><td>p_value</td><td>varchar2</td><td>—</td><td>是</td><td>任务参数的值。要查找任务参数的有效值，请运行以下查询。将 p_task_name 替换为 p_task_name 的有效值：<pre>COL PARAMETER_NAME FORMAT a30
COL PARAMETER_VALUE FORMAT a30
SELECT PARAMETER_NAME, PARAMETER
_VALUE
FROM DBA_ADVISOR_PARAMETERS
WHERE TASK_NAME=' p_task_name '
AND PARAMETER_VALUE != 'UNUSED'
ORDER BY PARAMETER_NAME;</pre></td></tr>
</table>

对于 ACCEPT_PLANS，以下 PL/SQL 程序可将 FALSE 设置为 SYS_AUTO_SPM_EVOLVE_TASK。SQL Plan Management 自动化任务可验证计划并生成其调查结果的报告，但不会自动发展计划。您可以使用报告来识别新的 SQL 计划基准并手动接受它们。

```
BEGIN
  rdsadmin.rdsadmin_util.advisor_task_set_parameter(
    p_task_name => 'SYS_AUTO_SPM_EVOLVE_TASK',
    p_parameter => 'ACCEPT_PLANS',
    p_value     => 'FALSE');
END;
```

对于 EXECUTION_DAYS_TO_EXPIRE，以下 PL/SQL 程序可将 10 设置为 AUTO_STATS_ADVISOR_TASK。预定义的任务 AUTO_STATS_ADVISOR_TASK 每天在维护时段自动运行一次。该示例将任务执行的保留周期设置为 10 天。

```
BEGIN
  rdsadmin.rdsadmin_util.advisor_task_set_parameter(
    p_task_name => 'AUTO_STATS_ADVISOR_TASK',
    p_parameter => 'EXECUTION_DAYS_TO_EXPIRE',
    p_value     => '10');
END;
```

禁用 AUTO_STATS_ADVISOR_TASK

要禁用 AUTO_STATS_ADVISOR_TASK，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_util.advisor_task_drop`。`advisor_task_drop` 过程接受以下参数。

参数名称	数据类型	默认值	必需	描述
p_task_name	varchar2	—	是	要禁用的顾问任务的名称。唯一有效值为 AUTO_STATS_ADVISOR_TASK 。

以下命令可删除 AUTO_STATS_ADVISOR_TASK。

```
EXEC rdsadmin.rdsadmin_util.advisor_task_drop('AUTO_STATS_ADVISOR_TASK')
```

您可以使用 `rdsadmin.rdsadmin_util.dbms_stats_init` 重新启用 AUTO_STATS_ADVISOR_TASK。

重新启用 AUTO_STATS_ADVISOR_TASK

要重新启用 AUTO_STATS_ADVISOR_TASK，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_util.dbms_stats_init`。`dbms_stats_init` 过程不需要任何参数。

以下命令可重新启用 AUTO_STATS_ADVISOR_TASK。

```
EXEC rdsadmin.rdsadmin_util.dbms_stats_init()
```

传输表空间

使用 Amazon RDS 包 `rdsadmin.rdsadmin_transport_util` 将一组表空间从本地 Oracle 数据库复制到 RDS for Oracle 数据库实例。在物理层，可传输的表空间功能以递增方式将源数据文件和元数据文件复制到您的目标实例。您可以使用 Amazon EFS 或 Amazon S3 传输文件。有关更多信息，请参阅 使用 Oracle 可传输表空间进行迁移。

主题

- 将传输的表空间导入到您的数据库实例
- 将可传输的表空间元数据导入到数据库实例中
- 列出表空间导入后的孤立文件
- 表空间导入后的孤立数据文件

将传输的表空间导入到您的数据库实例

使用过程 `rdsadmin.rdsadmin_transport_util.import_xtts_tablespaces` 还原您之前从源数据库实例导出的表空间。在传输阶段中，您将备份只读表空间，导出 Data Pump 元数据，将这些文件传输到目标数据库实例，然后导入表空间。有关更多信息，请参阅 阶段 4：传输表空间。

语法

```
FUNCTION import_xtts_tablespaces(
    p_tablespace_list IN CLOB,
    p_directory_name  IN VARCHAR2,
    p_platform_id     IN NUMBER DEFAULT 13,
    p_parallel        IN INTEGER DEFAULT 0) RETURN VARCHAR2;
```

参数

参数名称	数据类型	默认值	必需	描述
p_tablespace_list	CLOB	—	是	要导入的表空间列表。
p_directory_name	VARCHAR2	—	是	包含表空间备份的目录。
p_platform_id	NUMBER	13	否	提供与在备份阶段中指定的平台 ID 相匹配的平台 ID。要查找平台的列表，请查询 V$TRANSPORTABLE_PLATFORM 。原定设置平台是 Linux x86 64 位，以 little endian 格式表示。
p_parallel	INTEGER	0	否	并行度。原定设置情况下，并行度处于禁用状态。

示例

以下示例从 *DATA_PUMP_DIR* 目录中导入表空间 *TBS1*、*TBS2* 和 *TBS3*。源平台是基于 AIX 的系统（64 位），其平台 ID 为 6。您可以通过查询 V$TRANSPORTABLE_PLATFORM 查找平台 ID。

```
VAR task_id CLOB

BEGIN
  :task_id:=rdsadmin.rdsadmin_transport_util.import_xtts_tablespaces(
        'TBS1,TBS2,TBS3',
        'DATA_PUMP_DIR',
        p_platform_id => 6);
END;
/

PRINT task_id
```

将可传输的表空间元数据导入到数据库实例中

使用过程 `rdsadmin.rdsadmin_transport_util.import_xtts_metadata` 将可传输的表空间元数据导入您的 RDS for Oracle 数据库实例。在操作期间，元数据导入的状态显示在表 `rdsadmin.rds_xtts_operation_info` 中。有关更多信息，请参阅 [步骤 5：在目标数据库实例上导入表空间元数据](#)。

语法

```
PROCEDURE import_xtts_metadata(
    p_datapump_metadata_file IN SYS.DBA_DATA_FILES.FILE_NAME%TYPE,
    p_directory_name         IN VARCHAR2,
    p_exclude_stats          IN BOOLEAN DEFAULT FALSE,
    p_remap_tablespace_list  IN CLOB DEFAULT NULL,
    p_remap_user_list        IN CLOB DEFAULT NULL);
```

参数

参数名称	数据类型	默认值	必需	描述
`p_datapump_metadata_file`	`SYS.DBA_DATA_FILES.FILE_NAME%TYPE`	—	是	包含可传输表空间的元数据的 Oracle Data Pump 文件的名称。
`p_directory_name`	`VARCHAR2`	—	是	包含 Data Pump 文件的目录。
`p_exclude_stats`	`BOOLEAN`	`FALSE`	否	指示是否排除统计数据的标志。
`p_remap_tablespace_list`	`CLOB`	NULL	否	要在元数据导入期间重新映射的表空间列表。采用格式 *from_tbs*:*to_tbs*。例如，指定 `users:user_data`。
`p_remap_user_list`	`CLOB`	NULL	否	要在元数据导入期间重新映射的用

参数名称	数据类型	默认值	必需	描述
				户架构列表。采用格式 *from_schema_name* :*to_schema_name* 。例如，指定 hr:human_resources 。

示例

该示例从文件 *xttdump.dmp* 导入表空间元数据，该文件位于目录 *DATA_PUMP_DIR* 中。

```
BEGIN
  rdsadmin.rdsadmin_transport_util.import_xtts_metadata('xttdump.dmp','DATA_PUMP_DIR');
END;
/
```

列出表空间导入后的孤立文件

使用 `rdsadmin.rdsadmin_transport_util.list_xtts_orphan_files` 过程列出表空间导入后孤立的数据文件。确定数据文件后，可以通过调用 `rdsadmin.rdsadmin_transport_util.cleanup_incomplete_xtts_import` 将其删除。

语法

```
FUNCTION list_xtts_orphan_files RETURN xtts_orphan_files_list_t PIPELINED;
```

示例

以下示例运行过程 `rdsadmin.rdsadmin_transport_util.list_xtts_orphan_files`。输出显示了两个孤立的数据文件。

```
SQL> SELECT * FROM TABLE(rdsadmin.rdsadmin_transport_util.list_xtts_orphan_files);

FILENAME       FILESIZE
-------------- ---------
datafile_7.dbf 104865792
datafile_8.dbf 104865792
```

表空间导入后的孤立数据文件

使用 rdsadmin.rdsadmin_transport_util.list_xtts_orphan_files 过程删除表空间导入后孤立的数据文件。运行此命令会在 BDUMP 目录中生成一个使用名称格式 rds-xtts-delete_xtts_orphaned_files-*YYYY-MM-DD.HH24-MI-SS.FF*.log 的日志文件。使用过程 rdsadmin.rdsadmin_transport_util.cleanup_incomplete_xtts_import 找出孤立的文件。您可以通过调用过程 rdsadmin.rds_file_util.read_text_file 来读取日志文件。有关更多信息，请参阅 第 6 阶段：清理剩余文件。

语法

```
PROCEDURE cleanup_incomplete_xtts_import(
    p_directory_name IN VARCHAR2);
```

参数

参数名称	数据类型	默认值	必需	描述
p_directory_name	VARCHAR2	—	是	包含孤立数据文件的目录。

示例

以下示例删除了 *DATA_PUMP_DIR* 中的孤立数据文件。

```
BEGIN
  rdsadmin.rdsadmin_transport_util.cleanup_incomplete_xtts_import('DATA_PUMP_DIR');
END;
/
```

以下示例读取了由上一个命令生成的日志文件。

```
SELECT *
FROM TABLE(rdsadmin.rds_file_util.read_text_file(
       p_directory => 'BDUMP',
       p_filename  => 'rds-xtts-
delete_xtts_orphaned_files-2023-06-01.09-33-11.868894000.log'));

TEXT
```

```
--------------------------------------------------------------------------------
orphan transported datafile datafile_7.dbf deleted.
orphan transported datafile datafile_8.dbf deleted.
```

配置高级 RDS for Oracle 功能

RDS for Oracle 支持各种高级功能，包括大页、实例存储和扩展数据类型。

主题

在 RDS for Oracle 实例存储中存储临时数据

在支持的 RDS for Oracle 数据库实例上，为临时表空间和数据库智能闪存缓存（闪存缓存）使用实例存储。

主题

RDS for Oracle 实例存储的概述

实例存储为 RDS for Oracle 数据库实例提供临时性块级存储。您可以使用实例存储来临时存储经常变化的信息。

实例存储基于物理附加到主机的非易失性存储规范（NVMe）设备。此存储针对低延迟、随机 I/O 性能和连续读取吞吐量进行了优化。

实例存储的大小因数据库实例类型而异。有关实例存储的更多信息，请参阅《适用于 Linux 实例的 Amazon Elastic Compute Cloud 用户指南》中的 Amazon EC2 实例存储。

主题

RDS for Oracle 实例存储中的数据类型

您可以将以下类型的 RDS for Oracle 临时数据放入实例存储中：

临时表空间

Oracle 数据库使用临时表空间来存储不适合放入内存中的中间查询结果。较大的查询会生成大量的中间数据，这些数据需要进行临时缓存，但不需要持久保留。尤其是，临时表空间可用于排序、哈希聚合和联接。如果您的 RDS for Oracle 数据库实例使用企业版或标准版 2，则可以将临时表空间放在实例存储中。

闪存缓存

闪存缓存可提高传统路径中单块随机读取的性能。最佳做法是调整缓存的大小以容纳大部分活动的数据集。如果 RDS for Oracle 数据库实例使用企业版，则可以将闪存缓存放在实例存储中。

原定设置情况下，将实例存储配置用于临时表空间而非闪存缓存。您无法将 Oracle 数据文件和数据库日志文件放在实例存储中。

RDS for Oracle 实例存储的优点

您可以考虑使用实例存储来存储可以承受丢失的临时文件和缓存。如果您想提高数据库性能，或者，如果增加的工作负载导致 Amazon EBS 存储出现性能问题，请考虑扩展到支持实例存储的实例类。

通过将临时表空间和闪存缓存放在实例存储上，您可以获得以下优势：

- 降低读取延迟
- 提高吞吐量
- 减少了 Amazon EBS 卷上的负载
- 由于减少了 Amazon EBS 负载，因此降低了存储和快照成本
- 减少了预调配高 IOPS 的需要，可能会降低您的总体成本

通过将临时表空间放在实例存储上，您可以立即提高使用临时空间的查询的性能。当您将闪存缓存放在实例存储上时，缓存块读取的延迟通常比 Amazon EBS 读取的延迟要低得多。闪存缓存需要“预热”，然后才能提供性能优势。缓存会自行预热，因为当块从数据库缓冲区缓存中过期时，数据库会将块写入闪存缓存。

> **Note**
>
> 在某些情况下，闪存缓存会因为缓存管理而导致性能开销。在生产环境中启用闪存缓存之前，我们建议您分析工作负载并在测试环境中测试缓存。

RDS for Oracle 实例存储支持的实例类

Amazon RDS 对于以下数据库实例类支持实例存储：

- db.m5d
- db.r5d
- db.x2idn
- db.x2iedn

RDS for Oracle 仅对于 BYOL 许可模式支持上述数据库实例类。有关更多信息，请参阅支持的 RDS for Oracle 数据库实例类 和EE 和 SE2 自带许可（BYOL）。

要查看受支持的数据库实例类型的总实例存储量，请在 Amazon CLI 中运行以下命令。

Example

```
aws ec2 describe-instance-types \
  --filters "Name=instance-type,Values=*5d.*large*" \
  --query "InstanceTypes[?contains(InstanceType,'m5d')||contains(InstanceType,'r5d')]
[InstanceType, InstanceStorageInfo.TotalSizeInGB]" \
  --output table
```

前面的命令返回实例存储的原始设备大小。RDS for Oracle 使用此空间的一小部分进行配置。实例存储中可用于临时表空间或闪存缓存的空间稍小。

RDS for Oracle 实例存储支持的引擎版本

以下 RDS for Oracle 数据库引擎版本支持实例存储：

- 21.0.0.0.ru-2022-01.rur-2022-01.r1 或更高的 Oracle Database 21c 版本
- 19.0.0.0.ru-2021-10.rur-2021-10.r1 或更高的 Oracle Database 19c 版本

RDS for Oracle 实例存储支持的 Amazon Web Services 区域

在支持其中一种或多种实例类型的所有 Amazon Web Services 区域中均可使用实例存储。有关 db.m5d 和 db.r5d 实例类的更多信息，请参阅 数据库实例类。有关 Amazon RDS for Oracle 支持的实例类的更多信息，请参阅 RDS for Oracle 数据库实例类。

RDS for Oracle 实例存储的成本

实例存储的成本计入已开启实例存储的实例的成本中。对 RDS for Oracle 数据库实例启用实例存储不会产生额外费用。有关已开启实例存储的实例的更多信息，请参阅 RDS for Oracle 实例存储支持的实例类。

开启 RDS for Oracle 实例存储

要为 RDS for Oracle 临时数据开启实例存储，请执行下列操作之一：

- 使用支持的实例类创建 RDS for Oracle 数据库实例。有关更多信息，请参阅 创建 Amazon RDS 数据库实例。
- 修改现有 RDS for Oracle 数据库实例以使用支持的实例类。有关更多信息，请参阅 修改 Amazon RDS 数据库实例。

配置 RDS for Oracle 实例存储

原定设置情况下，100% 的实例存储空间分配给临时表空间。要将实例存储配置为向闪存缓存和临时表空间分配空间，请在实例的参数组中设置以下参数：

db_flash_cache_size={DBInstanceStore*{0,2,4,6,8,10}/10}

> 此参数指定为闪存缓存分配的存储空间量。此参数仅对 Oracle Database 企业版有效。默认值为 `{DBInstanceStore*0/10}`。如果您为 `db_flash_cache_size` 设置非零值，则 RDS for Oracle 实例将在您重新启动实例后启用闪存缓存。

rds.instance_store_temp_size={DBInstanceStore*{0,2,4,6,8,10}/10}

> 此参数指定为临时表空间分配的存储空间量。默认值为 `{DBInstanceStore*10/10}`。对于 Oracle Database 企业版，此参数是可修改的，而对于标准版 2 为只读的。如果您为

rds.instance_store_temp_size 设置非零值，Amazon RDS 将在实例存储中为临时表空间分配空间。

您可以为不使用实例存储的数据库实例设置 db_flash_cache_size 和 rds.instance_store_temp_size 参数。在这种情况下，这两个设置的计算结果均为 0，这会关闭该功能。在这种情况下，您可以为不同的实例大小和不使用实例存储的实例使用相同的参数组。如果您修改这些参数，则确保重启关联的实例，以使更改生效。

> **Important**
>
> 如果您为临时表空间分配空间，Amazon RDS 不会自动创建临时表空间。要了解如何在实例存储上创建临时表空间，请参阅 在实例存储上创建临时表空间。

上述参数的组合值不得超过 10/10 或 100%。下表说明了有效和无效的参数设置。

db_flash_cache_size 设置	rds.instance_store_temp_size 设置	说明
db_flash_cache_size={DBInstanceStore*0/10}	rds.instance_store_temp_size={DBInstanceStore*10/10}	这是适用于所有 Oracle Database 版本的有效配置。Amazon RDS 将 100% 的实例存储空间分配给临时表空间。这是默认模式。
db_flash_cache_size={DBInstanceStore*10/10}	rds.instance_store_temp_size={DBInstanceStore*0/10}	这是仅适用于 Oracle Database 企业版的有效配置。Amazon RDS 将 100% 的实例存储空间分配给闪存缓存。

db_flash_cache_size 设置	rds.instance_store_temp_size 设置	说明
db_flash_cache_size={DBInstanceStore*2/10}	rds.instance_store_temp_size={DBInstanceStore*8/10}	这是仅适用于 Oracle Database 企业版的有效配置。Amazon RDS 将 20% 的实例存储空间分配给闪存缓存，而将 80% 的实例存储空间分配给临时表空间。
db_flash_cache_size={DBInstanceStore*6/10}	rds.instance_store_temp_size={DBInstanceStore*4/10}	这是仅适用于 Oracle Database 企业版的有效配置。Amazon RDS 将 60% 的实例存储空间分配给闪存缓存，而将 40% 的实例存储空间分配给临时表空间。
db_flash_cache_size={DBInstanceStore*2/10}	rds.instance_store_temp_size={DBInstanceStore*4/10}	这是仅适用于 Oracle Database 企业版的有效配置。Amazon RDS 将 20% 的实例存储空间分配给闪存缓存，而将 40% 的实例存储空间分配给临时表空间。

db_flash_cache_size 设置	rds.instance_store_temp_size 设置	说明
db_flash_cache_size={DBInstanceStore*8/10}	rds.instance_store_temp_size={DBInstanceStore*8/10}	这是无效的配置，因为实例存储空间的总百分比超过 100%。在这种情况下，Amazon RDS 会使此尝试失败。

更改数据库实例类型时的注意事项

如果您更改数据库实例类型，则这可能会影响实例存储上闪存缓存或临时表空间的配置。考虑以下修改及其影响：

您可以纵向扩展或缩减支持实例存储的数据库实例。

以下各值根据实例存储的新大小成比例增加或减少：

- 闪存缓存的新大小。
- 分配给驻留在实例存储中的临时表空间的空间。

例如，db.m5d.4xlarge 实例上的设置 `db_flash_cache_size={DBInstanceStore*6/10}` 提供了大约 340GB 的闪存缓存空间。如果您将实例类型纵向扩展到 db.m5d.8xlarge，则闪存缓存空间将增加到大约 680GB。

您将不使用实例存储的数据库实例修改为确实使用实例存储的实例。

如果将 `db_flash_cache_size` 设置为大于 `0` 的值，则配置闪存缓存。如果将 `rds.instance_store_temp_size` 设置为大于 `0` 的值，则会分配实例存储空间供临时表空间使用。RDS for Oracle 不会自动将临时文件移动到实例存储。有关使用分配的空间的信息，请参阅 在实例存储上创建临时表空间 或 在只读副本上向实例存储中添加临时文件。

您将使用实例存储的数据库实例修改为不使用实例存储的实例。

在这种情况下，RDS for Oracle 会删除闪存缓存。RDS 在 Amazon EBS 卷上重新创建当前位于实例存储中的临时文件。新临时文件的最大大小是 `rds.instance_store_temp_size` 参数以前的大小。

在 Oracle 只读副本上使用实例存储

只读副本在实例存储上支持闪存缓存和临时表空间。虽然闪存缓存的工作方式与在主数据库实例上的工作方式相同，但请注意对于临时表空间的以下区别：

- 您无法在只读副本上创建临时表空间。如果您在主实例上创建新的临时表空间，则 RDS for Oracle 会在没有临时文件的情况下复制表空间信息。要添加新的临时文件，请使用下列任一方法：
  - 适用 Amazon RDS 过程 `rdsadmin.rdsadmin_util.add_inst_store_tempfile`。RDS for Oracle 在只读副本上的实例存储中创建一个临时文件，并将其添加到指定的临时表空间中。
  - 运行 `ALTER TABLESPACE … ADD TEMPFILE` 命令。RDS for Oracle 将临时文件放在 Amazon EBS 存储上。

> **Note**
> 主数据库实例和只读副本上的临时文件大小和存储类型可能不同。

- 您只能在主数据库实例上管理原定设置临时表空间设置。RDS for Oracle 会将该设置复制到所有只读副本。
- 您只能在主数据库实例上配置临时表空间组。RDS for Oracle 会将该设置复制到所有只读副本。

在实例存储和 Amazon EBS 上配置临时表空间组

您可以将临时表空间组配置为包括实例存储和 Amazon EBS 上的临时表空间。当您想要的临时存储空间超过 `rds.instance_store_temp_size` 的最大设置所允许的空间时，此方法很有用。

当您在实例存储和 Amazon EBS 上配置临时表空间组时，这两个表空间的性能特征显著不同。Oracle 数据库根据内部算法，选择表空间来处理查询。因此，类似查询的性能可能会有所不同。

通常，您在实例存储中创建临时表空间，如下所示：

1. 在实例存储中创建临时表空间。
2. 将新的表空间设置为数据库原定设置临时表空间。

如果实例存储中的表空间大小不足，则可以按如下方式创建额外的临时存储：

1. 将实例存储中的临时表空间分配给临时表空间组。
2. 如果临时表空间不存在，请在 Amazon EBS 中创建一个新的临时表空间。

3. 将 Amazon EBS 中的临时表空间分配给包含实例存储表空间的同一个表空间组。
4. 将表空间组设置为原定设置临时表空间。

以下示例假设实例存储中临时表空间的大小不符合您的应用程序要求。该示例在实例存储中创建临时表空间 `temp_in_inst_store`，将其分配给表空间组 `temp_group`，将名为 `temp_in_ebs` 的现有 Amazon EBS 表空间添加到该组中，并将该组设置为原定设置临时表空间。

```
SQL> EXEC rdsadmin.rdsadmin_util.create_inst_store_tmp_tblspace('temp_in_inst_store');

PL/SQL procedure successfully completed.

SQL> ALTER TABLESPACE temp_in_inst_store TABLESPACE GROUP temp_group;

Tablespace altered.

SQL> ALTER TABLESPACE temp_in_ebs TABLESPACE GROUP temp_group;

Tablespace altered.

SQL> EXEC rdsadmin.rdsadmin_util.alter_default_temp_tablespace('temp_group');

PL/SQL procedure successfully completed.

SQL> SELECT * FROM DBA_TABLESPACE_GROUPS;

GROUP_NAME                     TABLESPACE_NAME
------------------------------ ------------------------------
TEMP_GROUP                     TEMP_IN_EBS
TEMP_GROUP                     TEMP_IN_INST_STORE

SQL> SELECT PROPERTY_VALUE FROM DATABASE_PROPERTIES WHERE
 PROPERTY_NAME='DEFAULT_TEMP_TABLESPACE';

PROPERTY_VALUE
--------------
TEMP_GROUP
```

删除 RDS for Oracle 实例存储

要删除实例存储，请修改 RDS for Oracle 数据库实例以使用不支持实例存储的实例类型，如 db.m5 或 db.r5。

为 RDS for Oracle 实例开启大页

Amazon RDS for Oracle 支持 Linux 内核大页，提高了数据库扩展能力。大页会导致页表变小，花在内存管理上的 CPU 时间也减少，从而提高大数据库实例的性能。有关更多信息，请参阅 Oracle 文档中的巨页概述。

您可以将大页与所有支持的 RDS for Oracle 版本和修订版一起使用。

`use_large_pages` 参数控制是否为数据库实例开启大页。该参数的可能设置包括 `ONLY`、`FALSE` 和 `{DBInstanceClassHugePagesDefault}`。在 Oracle 的默认数据库参数组中，`use_large_pages` 参数设置为 `{DBInstanceClassHugePagesDefault}`。

要控制是否自动为数据库实例开启大页，您可以在参数组中使用 `DBInstanceClassHugePagesDefault` 公式变量。该值如下所示确定：

- 对于下表中提及的数据库实例类别，`DBInstanceClassHugePagesDefault` 默认情况下计算结果始终为 `FALSE`，而 `use_large_pages` 的计算结果为 `FALSE`。如果数据库实例类至少具有 14GiB 内存，您可为这些数据库实例类手动开启大页。
- 对于未在下表中提及的数据库实例类别，如果数据库实例类别内存小于 14 GiB，则 `DBInstanceClassHugePagesDefault` 的计算结果始终为 `FALSE`。此外，`use_large_pages` 计算结果为 `FALSE`。
- 对于未在下表中提及的数据库实例类别，如果实例类别内存至少为 14 GiB 且小于 100 GiB，则 `DBInstanceClassHugePagesDefault` 默认情况下计算结果为 `TRUE`。此外，`use_large_pages` 计算结果为 `ONLY`。您可以通过将 `use_large_pages` 设置为 `FALSE`，手动关闭大页。
- 对于未在下表中提及的数据库实例类别，如果实例类内存至少为 100 GiB，则 `DBInstanceClassHugePagesDefault` 的计算结果始终为 `TRUE`。此外，`use_large_pages` 的计算结果为 `ONLY`，并且不能禁用大页。

原定设置情况下，不为以下数据库实例类开启大页。

数据库实例类系列	原定设置情况下未开启大页的数据库实例类
db.m5	db.m5.large
db.m4	db.m4.large，db.m4.xlarge，db.m4.2xlarge，db.m4.4xlarge，db.m4.10xlarge

数据库实例类系列	原定设置情况下未开启大页的数据库实例类
db.t3	db.t3.micro，db.t3.small，db.t3.medium，db.t3.large

有关数据库实例类的更多信息，请参阅数据库实例类的硬件规格。

要为新的或现有的数据库实例手动开启大页，请将 use_large_pages 参数设置为 ONLY。大页不能与 Oracle 自动内存管理 (AMM) 一起使用。如果将 use_large_pages 参数设置为 ONLY，则还必须将 memory_target 和 memory_max_target 都设置为 0。有关为数据库实例设置数据库参数的更多信息，请参阅Amazon RDS 的参数组。

您也可以设置 sga_target、sga_max_size 和 pga_aggregate_target 参数。当设置系统全局区域 (SGA) 和程序全局区域 (PGA) 内存参数时，请将这些值加在一起。从可用实例内存 (DBInstanceClassMemory) 中减去此计算得到的总计值，可确定大页分配后的可用内存。必须保留至少 2 GiB 可用内存或总可用实例内存的 10% (取两者中较小的值)。

配置参数之后，必须重启数据库实例才能使更改生效。有关更多信息，请参阅“重启中的数据库实例”。

> **Note**
>
> Oracle 数据库实例会推迟对与 SGA 相关的初始化参数的更改，直到您重启实例而不进行故障转移。在 Amazon RDS 控制台中，选择重启，但不要 选择通过故障转移重启。在 Amazon CLI 中，调用带 reboot-db-instance 参数的 --no-force-failover 命令。在故障转移期间或导致实例重启的其他维护操作期间，数据库实例不会处理与 SGA 相关的参数。

针对大页，以下示例参数配置手动启用大页。您应根据自己的需要设置这些值。

```
memory_target                = 0
memory_max_target            = 0
pga_aggregate_target         = {DBInstanceClassMemory*1/8}
sga_target                   = {DBInstanceClassMemory*3/4}
sga_max_size                 = {DBInstanceClassMemory*3/4}
use_large_pages              = ONLY
```

假设在参数组中设置以下参数值。

```
memory_target                = IF({DBInstanceClassHugePagesDefault}, 0,
 {DBInstanceClassMemory*3/4})
```

```
memory_max_target         = IF({DBInstanceClassHugePagesDefault}, 0,
 {DBInstanceClassMemory*3/4})
pga_aggregate_target      = IF({DBInstanceClassHugePagesDefault},
 {DBInstanceClassMemory*1/8}, 0)
sga_target                = IF({DBInstanceClassHugePagesDefault},
 {DBInstanceClassMemory*3/4}, 0)
sga_max_size              = IF({DBInstanceClassHugePagesDefault},
 {DBInstanceClassMemory*3/4}, 0)
use_large_pages           = {DBInstanceClassHugePagesDefault}
```

参数组由内存小于 100 GiB 的 db.r4 数据库实例类使用。如果使用这些参数设置并将 `use_large_pages` 设置为 `{DBInstanceClassHugePagesDefault}`，则为 db.r4 实例开启大页。

请考虑在参数组中设置了以下参数值的另一个示例。

```
memory_target            = IF({DBInstanceClassHugePagesDefault}, 0,
 {DBInstanceClassMemory*3/4})
memory_max_target        = IF({DBInstanceClassHugePagesDefault}, 0,
 {DBInstanceClassMemory*3/4})
pga_aggregate_target     = IF({DBInstanceClassHugePagesDefault},
 {DBInstanceClassMemory*1/8}, 0)
sga_target               = IF({DBInstanceClassHugePagesDefault},
 {DBInstanceClassMemory*3/4}, 0)
sga_max_size             = IF({DBInstanceClassHugePagesDefault},
 {DBInstanceClassMemory*3/4}, 0)
use_large_pages          = FALSE
```

该参数组由内存小于 100 GiB 的 db.r4 数据库实例类和 db.r5 数据库实例类使用。如果使用这些参数设置，将在 db.r4 和 db.r5 实例上关闭大页。

> **Note**
>
> 如果该参数组由具有至少 100 GiB 内存的 db.r4 数据库实例类或 db.r5 数据库实例类使用，则将覆盖 `FALSE` 的 `use_large_pages` 设置并将其设置为 `ONLY`。在这种情况下，将发送有关覆盖的客户通知。

当大页在数据库实例上处于活动状态后，您可以启用增强监控以查看大页信息。有关更多信息，请参阅[使用增强监控来监控操作系统指标]。

在 RDS for Oracle 中开启扩展数据类型

Amazon RDS for Oracle 支持扩展数据类型。对于扩展数据类型，VARCHAR2、NVARCHAR2 和 RAW 数据类型的最大大小为 32767 字节。要使用扩展数据类型，请将 MAX_STRING_SIZE 参数设置为 EXTENDED。有关更多信息，请参阅 Oracle 文档中的扩展数据类型。

如果您不希望使用扩展数据类型，请将 MAX_STRING_SIZE 参数设置为 STANDARD（默认值）。在这种情况下，VARCHAR2 和 NVARCHAR2 数据类型的大小限制为 4,000 字节，RAW 数据类型的大小限制为 2,000 字节。

您可以在新的或现有的数据库实例上开启扩展数据类型。对于新的数据库实例，在开启扩展数据类型时，数据库实例的创建时间通常较长。对于现有的数据库实例，数据库实例在转换过程中不可用。

扩展数据类型的注意事项

为数据库实例启用扩展数据类型时，请考虑以下几点：

- 在为新的或现有的数据库实例开启扩展数据类型时，必须重启实例才能使更改生效。
- 在开启扩展数据类型之后，无法将数据库实例改回为使用数据类型的标准大小。如果将 MAX_STRING_SIZE 参数重新设置为 STANDARD，则会导致 incompatible-parameters 状态。
- 在还原使用扩展数据类型的数据库实例时，您必须指定一个参数组并将 MAX_STRING_SIZE 参数设置为 EXTENDED。在还原期间，如果指定默认参数组或任何其他参数组并将 MAX_STRING_SIZE 设置为 STANDARD，则会导致 incompatible-parameters 状态。
- 如果由于 incompatible-parameters 设置而导致数据库实例状态为 MAX_STRING_SIZE，数据库实例将保持不可用状态，直到您将 MAX_STRING_SIZE 参数设置为 EXTENDED 并重新引导数据库实例。

为新数据库实例开启扩展数据类型

当您创建将 MAX_STRING_SIZE 设置为 EXTENDED 的数据库实例时，该实例将显示 MAX_STRING_SIZE 设置为默认 STANDARD。重启实例以启用更改。

为新数据库实例开启扩展数据类型

1. 在参数组中将 MAX_STRING_SIZE 参数设置为 EXTENDED。

   要设置该参数，您可以创建新的参数组或修改现有的参数组。

   有关更多信息，请参阅 Amazon RDS 的参数组。

2. 创建新的 RDS for Oracle 数据库实例。

   有关更多信息，请参阅 创建 Amazon RDS 数据库实例。
3. 将其 MAX_STRING_SIZE 设置为 EXTENDED 的参数组与数据库实例相关联。

   有关更多信息，请参阅 创建 Amazon RDS 数据库实例。
4. 重新引导数据库实例以使参数更改生效。

   有关更多信息，请参阅 重启中的数据库实例。

为现有数据库实例开启扩展数据类型

在修改数据库实例以开启扩展数据类型时，RDS 将转换数据库中的数据以使用扩展的大小。参数更改后，下次重启数据库时会发生转换和停机。数据库实例在转换期间不可用。

转换数据所需的时间取决于数据库实例类、数据库大小和最后一个数据库快照的时间。为减少停机时间，请考虑就在重启前拍摄快照。这缩短了转换工作流程期间执行的备份的时间。

> **Note**
>
> 在开启扩展数据类型后，您无法在转换期间执行时间点还原以还原到某个时间。您可以还原到紧靠转换前或转换后的时间。

为现有数据库实例开启扩展数据类型

1. 拍摄数据库的快照。

   如果在数据库中具有无效的对象，Amazon RDS 将尝试重新编译这些对象。如果 Amazon RDS 无法重新编译无效的对象，转换为扩展数据类型可能会失败。如果在转换过程中出现问题，您可以使用快照还原数据库。请始终在转换之前检查无效的对象，并修复或删除这些无效的对象。对于生产数据库，我们建议先在数据库实例的副本上测试转换过程。

   有关更多信息，请参阅“为 Amazon RDS 的单可用区数据库实例创建数据库快照”。
2. 在参数组中将 MAX_STRING_SIZE 参数设置为 EXTENDED。

   要设置该参数，您可以创建新的参数组或修改现有的参数组。

   有关更多信息，请参阅“Amazon RDS 的参数组”。

3. 修改数据库实例以将其与 `MAX_STRING_SIZE` 设置为 `EXTENDED` 的参数组相关联。

   有关更多信息，请参阅“修改 Amazon RDS 数据库实例”。

4. 重新引导数据库实例以使参数更改生效。

   有关更多信息，请参阅“重启中的数据库实例”。

将数据导入到 Amazon RDS 上的 Oracle

将数据导入 Amazon RDS for Oracle 数据库实例的方式取决于以下各项：

- 您拥有的数据量
- 数据库中的数据库对象的数量
- 数据库中的各种数据库对象

例如，您可以根据需要使用以下工具：

- Oracle SQL Developer – 导入一个简单的、大小为 20MB 的数据库。
- Oracle Data Pump – 导入复杂的数据库，或大小为数百 MB 甚至 TB 级的数据库。例如，您可以将表空间从本地数据库传输到 RDS for Oracle 数据库实例。您可以使用 Amazon S3 或 Amazon EFS 来传输数据文件和元数据。有关更多信息，请参阅 使用 Oracle 可传输表空间进行迁移、Amazon EFS 集成 和 Amazon S3 集成。
- Amazon Database Migration Service（Amazon DMS）– 无需停机即可迁移数据库。有关 Amazon DMS 的更多信息，请参阅 Amazon Database Migration Service 是什么和博文 Migrating Oracle databases with near-zero downtime using Amazon DMS。

> **Important**
>
> 在使用前述的迁移技术之前，我们建议您对数据库进行备份。要导入数据，您可以通过创建快照来备份您的 RDS for Oracle 数据库实例。稍后，您可以还原快照。有关更多信息，请参阅"备份、还原和导出数据"。

对于许多数据库引擎，将继续进行持续复制，直到您已准备好切换到目标数据库。您可以使用 Amazon DMS 从同一数据库引擎或其他引擎迁移到 RDS for Oracle。如果您从其他数据库引擎进行迁移，可以使用 Amazon Schema Conversion Tool 迁移 Amazon DMS 未迁移的架构对象。

主题

- 使用 Oracle SQL Developer 导入
- 使用 Oracle 可传输表空间进行迁移
- 使用 Oracle Data Pump 导入
- 使用 Oracle Export/Import 进行导入

- 使用 Oracle SQL*Loader 进行导入
- 使用 Oracle 实体化视图进行迁移

使用 Oracle SQL Developer 导入

Oracle SQL Developer 是 Oracle 发布的一种免费的图形化 Java 工具。SQL Developer 为用户提供了选项，可在两个 Oracle 数据库之间迁移数据，或将数据从其他数据库（如 MySQL）迁移到 Oracle 数据库。此工具最适合用于迁移小型数据库。

您可以将此工具安装到台式计算机 (Windows、Linux 或 Mac) 或服务器上。安装 SQL Developer 后，您可以使用该工具连接至源数据库和目标数据库。可使用“工具”菜单上的数据库复制命令将数据复制到 RDS for Oracle 数据库实例。

要下载 SQL Developer，请转到 http://www.oracle.com/technetwork/developer-tools/sql-developer。

建议您在开始迁移数据前先阅读 Oracle SQL Developer 产品文档。Oracle 还提供关于如何从其他数据库 (包括 MySQL 和 SQL Server) 中迁移数据的文档。有关更多信息，请参阅 Oracle 文档中的 http://www.oracle.com/technetwork/database/migration。

使用 Oracle 可传输表空间进行迁移

您可以使用 Oracle 可传输表空间功能，将一组表空间从本地 Oracle 数据库复制到 RDS for Oracle 数据库实例。在物理层，您可以使用 Amazon EFS 或 Amazon S3 将源数据文件和元数据文件传输到目标数据库实例。可传输表空间功能使用 `rdsadmin.rdsadmin_transport_util` 软件包。有关此软件包的语法和语义，请参阅传输表空间。

有关阐述如何传输表空间的博客文章，请参阅 Migrate Oracle Databases to Amazon using transportable tablespace，以及 Amazon RDS for Oracle Transportable Tablespaces using RMAN。

主题

- Oracle 可传输表空间概述
- 第 1 阶段：设置源主机
- 第 2 阶段：准备完整的表空间备份
- 第 3 阶段：制作和传输增量备份
- 阶段 4：传输表空间
- 第 5 阶段：验证传输的表空间

- 第 6 阶段：清理剩余文件

Oracle 可传输表空间概述

可传输的表空间集由正在传输的表空间集的数据文件和包含表空间元数据的导出转储文件组成。在诸如可传输表空间之类的物理迁移解决方案中，您可以传输物理文件：数据文件、配置文件和 Data Pump 转储文件。

主题

- 可传输表空间的优缺点
- 可传输表空间的限制
- 可传输表空间的先决条件

可传输表空间的优缺点

当您需要以最少的停机时间将一个或多个大型表空间迁移到 RDS 时，建议您使用可传输的表空间。与逻辑迁移相比，可传输的表空间具有以下优势：

- 停机时间低于大多数其他 Oracle 迁移解决方案。
- 由于可传输表空间功能仅复制物理文件，因此，它避免了在逻辑迁移中可能发生的数据完整性错误和逻辑损坏。
- 不需要额外的许可证。
- 您可以跨不同的平台和字节顺序类型迁移一组表空间，例如，从 Oracle Solaris 平台迁移到 Linux。但是，不支持在 Windows 服务器之间传输表空间。

> **Note**
>
> Linux 经过充分测试并得到支持。并非所有 UNIX 版本都经过测试。

如果您使用可传输的表空间，则可以使用 Amazon S3 或 Amazon EFS 传输数据：

- 使用 EFS 时，您的备份在导入期间将保留在 EFS 文件系统中。之后，您可以删除这些文件。在此技术中，您无需为数据库实例预调配 EBS 存储。出于这个原因，我们建议使用 Amazon EFS 而不是 S3。有关更多信息，请参阅 Amazon EFS 集成。
- 使用 S3 时，您可以将 RMAN 备份下载到附加到数据库实例的 EBS 存储。在导入期间，文件将保留在您的 EBS 存储中。导入后，您可以释放此空间，该空间仍保持分配给您的数据库实例。

可传输表空间的主要缺点是您需要相对较高级的 Oracle 数据库知识。有关更多信息，请参见《Oracle 数据库管理员指南》中的在数据库之间传输表空间。

可传输表空间的限制

当您在 RDS for Oracle 中使用可传输表空间时，将适用 Oracle 数据库对此功能的限制。有关更多信息，请参见《Oracle 数据库管理员指南》中的可传输表空间的限制和传输数据的一般限制。请注意 RDS for Oracle 中可传输表空间的以下其他限制：

- 源数据库或目标数据库都不能使用标准版 2（SE2）。仅支持企业版。
- 您不能使用 Oracle Database 11g 数据库作为源。RMAN 跨平台可传输表空间功能依赖于 RMAN 传输机制，而 Oracle Database 11g 不支持该机制。
- 您无法使用可传输的表空间从 RDS for Oracle 数据库实例迁移数据。您只能使用可传输的表空间将数据迁移到 RDS for Oracle 数据库实例。
- 不支持 Windows 操作系统。
- 您无法将表空间传输到较低版本级别的数据库中。目标数据库的版本级别必须与源数据库相同或更高。例如，您无法将表空间从 Oracle Database 21c 传输到 Oracle Database 19c。
- 您无法传输诸如 `SYSTEM` 和 `SYSAUX` 之类的管理表空间。
- 您无法传输非数据对象，例如 PL/SQL 软件包、Java 类、视图、触发器、序列、用户、角色和临时表。要传输非数据对象，请手动创建它们或使用 Data Pump 元数据导出和导入。有关更多信息，请参阅 My Oracle Support Note 1454872.1。
- 您无法传输已加密或使用加密列的表空间。
- 如果您使用 Amazon S3 传输文件，则支持的最大文件大小为 5TiB。
- 如果源数据库使用诸如 Spatial 等 Oracle 选项，则除非在目标数据库上配置了相同的选项，否则无法传输表空间。
- 在 Oracle 副本配置中，您无法将表空间传输到 RDS for Oracle 数据库实例。解决方法是，您可以删除所有副本，传输表空间，然后重新创建副本。

可传输表空间的先决条件

开始之前，完成以下任务：

- 查看 My Oracle Support 的以下文档中描述的可传输表空间的要求：
  - 使用跨平台增量备份减少可传输表空间停机时间（文档 ID 2471245.1）
  - 可传输表空间（TTS）局限性和限制：详细信息、参考和适用的版本（文档 ID 1454872.1）

  - 可传输表空间（TTS）的主要注意事项 -- 常见疑问和问题（文档 ID 1166564.1）
- 计划字节顺序转换。如果您指定源平台 ID，则 RDS for Oracle 会自动转换字节顺序。要了解如何查找平台 ID，请参阅 Data Guard 对同一 Data Guard 配置中异构主备用服务器和物理备用服务器的支持（文档 ID 413484.1）。
- 确保在目标数据库实例上启用了可传输表空间功能。只有在运行以下查询时没有出现 ORA-20304 错误的情况下，才会启用该功能：

```
SELECT * FROM TABLE(rdsadmin.rdsadmin_transport_util.list_xtts_orphan_files);
```

  如果未启用可传输表空间功能，请重启您的数据库实例。有关更多信息，请参阅 重启中的数据库实例。
- 确保源和目标数据库中的时区文件相同。
- 确保源数据库和目标数据库上的数据库字符集满足以下任一要求：
  - 字符集是相同的。
  - 字符集是兼容的。有关兼容性要求的列表，请参阅 Oracle 数据库文档中的 General Limitations on Transporting Data。
- 如果您计划使用 Amazon S3 传输文件，请执行以下操作：
  - 确保 Amazon S3 桶可用于文件传输，并且 Amazon S3 桶与数据库实例处于相同的 Amazon 区域中。有关说明，请参阅 Amazon Simple Storage Service 入门指南 中的创建存储桶。
  - 按照为与 Amazon S3 集成的 RDS for Oracle 配置 IAM 权限中的说明为 Amazon RDS 集成准备 Amazon S3 桶。
- 如果您计划使用 Amazon EFS 传输文件，请确保已按照Amazon EFS 集成中的说明配置 EFS。
- 我们强烈建议您在目标数据库实例中开启自动备份。由于元数据导入步骤可能会失败，因此，必须能够将数据库实例还原到其导入前的状态，从而避免需要再次备份、传输和导入表空间。

第 1 阶段：设置源主机

在此步骤中，您将复制 My Oracle Support 提供的传输表空间脚本，并设置必要的配置文件。在以下步骤中，源主机正在运行包含要传输到目标实例的表空间的数据库。

设置您的源主机

1. 以 Oracle 主目录的所有者身份登录到您的源主机。
2. 确保 ORACLE_HOME 和 ORACLE_SID 环境变量指向您的源数据库。

3. 以管理员身份登录数据库，并验证时区版本、数据库字符集和国家/地区字符集是否与目标数据库中相同。

```
SELECT * FROM V$TIMEZONE_FILE;
SELECT * FROM NLS_DATABASE_PARAMETERS
  WHERE PARAMETER IN ('NLS_CHARACTERSET','NLS_NCHAR_CHARACTERSET');
```

4. 按照 Oracle 支持说明 2471245.1 中所述，设置可传输表空间实用程序。

   安装包括在源主机上编辑 `xtt.properties` 文件。以下示例 `xtt.properties` 文件指定 `/dsk1/backups` 目录中三个表空间的备份。这些是您打算传输到目标数据库实例的表空间。它还指定源平台 ID 以自动转换字节顺序。

   > **Note**
   >
   > 有关有效的平台 ID，请参阅 Data Guard 对同一 Data Guard 配置中异构主备用服务器和物理备用服务器的支持（文档 ID 413484.1）。

```
#linux system
platformid=13
#list of tablespaces to transport
tablespaces=TBS1,TBS2,TBS3
#location where backup will be generated
src_scratch_location=/dsk1/backups
#RMAN command for performing backup
usermantransport=1
```

第 2 阶段：准备完整的表空间备份

在此阶段中，您将首次备份表空间，将备份传输到目标主机，然后使用过程 `rdsadmin.rdsadmin_transport_util.import_xtts_tablespaces` 将其还原。此阶段完成后，初始表空间备份将驻留在您的目标数据库实例上，并且可以使用增量备份进行更新。

主题

- 步骤 1：备份源主机上的表空间
- 步骤 2：将备份文件传输到您的目标数据库实例
- 步骤 3：在目标数据库实例上导入表空间

步骤 1：备份源主机上的表空间

在此步骤中，您将使用 xttdriver.pl 脚本对表空间进行完整备份。xttdriver.pl 的输出存储在 TMPDIR 环境变量中。

备份您的表空间

1. 如果您的表空间处于只读模式，请以具有 ALTER TABLESPACE 权限的用户身份登录到源数据库，并将表空间置于读/写模式。否则，请跳到下一步。

   以下示例将 tbs1、tbs2 和 tbs3 置于读/写模式。

   ```
   ALTER TABLESPACE tbs1 READ WRITE;
   ALTER TABLESPACE tbs2 READ WRITE;
   ALTER TABLESPACE tbs3 READ WRITE;
   ```

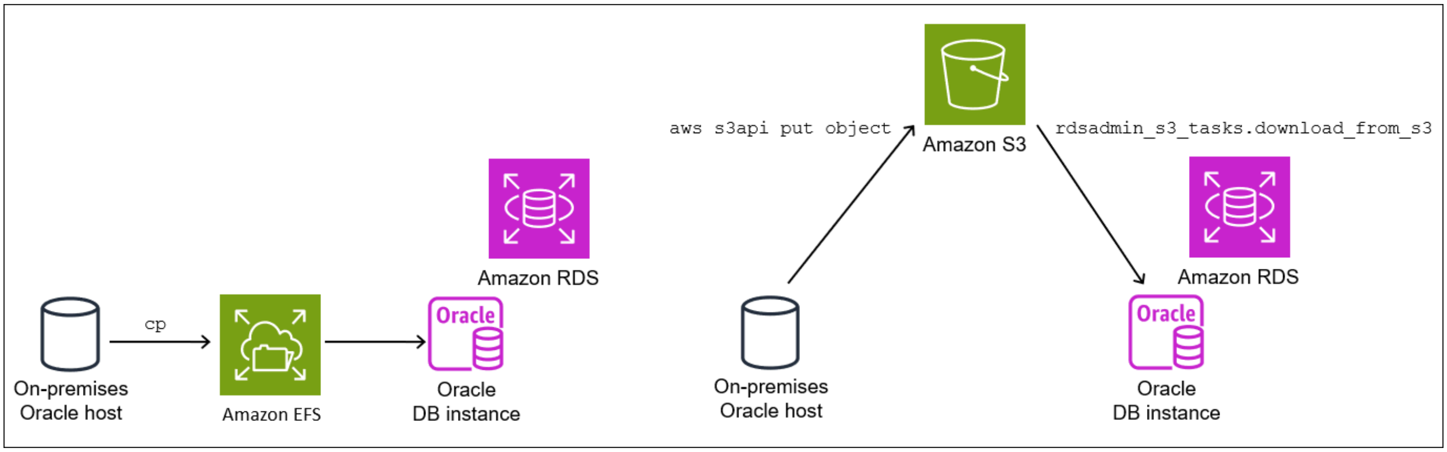
2. 使用 xttdriver.pl 脚本备份您的表空间。或者，您可以指定 --debug 以在调试模式下运行脚本。

   ```
   export TMPDIR=location_of_log_files
   cd location_of_xttdriver.pl
   $ORACLE_HOME/perl/bin/perl xttdriver.pl --backup
   ```

步骤 2：将备份文件传输到您的目标数据库实例

在此步骤中，将备份和配置文件从临时位置复制到目标数据库实例。请选择以下选项之一：

- 如果源主机和目标主机共享 Amazon EFS 文件系统，请使用操作系统实用程序（例如 cp）将您的备份文件和 res.txt 文件从临时位置复制到共享目录。然后跳至 步骤 3：在目标数据库实例上导入表空间。
- 如果您需要将备份暂存到 Amazon S3 桶，请完成以下步骤。



步骤 2.2：将备份上传到 Amazon S3 桶

将您的备份和 `res.txt` 文件从临时目录上传到您的 Amazon S3 桶。有关更多信息，请参阅《Amazon Simple Storage Service 开发人员指南》中的上传对象。

步骤 2.3：将备份从 Amazon S3 桶下载到目标数据库实例

在此步骤中，您将使用过程 `rdsadmin.rdsadmin_s3_tasks.download_from_s3` 将备份下载到您的 RDS for Oracle 数据库实例。

从您的 Amazon S3 桶下载备份

1. 启动 SQL*Plus 或 Oracle SQL Developer，并登录到 RDS for Oracle 数据库实例。
2. 使用 Amazon RDS 过程 `rdsadmin.rdsadmin_s3_tasks.download_from_s3` 将备份从 Amazon S3 桶下载到您的目标数据库实例。以下示例将所有文件从名为 *amzn-s3-demo-bucket* 的 Amazon S3 存储桶下载到 *DATA_PUMP_DIR* 目录中。

```
EXEC UTL_FILE.FREMOVE ('DATA_PUMP_DIR', 'res.txt');
SELECT rdsadmin.rdsadmin_s3_tasks.download_from_s3(
  p_bucket_name    =>  'amzn-s3-demo-bucket',
  p_directory_name =>  'DATA_PUMP_DIR')
AS TASK_ID FROM DUAL;
```

`SELECT` 语句返回 `VARCHAR2` 数据类型的任务 ID。有关更多信息，请参阅 将文件从 Amazon S3 存储桶下载到 Oracle 数据库实例。

步骤 3：在目标数据库实例上导入表空间

使用过程 `rdsadmin.rdsadmin_transport_util.import_xtts_tablespaces` 将表空间还原到目标数据库实例。此过程会自动将数据文件转换为正确的字节顺序格式。

如果您从 Linux 以外的平台导入，请在调用 `p_platform_id` 时使用参数 `import_xtts_tablespaces` 指定源平台。确保您指定的平台 ID 与 [步骤 2：在源主机上导出表空间元数据](#) 中 `xtt.properties` 文件中指定的平台 ID 匹配。

在目标数据库实例上导入表空间

1. 启动 Oracle SQL 客户端，并以主用户身份登录到目标 RDS for Oracle 数据库实例。
2. 运行过程 `rdsadmin.rdsadmin_transport_util.import_xtts_tablespaces`，同时指定要导入的表空间和包含备份的目录。

   以下示例从 *DATA_PUMP_DIR* 目录中导入表空间 *TBS1*、*TBS2* 和 *TBS3*。源平台是基于 AIX 的系统（64 位），其平台 ID 为 6。您可以通过查询 `V$TRANSPORTABLE_PLATFORM` 查找平台 ID。

   ```
   VAR task_id CLOB

   BEGIN
     :task_id:=rdsadmin.rdsadmin_transport_util.import_xtts_tablespaces(
           'TBS1,TBS2,TBS3',
           'DATA_PUMP_DIR',
           p_platform_id => 6);
   END;
   /

   PRINT task_id
   ```

3. （可选）通过查询表 `rdsadmin.rds_xtts_operation_info` 来监控进度。`xtts_operation_state` 列显示值 `EXECUTING`、`COMPLETED` 或 `FAILED`。

   ```
   SELECT * FROM rdsadmin.rds_xtts_operation_info;
   ```

   > **Note**
   >
   > 对于长时间运行的操作，也可以查询 `V$SESSION_LONGOPS`、`V$RMAN_STATUS` 和 `V$RMAN_OUTPUT`。

4. 使用上一步中的任务 ID 查看已完成的导入的日志。

```
SELECT * FROM TABLE(rdsadmin.rds_file_util.read_text_file('BDUMP',
 'dbtask-'||'&task_id'||'.log'));
```

在继续下一步之前，请确保导入成功。

第 3 阶段：制作和传输增量备份

在此阶段，您需要在源数据库处于活动状态时定期制作和传输增量备份。此技术减小了最终表空间备份的大小。如果您进行多个增量备份，则必须先在最后一个增量备份之后复制 `res.txt` 文件，然后才能将其应用到目标实例。

步骤与第 2 阶段：准备完整的表空间备份中的步骤相同，不同之处为导入步骤是可选的。

阶段 4：传输表空间

在此阶段，您将备份只读表空间并导出 Data Pump 元数据，将这些文件传输到目标主机，然后导入表空间和元数据。

主题

步骤 1：备份只读表空间

此步骤与步骤 1：备份源主机上的表空间相同，但有一个关键区别：在最后一次备份表空间之前，您将表空间置于只读模式。

以下示例将 `tbs1`、`tbs2`、和 `tbs3` 置于只读模式。

```
ALTER TABLESPACE tbs1 READ ONLY;
ALTER TABLESPACE tbs2 READ ONLY;
ALTER TABLESPACE tbs3 READ ONLY;
```

步骤 2：在源主机上导出表空间元数据

通过在源主机上运行 expdb 实用程序来导出表空间元数据。以下示例将表空间 *TBS1*、*TBS2* 和 *TBS3* 导出到目录 *DATA_PUMP_DIR* 中的转储文件 *xttdump.dmp*。

```
expdp username/pwd \
dumpfile=xttdump.dmp \
directory=DATA_PUMP_DIR \
statistics=NONE \
transport_tablespaces=TBS1,TBS2,TBS3 \
transport_full_check=y \
logfile=tts_export.log
```

如果 *DATA_PUMP_DIR* 是 Amazon EFS 中的共享目录，请跳至步骤 4：在目标数据库实例上导入表空间。

步骤 3：（仅限 Amazon S3）将备份和导出文件传输到您的目标数据库实例

如果您使用 Amazon S3 暂存表空间备份和 Data Pump 导出文件，请完成以下步骤。

步骤 3.1：将备份和转储文件从源主机上传到您的 Amazon S3 桶

将您的备份和转储文件从源主机上传到 Amazon S3 桶。有关更多信息，请参阅《Amazon Simple Storage Service 开发人员指南》中的上传对象。

步骤 3.2：将备份和转储文件从 Amazon S3 桶下载到目标数据库实例

在此步骤中，您将使用过程 `rdsadmin.rdsadmin_s3_tasks.download_from_s3` 将备份和转储文件下载到您的 RDS for Oracle 数据库实例。按照步骤 2.3：将备份从 Amazon S3 桶下载到目标数据库实例中的步骤操作。

步骤 4：在目标数据库实例上导入表空间

使用过程 `rdsadmin.rdsadmin_transport_util.import_xtts_tablespaces` 还原表空间。有关此过程的语法和语义，请参阅将传输的表空间导入到您的数据库实例

> **Important**
>
> 完成最终表空间导入后，下一步是导入 Oracle Data Pump 元数据。如果导入失败，请务必将数据库实例恢复到失败前的状态。因此，建议您按照为 Amazon RDS 的单可用区数据库实例

创建数据库快照中的说明创建数据库实例的数据库快照。快照将包含所有导入的表空间，因此，如果导入失败，则无需重复备份和导入过程。
如果您的目标数据库实例已开启自动备份，并且 Amazon RDS 在您导入元数据之前未检测到已启动有效的快照，则 RDS 会尝试创建快照。根据您的实例活动，此快照可能会成功，也可能不成功。如果未检测到有效的快照或无法启动快照，则元数据导入将退出并显示错误。

在目标数据库实例上导入表空间

1. 启动 Oracle SQL 客户端，并以主用户身份登录到目标 RDS for Oracle 数据库实例。
2. 运行过程 `rdsadmin.rdsadmin_transport_util.import_xtts_tablespaces`，同时指定要导入的表空间和包含备份的目录。

   以下示例从 *DATA_PUMP_DIR* 目录中导入表空间 *TBS1*、*TBS2* 和 *TBS3*。

```
BEGIN
  :task_id:=rdsadmin.rdsadmin_transport_util.import_xtts_tablespaces('TBS1,TBS2,TBS3','DATA_
END;
/
PRINT task_id
```

3. （可选）通过查询表 `rdsadmin.rds_xtts_operation_info` 来监控进度。`xtts_operation_state` 列显示值 `EXECUTING`、`COMPLETED` 或 `FAILED`。

```
SELECT * FROM rdsadmin.rds_xtts_operation_info;
```

   Note

   对于长时间运行的操作，也可以查询 `V$SESSION_LONGOPS`、`V$RMAN_STATUS` 和 `V$RMAN_OUTPUT`。

4. 使用上一步中的任务 ID 查看已完成的导入的日志。

```
SELECT * FROM TABLE(rdsadmin.rds_file_util.read_text_file('BDUMP',
 'dbtask-'||'&task_id'||'.log'));
```

   在继续下一步之前，请确保导入成功。

5. 按照为 Amazon RDS 的单可用区数据库实例创建数据库快照中的说明制作手动数据库快照。

步骤 5：在目标数据库实例上导入表空间元数据

在此步骤中，使用过程 `rdsadmin.rdsadmin_transport_util.import_xtts_metadata` 将可传输的表空间元数据导入您的 RDS for Oracle 数据库实例。有关此过程的语法和语义，请参阅将可传输的表空间元数据导入到数据库实例中。在操作期间，导入的状态显示在表 `rdsadmin.rds_xtts_operation_info` 中。

> **Important**
>
> 在导入元数据之前，我们强烈建议您确认在导入表空间后已成功创建数据库快照。如果导入步骤失败，请还原数据库实例，解决导入错误，然后再次尝试导入。

将 Data Pump 元数据导入您的 RDS for Oracle 数据库实例

1. 启动 Oracle SQL 客户端，并以主用户身份登录到目标数据库实例。
2. 在传输的表空间中创建拥有模式的用户（如果这些用户尚不存在）。

   ```
   CREATE USER tbs_owner IDENTIFIED BY password;
   ```

3. 导入元数据，同时指定转储文件的名称及其目录位置。

   ```
   BEGIN
     rdsadmin.rdsadmin_transport_util.import_xtts_metadata('xttdump.dmp','DATA_PUMP_DIR');
   END;
   /
   ```

4. （可选）查询可传输的表空间历史记录表以查看元数据导入的状态。

   ```
   SELECT * FROM rdsadmin.rds_xtts_operation_info;
   ```

   操作完成后，您的表空间将处于只读模式。
5. （可选）查看日志文件。

   以下示例列出了 BDUMP 目录的内容，然后查询导入日志。

```
SELECT * FROM TABLE(rdsadmin.rds_file_util.listdir(p_directory => 'BDUMP'));

SELECT * FROM TABLE(rdsadmin.rds_file_util.read_text_file(
  p_directory => 'BDUMP',
  p_filename => 'rds-xtts-
import_xtts_metadata-2023-05-22.01-52-35.560858000.log'));
```

第 5 阶段：验证传输的表空间

在此可选步骤中，使用过程 `rdsadmin.rdsadmin_rman_util.validate_tablespace` 验证传输的表空间，然后将表空间置于读/写模式。

验证传输的数据

1. 启动 SQL*Plus 或 SQL Developer，并以主用户身份登录到目标数据库实例。
2. 使用过程 `rdsadmin.rdsadmin_rman_util.validate_tablespace` 验证表空间。

```
SET SERVEROUTPUT ON
BEGIN
    rdsadmin.rdsadmin_rman_util.validate_tablespace(
        p_tablespace_name     => 'TBS1',
        p_validation_type     => 'PHYSICAL+LOGICAL',
        p_rman_to_dbms_output => TRUE);
    rdsadmin.rdsadmin_rman_util.validate_tablespace(
        p_tablespace_name     => 'TBS2',
        p_validation_type     => 'PHYSICAL+LOGICAL',
        p_rman_to_dbms_output => TRUE);
    rdsadmin.rdsadmin_rman_util.validate_tablespace(
        p_tablespace_name     => 'TBS3',
        p_validation_type     => 'PHYSICAL+LOGICAL',
        p_rman_to_dbms_output => TRUE);
END;
/
```

3. 将表空间置于读/写模式。

```
ALTER TABLESPACE TBS1 READ WRITE;
ALTER TABLESPACE TBS2 READ WRITE;
ALTER TABLESPACE TBS3 READ WRITE;
```

第 6 阶段：清理剩余文件

在此可选步骤中，您将删除所有不需要的文件。使用 rdsadmin.rdsadmin_transport_util.list_xtts_orphan_files 过程列出表空间导入后孤立的数据文件，然后使用 rdsadmin.rdsadmin_transport_util.list_xtts_orphan_files 过程删除它们。有关这些过程的语法和语义，请参阅列出表空间导入后的孤立文件和表空间导入后的孤立数据文件。

清理剩余文件

1. 按如下方式删除 *DATA_PUMP_DIR* 中的旧备份：

   a. 通过运行 rdsadmin.rdsadmin_file_util.listdir 列出备份文件。

   ```
   SELECT * FROM TABLE(rdsadmin.rds_file_util.listdir(p_directory =>
    'DATA_PUMP_DIR'));
   ```

   b. 通过调用 UTL_FILE.FREMOVE 逐一删除备份。

   ```
   EXEC UTL_FILE.FREMOVE ('DATA_PUMP_DIR', 'backup_filename');
   ```

2. 如果您导入了表空间，但没有为这些表空间导入元数据，则可以按如下方式删除孤立的数据文件：

   a. 列出需要删除的孤立数据文件。以下示例运行过程 rdsadmin.rdsadmin_transport_util.list_xtts_orphan_files。

   ```
   SQL> SELECT * FROM
    TABLE(rdsadmin.rdsadmin_transport_util.list_xtts_orphan_files);

   FILENAME       FILESIZE
   -------------- ---------
   datafile_7.dbf 104865792
   datafile_8.dbf 104865792
   ```

   b. 通过运行过程 rdsadmin.rdsadmin_transport_util.cleanup_incomplete_xtts_import 删除孤立文件。

   ```
   BEGIN

    rdsadmin.rdsadmin_transport_util.cleanup_incomplete_xtts_import('DATA_PUMP_DIR');
   ```

```
END;
/
```

清理操作会在 BDUMP 目录中生成一个使用名称格式 rds-xtts-delete_xtts_orphaned_files-*YYYY-MM-DD.HH24-MI-SS.FF*.log 的日志文件。

c. 读取上一步中生成的日志文件。以下示例读取日志 rds-xtts-delete_xtts_orphaned_files-*2023-06-01.09-33-11.868894000*.log。

```
SELECT *
FROM TABLE(rdsadmin.rds_file_util.read_text_file(
       p_directory => 'BDUMP',
       p_filename  => 'rds-xtts-
delete_xtts_orphaned_files-2023-06-01.09-33-11.868894000.log'));

TEXT
--------------------------------------------------------------------------------
orphan transported datafile datafile_7.dbf deleted.
orphan transported datafile datafile_8.dbf deleted.
```

3. 如果您导入了表空间并为这些表空间导入了元数据，但遇到了兼容性错误或其他 Oracle Data Pump 问题，请按如下方式清理部分传输的数据文件：

   a. 通过查询 DBA_TABLESPACES 列出包含部分传输的数据文件的表空间。

```
SQL> SELECT TABLESPACE_NAME FROM DBA_TABLESPACES WHERE PLUGGED_IN='YES';

TABLESPACE_NAME
--------------------------------------------------------------------------------
TBS_3
```

   b. 删除表空间和部分传输的数据文件。

```
DROP TABLESPACE TBS_3 INCLUDING CONTENTS AND DATAFILES;
```

使用 Oracle Data Pump 导入

Oracle Data Pump 是一个实用程序，可让您将 Oracle 数据导出到转储文件并将其导入到另一个 Oracle 数据库。长期来看，它最终会取代 Oracle Export/Import 实用程序。Oracle Data Pump 是将大量数据从 Oracle 数据库迁移到 Amazon RDS 数据库实例的建议方法。

本节中的示例展示了将数据导入 Oracle 数据库的一种方法，但 Oracle Data Pump 支持其他方法。有关更多信息，请参阅 Oracle Database 文档。

本节中的示例使用 DBMS_DATAPUMP 包。您可以使用 Oracle Data Pump 命令行实用程序 impdp 和 expdp 完成相同的任务。您可以在远程主机上安装这些实用程序以作为 Oracle 客户端安装的一部分，包括 Oracle 即时客户端。有关更多信息，请参阅如何使用 Oracle Instant Client 为我的 Amazon RDS for Oracle 数据库实例运行 Data Pump 导入或导出？

主题

- Oracle Data Pump 概述
- 使用 Oracle Data Pump 和 Amazon S3 存储桶导入数据
- 使用 Oracle Data Pump 和数据库链接导入数据

Oracle Data Pump 概述

Oracle Data Pump 由以下组件组成：

- 命令行客户端 expdp 和 impdp
- DBMS_DATAPUMP PL/SQL 程序包
- DBMS_METADATA PL/SQL 程序包

您可以将 Oracle Data Pump 用于以下场景：

- 将数据从 Oracle 数据库（可以是本地或在 Amazon EC2 实例上）导入到 RDS for Oracle 数据库实例。
- 将数据从 RDS for Oracle 数据库实例导入到 Oracle 数据库（可以是本地或在 Amazon EC2 实例上）。
- 在 RDS for Oracle 数据库实例之间导入数据，例如将数据从 EC2-Classic 迁移到 VPC。

要下载 Oracle Data Pump 实用工具，请参阅 Oracle Technology Network 网站上的 Oracle 数据库软件下载。有关在 Oracle Database 的版本之间进行迁移时的兼容性注意事项，请参阅 Oracle Database 文档。

Oracle Data Pump 工作流

通常，您在以下阶段使用 Oracle Data Pump：

1. 将数据导出到源数据库上的转储文件。
2. 将转储文件上载到目标 RDS for Oracle 数据库实例。您可以使用 Amazon S3 桶或使用两个数据库之间的数据库链接进行传输。
3. 将数据从转储文件中导入到 RDS for Oracle 数据库实例。

Oracle Data Pump 最佳实践

当您使用 Oracle Data Pump 将数据导入到 RDS for Oracle 实例时，我们建议使用以下最佳实践：

- 在 `schema` 或 `table` 模式中执行导入，以便导入特定架构和对象。
- 将您导入的架构限制为您的应用程序所需的架构。
- 请勿在 `full` 模式下导入，也不要导入系统维护的组件的架构。

  因为 RDS for Oracle 不允许 `SYS` 或 `SYSDBA` 管理用户进行访问，所以，这些操作可能会损坏 Oracle 数据词典并影响数据库的稳定性。
- 加载大量数据时，请执行以下操作：
  1. 将转储文件转移到目标 RDS for Oracle 数据库实例。
  2. 为您的数据库实例拍摄快照。
  3. 测试导入以验证导入是否成功。

  如果数据库组件失效，则您可以删除数据库实例并通过数据库快照重新创建它。还原的数据库实例包括当您拍摄数据库快照时暂存在数据库实例上的任何转储文件。
- 请勿导入使用 Oracle Data Pump 导出参数 `TRANSPORT_TABLESPACES`、`TRANSPORTABLE`、或 `TRANSPORT_FULL_CHECK` 创建的转储文件。RDS for Oracle 数据库实例不支持导入这些转储文件。
- 不要导入包含 Oracle 计划程序对象的转储文件 `SYS`、`SYSTEM`、`RDSADMIN`、`RDSSEC` 和 `RDS_DATAGUARD`，以及属于以下类别的文件：
  - 作业
  - 计划
  - Schedules
  - 链
  - 规则
  - 评估背景
  - 规则集

RDS for Oracle 数据库实例不支持导入这些转储文件。

- 要排除不受支持的 Oracle 计划程序对象，请在 Data Pump 导出期间使用其他指令。如果您使用 DBMS_DATAPUMP，则可以在 METADATA_FILTER 之前添加额外的 DBMS_METADATA.START_JOB：

```
DBMS_DATAPUMP.METADATA_FILTER(
  v_hdnl,
  'EXCLUDE_NAME_EXPR',
  q'[IN (SELECT NAME FROM SYS.OBJ$
         WHERE TYPE# IN (66,67,74,79,59,62,46)
         AND OWNER# IN
           (SELECT USER# FROM SYS.USER$
            WHERE NAME IN ('RDSADMIN','SYS','SYSTEM','RDS_DATAGUARD','RDSSEC')
            )
        )
  ]',
  'PROCOBJ'
);
```

如果使用 expdp，请创建包含以下示例中所示的 exclude 指令的参数文件。然后，将 PARFILE=*parameter_file* 与您的 expdp 命令一起使用。

```
exclude=procobj:"IN
  (SELECT NAME FROM sys.OBJ$
   WHERE TYPE# IN (66,67,74,79,59,62,46)
   AND OWNER# IN
     (SELECT USER# FROM SYS.USER$
      WHERE NAME IN ('RDSADMIN','SYS','SYSTEM','RDS_DATAGUARD','RDSSEC')
     )
  )"
```

使用 Oracle Data Pump 和 Amazon S3 存储桶导入数据

以下导入过程使用 Oracle Data Pump 和 Amazon S3 存储桶。步骤如下所示：

1. 使用 Oracle DBMS_DATAPUMP 程序包导出源数据库上的数据。
2. 将转储文件置于 Amazon S3 桶中。
3. 将转储文件从 Amazon S3 桶下载到目标 RDS for Oracle 数据库实例上的 DATA_PUMP_DIR 目录。

4. 使用 DBMS_DATAPUMP 程序包，将数据从复制的转储文件导入到 RDS for Oracle 数据库实例。

主题

使用 Oracle Data Pump 和 Amazon S3 桶导入数据的要求

此过程有以下要求：

- 确保 Amazon S3 桶可用于文件传输，并且 Amazon S3 桶与数据库实例处于相同的 Amazon Web Services 区域中。有关说明，请参阅 Amazon Simple Storage Service 入门指南 中的创建存储桶。
- 您上传到 Amazon S3 存储桶的对象不得超过 5 TB。有关在 Amazon S3 中使用对象的更多信息，请参阅 Amazon Simple Storage Service 用户指南。

> **Note**
>
> 如果要转储超过 5 TB 的文件，则可以使用并行选项来运行 Oracle Data Pump 导出。此操作会将数据分散到多个转储文件中，以便不会超过单个文件的 5 TB 限制。

- 您必须按照 为与 Amazon S3 集成的 RDS for Oracle 配置 IAM 权限 中的说明为 Amazon RDS 集成准备 Amazon S3 存储桶。
- 您必须确保有足够的存储空间来存储源实例和目标数据库实例上的转储文件。

> **Note**
>
> 此过程将转储文件导入到 DATA_PUMP_DIR 目录（所有 Oracle 数据库实例上的预配置目录）中。此目录位于您的数据文件所在的存储卷上。当您导入转储文件时，现有 Oracle 文件使用更多空间。因此，您应该确保您的数据库实例可适应这个额外的空间使用。导入的转储文件不会

自动从 DATA_PUMP_DIR 目录中删除或清除。要删除导入的转储文件，请使用 Oracle 网站上的 UTL_FILE.FREMOVE。

步骤 1：在 RDS for Oracle 目标数据库实例上向数据库用户授予权限

在此步骤中，您将创建您计划向其中导入数据的架构，并向用户授予必要的权限。

在 RDS for Oracle 目标实例上创建用户并授予必要的权限

1. 使用 SQL*Plus 或 Oracle SQL Developer，作为主用户登录到将向其中导入数据的 RDS for Oracle 数据库实例。有关连接到数据库实例的信息，请参阅 连接到 Oracle 数据库实例。
2. 在导入数据之前，创建所需的表空间。有关更多信息，请参阅 创建表空间并配置其大小。
3. 如果要向其中导入数据的用户账户不存在，请创建用户账户并授予必需的权限和角色。如果您打算将数据导入到多个用户架构，请创建各个用户账户并向其授予所需的权限和角色。

   例如，以下 SQL 语句创建新的用户并授予所需权限和角色，以将数据导入到此用户拥有的架构中。将 *schema_1* 替换为此步骤中和下列步骤中架构的名称。

   ```
   CREATE USER schema_1 IDENTIFIED BY my_password;
   GRANT CREATE SESSION, RESOURCE TO schema_1;
   ALTER USER schema_1 QUOTA 100M ON users;
   ```

   > **Note**
   > 作为安全最佳实践，请指定除此处所示提示以外的密码。

   前面的语句向新用户授予 CREATE SESSION 权限和 RESOURCE 角色。根据导入的数据库对象，您可能需要其他权限和角色。

步骤 2：使用 DBMS_DATAPUMP 将数据导出到转储文件中

要创建转储文件，请使用 DBMS_DATAPUMP 程序包。

将 Oracle 数据导出到转储文件

1. 使用 SQL Plus 或 Oracle SQL Developer，以管理用户身份连接到源 RDS for Oracle 数据库实例。如果源数据库是 RDS for Oracle 数据库实例，请使用 Amazon RDS 主用户身份进行连接。

2. 通过调用 DBMS_DATAPUMP 过程导出数据。

   以下脚本将 *SCHEMA_1* 架构导出到 DATA_PUMP_DIR 目录中名为 sample.dmp 的转储文件中。将 *SCHEMA_1* 替换为要导出的架构的名称。

```
DECLARE
  v_hdnl NUMBER;
BEGIN
  v_hdnl := DBMS_DATAPUMP.OPEN(
    operation => 'EXPORT',
    job_mode  => 'SCHEMA',
    job_name  => null
  );
  DBMS_DATAPUMP.ADD_FILE(
    handle    => v_hdnl         ,
    filename  => 'sample.dmp'   ,
    directory => 'DATA_PUMP_DIR',
    filetype  => dbms_datapump.ku$_file_type_dump_file
  );
  DBMS_DATAPUMP.ADD_FILE(
    handle    => v_hdnl,
    filename  => 'sample_exp.log',
    directory => 'DATA_PUMP_DIR' ,
    filetype  => dbms_datapump.ku$_file_type_log_file
  );
  DBMS_DATAPUMP.METADATA_FILTER(v_hdnl,'SCHEMA_EXPR','IN (''SCHEMA_1'')');
  DBMS_DATAPUMP.METADATA_FILTER(
    v_hdnl,
    'EXCLUDE_NAME_EXPR',
    q'[IN (SELECT NAME FROM SYS.OBJ$
           WHERE TYPE# IN (66,67,74,79,59,62,46)
           AND OWNER# IN
             (SELECT USER# FROM SYS.USER$
              WHERE NAME IN ('RDSADMIN','SYS','SYSTEM','RDS_DATAGUARD','RDSSEC')
             )
          )
    ]',
    'PROCOBJ'
  );
  DBMS_DATAPUMP.START_JOB(v_hdnl);
END;
/
```

> **Note**
>
> Data Pump 异步启动任务。有关监控 Data Pump 作业的信息，请参阅 Oracle 文档中的监控作业状态。

3. （可选）通过调用 `rdsadmin.rds_file_util.read_text_file` 过程查看导出日志的内容。有关更多信息，请参阅 读取数据库实例目录中的文件。

步骤 3：将转储文件上传到 Amazon S3 存储桶。

使用 Amazon RDS 过程 `rdsadmin.rdsadmin_s3_tasks.upload_to_s3` 将转储文件复制到 Amazon S3 存储桶中。以下示例将 `DATA_PUMP_DIR` 目录中的所有文件上传到名为 *amzn-s3-demo-bucket* 的 Amazon S3 存储桶中。

```
SELECT rdsadmin.rdsadmin_s3_tasks.upload_to_s3(
  p_bucket_name    =>  'amzn-s3-demo-bucket',
  p_directory_name =>  'DATA_PUMP_DIR')
AS TASK_ID FROM DUAL;
```

`SELECT` 语句返回 `VARCHAR2` 数据类型的任务 ID。有关更多信息，请参阅 将文件从 RDS for Oracle 数据库实例上传到 Amazon S3 存储桶。

步骤 4：将转储文件从 Amazon S3 桶下载到目标数据库实例

使用 Amazon RDS 过程 `rdsadmin.rdsadmin_s3_tasks.download_from_s3` 执行此步骤。将文件下载到目录时，如果目录中已存在同名文件，则过程 `download_from_s3` 将跳过下载。要从下载目录中删除文件，请使用 Oracle 网站上的 UTL_FILE.FREMOVE。

下载转储文件

1. 启动 SQL*Plus 或 Oracle Developer 并以主用户身份登录 Amazon RDS 目标 Oracle 数据库实例
2. 使用 Amazon RDS 过程 `rdsadmin.rdsadmin_s3_tasks.download_from_s3` 下载转储文件。

   以下示例将所有文件从名为 *amzn-s3-demo-bucket* 的 Amazon S3 桶下载到 `DATA_PUMP_DIR` 目录中。

   ```
   SELECT rdsadmin.rdsadmin_s3_tasks.download_from_s3(
   ```

```
  p_bucket_name    =>  'amzn-s3-demo-bucket',
  p_directory_name =>  'DATA_PUMP_DIR')
AS TASK_ID FROM DUAL;
```

SELECT 语句返回 VARCHAR2 数据类型的任务 ID。有关更多信息，请参阅 将文件从 Amazon S3 存储桶下载到 Oracle 数据库实例。

步骤 5：使用 DBMS_DATAPUMP 将转储文件导入到目标数据库实例

使用 DBMS_DATAPUMP 将架构导入到 RDS for Oracle 数据库实例。可能需要额外的选项（例如 METADATA_REMAP）。

将数据导入到目标数据库实例

1. 启动 SQL*Plus 或 SQL Developer 并以主用户身份登录 RDS for Oracle 数据库实例。
2. 通过调用 DBMS_DATAPUMP 过程导入数据。

   下面的示例将 *SCHEMA_1* 数据从 sample_copied.dmp 导入到目标数据库实例。

```
DECLARE
  v_hdnl NUMBER;
BEGIN
  v_hdnl := DBMS_DATAPUMP.OPEN(
    operation => 'IMPORT',
    job_mode  => 'SCHEMA',
    job_name  => null);
  DBMS_DATAPUMP.ADD_FILE(
    handle    => v_hdnl,
    filename  => 'sample_copied.dmp',
    directory => 'DATA_PUMP_DIR',
    filetype  => dbms_datapump.ku$_file_type_dump_file);
  DBMS_DATAPUMP.ADD_FILE(
    handle    => v_hdnl,
    filename  => 'sample_imp.log',
    directory => 'DATA_PUMP_DIR',
    filetype  => dbms_datapump.ku$_file_type_log_file);
  DBMS_DATAPUMP.METADATA_FILTER(v_hdnl,'SCHEMA_EXPR','IN (''SCHEMA_1'')');
  DBMS_DATAPUMP.START_JOB(v_hdnl);
END;
/
```

> **Note**
>
> 数据抽取任务将异步开始。有关监控 Data Pump 作业的信息，请参阅 Oracle 文档中的监控作业状态。您可以使用 `rdsadmin.rds_file_util.read_text_file` 过程查看导入日志的内容。有关更多信息，请参阅 读取数据库实例目录中的文件。

3. 通过列出目标数据库实例上的架构表来验证数据导入。

   例如，以下查询将返回 *SCHEMA_1* 的表数量。

   ```
   SELECT COUNT(*) FROM DBA_TABLES WHERE OWNER='SCHEMA_1';
   ```

步骤 6：清除

导入数据后，可以删除不想保留的文件。

删除不需要的文件

1. 启动 SQL*Plus 或 SQL Developer 并以主用户身份登录 RDS for Oracle 数据库实例。
2. 使用以下命令列出 `DATA_PUMP_DIR` 中的文件。

   ```
   SELECT * FROM TABLE(rdsadmin.rds_file_util.listdir('DATA_PUMP_DIR')) ORDER BY
    MTIME;
   ```

3. 要删除 `DATA_PUMP_DIR` 中不再需要的文件，请使用以下命令。

   ```
   EXEC UTL_FILE.FREMOVE('DATA_PUMP_DIR','filename');
   ```

   例如，以下命令删除名为 `sample_copied.dmp` 的文件

   ```
   EXEC UTL_FILE.FREMOVE('DATA_PUMP_DIR','sample_copied.dmp');
   ```

使用 Oracle Data Pump 和数据库链接导入数据

以下导入过程使用 Oracle Data Pump 和 Oracle DBMS_FILE_TRANSFER 包。步骤如下所示：

1. 连接到源 Oracle 数据库，此数据库可以是本地数据库、Amazon EC2 实例 或 RDS for Oracle 数据库实例。
2. 使用 DBMS_DATAPUMP 程序包导出数据。
3. 使用 `DBMS_FILE_TRANSFER.PUT_FILE` 将转储文件从 Oracle 数据库复制到通过数据库链接连接到的目标 RDS for Oracle 数据库实例上的 `DATA_PUMP_DIR` 目录中。
4. 使用 `DBMS_DATAPUMP` 程序包，将数据从复制的转储文件导入到 RDS for Oracle 数据库实例。

使用 Oracle 数据转储和 `DBMS_FILE_TRANSFER` 包的导入过程具有以下步骤。

主题

使用 Oracle Data Pump 和数据库链接导入数据的要求

此过程有以下要求：

- 您必须对 `DBMS_FILE_TRANSFER` 和 `DBMS_DATAPUMP` 包具有执行权限。
- 您必须对源数据库实例上的 `DATA_PUMP_DIR` 目录具有写入权限。
- 您必须确保有足够的存储空间来存储源实例和目标数据库实例上的转储文件。

> **Note**
>
> 此过程将转储文件导入到 `DATA_PUMP_DIR` 目录（所有 Oracle 数据库实例上的预配置目录）中。此目录位于您的数据文件所在的存储卷上。当您导入转储文件时，现有 Oracle 文件使用更多空间。因此，您应该确保您的数据库实例可适应这个额外的空间使用。导入的转储文件不会

自动从 DATA_PUMP_DIR 目录中删除或清除。要删除导入的转储文件，请使用 Oracle 网站上的 UTL_FILE.FREMOVE。

步骤 1：在 RDS for Oracle 目标数据库实例上向用户授予权限

要向用户授予对 RDS for Oracle 目标数据库实例的权限，请执行以下步骤：

1. 使用 SQL Plus 或 Oracle SQL Developer 连接到要向其中导入数据的 RDS for Oracle 数据库实例。以 Amazon RDS 主用户的身份连接。有关连接到数据库实例的信息，请参阅连接到 Oracle 数据库实例。
2. 在导入数据之前，创建所需的表空间。有关更多信息，请参阅“创建表空间并配置其大小”。
3. 如果要导入数据的用户账户不存在，请创建用户账户并授予必需的权限和角色。如果您打算将数据导入到多个用户架构，请创建各个用户账户并向其授予所需的权限和角色。

   例如，以下命令创建名为 *schema_1* 的新用户，并向此用户授予所需权限和角色，以将数据导入到用户架构中。

   ```
   CREATE USER schema_1 IDENTIFIED BY my-password;
   GRANT CREATE SESSION, RESOURCE TO schema_1;
   ALTER USER schema_1 QUOTA 100M ON users;
   ```

   > Note
   >
   > 作为安全最佳实践，请指定除此处所示提示以外的密码。

   前一个示例为新用户授予 CREATE SESSION 权限和 RESOURCE 角色。根据导入的数据库对象，可能需要其他权限和角色。

   > Note
   >
   > 将 *schema_1* 替换为此步骤中和下列步骤中架构的名称。

步骤 2：向用户授予源数据库权限

使用 SQL*Plus 或 Oracle SQL Developer 连接到包含要导入的数据的 RDS for Oracle 数据库实例。如有必要，可创建用户账户并授予必要权限。

> **Note**
>
> 如果源数据库是 Amazon RDS 实例，则可以跳过此步骤。使用您的 Amazon RDS 主用户账户来执行导出。

以下命令创建新用户并授予必要权限。

```
CREATE USER export_user IDENTIFIED BY my-password;
GRANT CREATE SESSION, CREATE TABLE, CREATE DATABASE LINK TO export_user;
ALTER USER export_user QUOTA 100M ON users;
GRANT READ, WRITE ON DIRECTORY data_pump_dir TO export_user;
GRANT SELECT_CATALOG_ROLE TO export_user;
GRANT EXECUTE ON DBMS_DATAPUMP TO export_user;
GRANT EXECUTE ON DBMS_FILE_TRANSFER TO export_user;
```

> **Note**
>
> 作为安全最佳实践，请指定除此处所示提示以外的密码。

步骤 3：使用 DBMS_DATAPUMP 创建转储文件

要创建转储文件，请执行以下操作：

1. 使用 SQL*Plus 或 Oracle SQL Developer，以管理用户或在步骤 2 中创建的用户身份连接到源 Oracle 实例。如果源数据库是 Amazon RDS for Oracle 数据库实例，请使用 Amazon RDS 主用户身份连接。
2. 使用 Oracle Data Pump 实用程序创建转储文件。

   以下脚本在 DATA_PUMP_DIR 目录中创建一个名为 sample.dmp 的转储文件。

   ```
   DECLARE
     v_hdnl NUMBER;
   BEGIN
     v_hdnl := DBMS_DATAPUMP.OPEN(
   ```

```
    operation => 'EXPORT' ,
    job_mode  => 'SCHEMA' ,
    job_name  => null
  );
  DBMS_DATAPUMP.ADD_FILE(
    handle    => v_hdnl,
    filename  => 'sample.dmp'    ,
    directory => 'DATA_PUMP_DIR' ,
    filetype  => dbms_datapump.ku$_file_type_dump_file
  );
  DBMS_DATAPUMP.ADD_FILE(
    handle    => v_hdnl           ,
    filename  => 'sample_exp.log' ,
    directory => 'DATA_PUMP_DIR'  ,
    filetype  => dbms_datapump.ku$_file_type_log_file
  );
  DBMS_DATAPUMP.METADATA_FILTER(
    v_hdnl              ,
    'SCHEMA_EXPR'       ,
    'IN (''SCHEMA_1'')'
  );
  DBMS_DATAPUMP.METADATA_FILTER(
    v_hdnl,
    'EXCLUDE_NAME_EXPR',
    q'[IN (SELECT NAME FROM sys.OBJ$
           WHERE TYPE# IN (66,67,74,79,59,62,46)
           AND OWNER# IN
             (SELECT USER# FROM SYS.USER$
              WHERE NAME IN ('RDSADMIN','SYS','SYSTEM','RDS_DATAGUARD','RDSSEC')
             )
          )
    ]',
    'PROCOBJ'
  );
  DBMS_DATAPUMP.START_JOB(v_hdnl);
END;
/
```

> **Note**
>
> 数据抽取任务将异步开始。有关监控 Data Pump 作业的信息，请参阅 Oracle 文档中的监控作业状态。您可以使用 `rdsadmin.rds_file_util.read_text_file` 过程查看导出日志的内容。有关更多信息，请参阅“读取数据库实例目录中的文件”。

步骤 4：创建目标数据库实例的数据库链接

在源数据库实例和目标数据库实例之间创建数据库链接。您的本地 Oracle 实例必须具有至数据库实例的网络连接，然后才能创建数据库链接以及传输导出转储文件。

执行此步骤，使用与上一步中相同的用户账户连接。

如果您将在同一 VPC 或对等 VPC 中的两个数据库实例之间创建数据库链接，则这两个数据库实例之间应具有有效路由。每个数据库实例的安全组必须允许另一个数据库实例的传入和传出。安全组入站和出站规则可引用同一 VPC 或对等 VPC 中的安全组。有关更多信息，请参阅“在 VPC 中调整用于数据库实例的数据库链接”。

以下命令创建名为 `to_rds` 的数据库链接，连接到位于目标数据库实例中的 Amazon RDS 主用户。

```
CREATE DATABASE LINK to_rds
  CONNECT TO <master_user_account> IDENTIFIED BY <password>
  USING '(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=<dns or ip address of remote db>)
         (PORT=<listener port>))(CONNECT_DATA=(SID=<remote SID>)))';
```

步骤 5：使用 DBMS_FILE_TRANSFER 将导出的转储文件复制到目标数据库实例

使用 `DBMS_FILE_TRANSFER` 将转储文件从源数据库实例复制到目标数据库实例。以下脚本将名为 sample.dmp 的转储文件从源实例复制到名为 to_rds 的目标数据库链接（已在上一步中创建）。

```
BEGIN
  DBMS_FILE_TRANSFER.PUT_FILE(
    source_directory_object       => 'DATA_PUMP_DIR',
    source_file_name              => 'sample.dmp',
    destination_directory_object  => 'DATA_PUMP_DIR',
    destination_file_name         => 'sample_copied.dmp',
    destination_database          => 'to_rds' );
END;
/
```

步骤 6：使用 DBMS_DATAPUMP 将数据文件导入到目标数据库实例

在数据库实例中使用 Oracle Data Pump 导入架构。可能需要额外选项（例如，METADATA_REMAP）。

使用 Amazon RDS 主用户账户连接到数据库实例来执行导入。

```
DECLARE
  v_hdnl NUMBER;
BEGIN
  v_hdnl := DBMS_DATAPUMP.OPEN(
    operation => 'IMPORT',
    job_mode  => 'SCHEMA',
    job_name  => null);
  DBMS_DATAPUMP.ADD_FILE(
    handle    => v_hdnl,
    filename  => 'sample_copied.dmp',
    directory => 'DATA_PUMP_DIR',
    filetype  => dbms_datapump.ku$_file_type_dump_file );
  DBMS_DATAPUMP.ADD_FILE(
    handle    => v_hdnl,
    filename  => 'sample_imp.log',
    directory => 'DATA_PUMP_DIR',
    filetype  => dbms_datapump.ku$_file_type_log_file);
  DBMS_DATAPUMP.METADATA_FILTER(v_hdnl,'SCHEMA_EXPR','IN (''SCHEMA_1'')');
  DBMS_DATAPUMP.START_JOB(v_hdnl);
END;
/
```

> **Note**
>
> 数据抽取任务将异步开始。有关监控 Data Pump 作业的信息，请参阅 Oracle 文档中的监控作业状态。您可以使用 `rdsadmin.rds_file_util.read_text_file` 过程查看导入日志的内容。有关更多信息，请参阅“读取数据库实例目录中的文件”。

您可以通过查看数据库实例上用户的表来验证数据导入。例如，以下查询将返回 *schema_1* 的表数量。

```
SELECT COUNT(*) FROM DBA_TABLES WHERE OWNER='SCHEMA_1';
```

步骤 7：清除

导入数据后，可以删除不想保留的文件。您可以使用以下命令列出 DATA_PUMP_DIR 中的文件。

```
SELECT * FROM TABLE(rdsadmin.rds_file_util.listdir('DATA_PUMP_DIR')) ORDER BY MTIME;
```

要删除 DATA_PUMP_DIR 中不再需要的文件，请使用以下命令。

```
EXEC UTL_FILE.FREMOVE('DATA_PUMP_DIR','<file name>');
```

例如，以下命令删除名为 "sample_copied.dmp" 的文件

```
EXEC UTL_FILE.FREMOVE('DATA_PUMP_DIR','sample_copied.dmp');
```

使用 Oracle Export/Import 进行导入

在以下情况下，您可以考虑使用 Oracle Export/Import 实用程序进行迁移：

- 您的数据量很小。
- 不需要二进制浮点数和双精度等数据类型。

导入过程会创建必要的模式对象。因此，无需事先运行脚本来创建对象。

安装 Oracle 导出和导入实用程序的最简单方法是安装 Oracle Instant Client。要下载该软件，请前往 https://www.oracle.com/database/technologies/instant-client.html。有关文档，请参阅《Oracle 数据库实用程序》手册中的适用于 SQL*Loader、导出和导入的 Instant Client。

导出表，然后导入它们

1. 使用 exp 命令将表从源数据库中导出。

   以下命令导出名为 tab1、tab2 以及 tab3 的表。转储文件为 exp_file.dmp。

   ```
   exp cust_dba@ORCL FILE=exp_file.dmp TABLES=(tab1,tab2,tab3) LOG=exp_file.log
   ```

   导出会创建一个二进制转储文件，其中包含指定表的架构和数据。

2. 使用 imp 命令将架构和数据导入目标数据库。

   以下命令从转储文件 exp_file.dmp 中导入表 tab1、tab2 和 tab3。

```
imp cust_dba@targetdb FROMUSER=cust_schema TOUSER=cust_schema \
TABLES=(tab1,tab2,tab3) FILE=exp_file.dmp LOG=imp_file.log
```

“导出”和“导入”具有其他可能更适合您的要求的变体。有关完整详细信息，请参阅 Oracle Database 文档。

使用 Oracle SQL*Loader 进行导入

您可以考虑将 Oracle SQL*Loader 用于对象数量有限的大型数据库。由于从源数据库导出并加载到目标数据库的过程特定于架构，因此，下面的示例创建示例模式对象，从源导出，然后将数据加载到目标数据库。

安装 Oracle SQL*Loader 的最简单方法是安装 Oracle Instant Client。要下载该软件，请前往 https://www.oracle.com/database/technologies/instant-client.html。有关文档，请参阅《Oracle 数据库实用程序》手册中的[适用于 SQL*Loader、导出和导入的 Instant Client](#)。

使用 Oracle SQL*Loader 导入数据

1. 使用以下 SQL 语句创建示例源表。

```
CREATE TABLE customer_0 TABLESPACE users
   AS (SELECT ROWNUM id, o.*
       FROM   ALL_OBJECTS o, ALL_OBJECTS x
       WHERE  ROWNUM <= 1000000);
```

2. 在目标 RDS for Oracle 数据库实例上，创建用于加载数据的目标表。子句 `WHERE 1=2` 可确保复制 `ALL_OBJECTS` 的结构，但不复制任何行。

```
CREATE TABLE customer_1 TABLESPACE users
  AS (SELECT 0 AS ID, OWNER, OBJECT_NAME, CREATED
      FROM   ALL_OBJECTS
      WHERE  1=2);
```

3. 将数据从源数据库导出到文本文件。以下示例使用 SQL*Plus。对于数据，您可能需要生成用于执行数据库中所有对象导出操作的脚本。

```
ALTER SESSION SET NLS_DATE_FORMAT = 'YYYY/MM/DD HH24:MI:SS'

SET LINESIZE 800 HEADING OFF FEEDBACK OFF ARRAY 5000 PAGESIZE 0
```

```
SPOOL customer_0.out
SET MARKUP HTML PREFORMAT ON
SET COLSEP ','

SELECT id, owner, object_name, created
FROM   customer_0;

SPOOL OFF
```

4. 创建一个控制文件来描述数据。您可能需要编写脚本来执行此步骤。

```
cat << EOF > sqlldr_1.ctl
load data
infile customer_0.out
into table customer_1
APPEND
fields terminated by "," optionally enclosed by '"'
(
  id          POSITION(01:10)    INTEGER EXTERNAL,
  owner       POSITION(12:41)    CHAR,
  object_name POSITION(43:72)    CHAR,
  created     POSITION(74:92)    date "YYYY/MM/DD HH24:MI:SS"
)
```

   如果需要，将上述代码生成的文件复制到某一暂存区域，如 Amazon EC2 实例。

5. 通过适用于目标数据库的用户名和密码，使用 SQL*Loader 导入数据。

```
sqlldr cust_dba@targetdb CONTROL=sqlldr_1.ctl BINDSIZE=10485760 READSIZE=10485760
 ROWS=1000
```

使用 Oracle 实体化视图进行迁移

要高效率地迁移大型数据集，您可以使用 Oracle 实体化视图复制功能。通过复制，您可以使目标表与源表保持同步。这样，如果需要，您可以稍后切换到 Amazon RDS。

在可以使用实体化视图进行迁移之前，请确保满足以下要求：

- 配置从目标数据库到源数据库的访问权限。在以下示例中，在源数据库上启用了访问规则，以允许 RDS for Oracle 目标数据库通过 SQL*Net 与源数据库连接。
- 创建从 RDS for Oracle 数据库实例到源数据库的数据库链接。

使用实体化视图迁移数据

1. 在源实例和 RDS for Oracle 目标实例上，创建可以使用同一密码进行身份验证的用户账户。下面的示例创建了一个名为 dblink_user 的用户。

```
CREATE USER dblink_user IDENTIFIED BY my-password
  DEFAULT TABLESPACE users
  TEMPORARY TABLESPACE temp;

GRANT CREATE SESSION TO dblink_user;

GRANT SELECT ANY TABLE TO dblink_user;

GRANT SELECT ANY DICTIONARY TO dblink_user;
```

> **Note**
>
> 作为安全最佳实践，请指定除此处所示提示以外的密码。

2. 使用新创建的用户创建从 RDS for Oracle 目标实例到源实例的数据库链接。

```
CREATE DATABASE LINK remote_site
  CONNECT TO dblink_user IDENTIFIED BY my-password
  USING '(description=(address=(protocol=tcp) (host=my-host)
    (port=my-listener-port)) (connect_data=(sid=my-source-db-sid)))';
```

> **Note**
>
> 作为安全最佳实践，请指定除此处所示提示以外的密码。

3. 测试链接：

```
SELECT * FROM V$INSTANCE@remote_site;
```

4. 使用主键和源实例上的具体化视图日志创建示例表。

```
CREATE TABLE customer_0 TABLESPACE users
  AS (SELECT ROWNUM id, o.*
      FROM   ALL_OBJECTS o, ALL_OBJECTS x
      WHERE  ROWNUM <= 1000000);
```

```
ALTER TABLE customer_0 ADD CONSTRAINT pk_customer_0 PRIMARY KEY (id) USING INDEX;

CREATE MATERIALIZED VIEW LOG ON customer_0;
```

5. 在目标 RDS for Oracle 数据库实例上，创建实体化视图。

```
CREATE MATERIALIZED VIEW customer_0
  BUILD IMMEDIATE REFRESH FAST
  AS (SELECT *
      FROM   cust_dba.customer_0@remote_site);
```

6. 在目标 RDS for Oracle 实例上，刷新实体化视图。

```
EXEC DBMS_MVIEW.REFRESH('CUSTOMER_0', 'f');
```

7. 删除实体化视图并包括 `PRESERVE TABLE` 子句，以便保留实体化视图容器表及其内容。

```
DROP MATERIALIZED VIEW customer_0 PRESERVE TABLE;
```

   保留的表拥有与已删除具体化视图相同的名称。

使用 Amazon RDS for Oracle 的只读副本

要在 Oracle 数据库实例之间配置复制，您可以创建副本数据库。有关 Amazon RDS 只读副本的概览，请参阅 Amazon RDS 只读副本概述。有关 Oracle 副本和其他数据库引擎之间的差异摘要，请参阅 数据库引擎的只读副本之间的差异。

主题

RDS for Oracle 副本概述

Oracle 副本数据库是主数据库的物理副本。只读模式下的 Oracle 副本称为只读副本。装载模式下的 Oracle 副本称为装载副本。Oracle 数据库不允许在副本中进行写入操作，但是您可以升级副本以使其可写。提升的只读副本拥有提出提升请求时的已复制数据。

以下视频提供了 RDS for Oracle 灾难恢复的有用概述。

有关更多信息，请参阅博文 Managed disaster recovery with Amazon RDS for Oracle cross-Region automated backups - Part 1 和 Managed disaster recovery with Amazon RDS for Oracle cross-Region automated backups - Part 2。

主题

只读和装载副本

创建或修改 Oracle 副本时，可以将其置于以下任一模式：

Read-only

这是默认模式。Active Data Guard 会将源数据库中的更改传输并应用到所有只读副本数据库。

您可以从一个源数据库实例创建最多 5 个只读副本。有关适用于所有数据库引擎的只读副本的一般信息，请参阅使用数据库实例只读副本。有关 Oracle Data Guard 的信息，请参阅 Oracle 文档中的 Oracle Data Guard 概念和管理。

挂载模式

在这种情况下，复制使用 Oracle Data Guard，但副本数据库不接受用户连接。装载副本的主要用途是跨区域灾难恢复。

装载的副本无法为只读工作负载提供服务。不论存档日志的保留策略为何，装载副本都会在应用存档重做日志文件后删除这些文件。

您可以为同一源数据库实例同时创建装载和只读数据库副本。您可以将只读副本更改为装载模式，或将装载副本更改为只读模式。在两种情况下，Oracle 数据库都会保留存档日志保留设置。

CDB 的只读副本

RDS for Oracle 仅对单租户配置的 Oracle Database 19c 和 21c CDB 支持 Data Guard 只读副本。您可以在 CDB 中创建、管理和提升只读副本，就像在非 CDB 中一样。还支持已装载的副本。您可以获得以下好处：

- 托管式灾难恢复、高可用性和对副本的只读访问
- 能够在不同的 Amazon Web Services 区域中创建只读副本。
- 与现有 RDS 只读副本 API 集成：CreateDBInstanceReadReplica、PromoteReadReplica 和 SwitchoverReadReplica

要使用此功能，您需要有关副本和主数据库实例的有效 Data Guard 许可证和 Oracle 数据库企业版许可证。没有与使用 CDB 架构相关的额外费用。您只需为数据库实例付费。

有关 CDB 架构的单租户配置和多租户配置的更多信息，请参阅RDS for Oracle CDB 概述。

归档重做日志保留

如果主数据库实例没有跨区域只读副本，则 Amazon RDS for Oracle 会在源数据库实例上至少保留两个小时的存档重做日志。无论在 `rdsadmin.rdsadmin_util.set_configuration` 中 `archivelog retention hours` 的设置为何，都是这种情况。

两小时之后或在传递归档日志保留小时数设置之后，RDS 会从源数据库实例中清除日志，以时间较长者为准。传递归档日志保留小时数设置之后，仅当日志已成功应用到数据库时，RDS 才会从只读副本清除日志。

在某些情况下，主数据库实例可能具有一个或多个跨区域只读副本。如果是这样，则 Amazon RDS for Oracle 将保留源数据库实例上的事务日志，直到将它们传输并应用到所有跨区域只读副本为止。有关 `rdsadmin.rdsadmin_util.set_configuration` 的信息，请参阅保留归档重做日志。

Oracle 复制过程中的中断

当您创建只读副本时，Amazon RDS 会制作源数据库实例的数据库快照，然后开始复制。数据库快照操作开始时，源数据库实例会经历非常短暂的 I/O 暂停。这种 I/O 暂停通常持续大约一秒时间。如果源数据库实例为多可用区部署，则可避免 I/O 暂停，因为在这种情况下，会从辅助数据库实例制作数据库快照。

数据库快照变成 Oracle 副本。Amazon RDS 为源数据库和副本设置必要的参数和权限，而不会中断服务。同样，如果删除副本，也不会发生中断。

RDS for Oracle 副本的要求和注意事项

在创建 Oracle 副本之前，请先熟悉以下要求和注意事项。

主题

RDS for Oracle 副本的版本和许可要求

在创建 RDS for Oracle 副本之前，请考虑以下事项：

- 如果副本处于只读模式，请确保您拥有 Active Data Guard 许可证。如果将副本置于装载模式，则不需要 Active Data Guard 许可证。只有 Oracle 数据库引擎支持装载副本。
- Oracle 副本仅受 Oracle Enterprise Edition（EE）支持。
- 仅使用运行 Oracle Database 19c 的非 CDB 实例创建的数据库实例支持非 CDB 的 Oracle 副本。
- Oracle 副本仅适用于在具有两个或更多 vCPU 的数据库实例类上运行的数据库实例。源数据库实例不能使用 db.t3.small 实例类。
- 源数据库实例及其所有副本的 Oracle 数据库引擎版本必须相同。无论副本的维护时段为何时，Amazon RDS 都会在升级源数据库实例后立即升级副本。对于跨区域副本的主要版本升级，Amazon RDS 会自动执行以下操作：
  - 为目标版本生成选项组。
  - 将所有选项和选项设置从原始选项组复制到新选项组。
  - 将升级后的跨区域副本与新选项组关联。

  有关升级数据库引擎版本的更多信息，请参阅升级 RDS for Oracle 数据库引擎。

RDS for Oracle 副本的选项组限制

使用 RDS for Oracle 副本的选项组时，请考虑以下事项：

- 当源和副本位于同一 Amazon 区域时，您不能使用与源数据库实例选项组不同的副本选项组。

  对源选项组或源选项组成员资格做出的修改会传播到 Oracle 副本。不管副本的维护时段是什么，这些更改在应用到源数据库实例后，都会立即应用到副本。有关选项组的更多信息，请参阅 使用选项组。
- 您无法从自动为 RDS for Oracle 跨区域副本创建的专用选项组中删除该副本。
- 您无法将 RDS for Oracle 跨区域副本的专用选项组添加到其他数据库实例。
- 您无法从 RDS for Oracle 跨区域副本的专用选项组中添加或移除非复制选项，但以下选项除外：
  - `NATIVE_NETWORK_ENCRYPTION`
  - `OEM`
  - `OEM_AGENT`
  - `SSL`

  要将其他选项添加到 RDS for Oracle 跨区域副本，请将它们添加到源数据库实例的选项组中。选项还会安装在源数据库实例的所有副本上。对于许可选项，请确保副本具有足够的许可证。

提升 RDS for Oracle 跨区域副本时，提升后的副本的行为将与其他 Oracle 数据库实例的行为相同，包括其选项的管理。您可以通过删除副本的源数据库实例来显式或隐式提升副本。

有关选项组的更多信息，请参阅 使用选项组。

- 您无法向 RDS for Oracle 跨区域副本添加 EFS_INTEGRATION 选项。

RDS for Oracle 的备份和还原注意事项

在创建 RDS for Oracle 副本之前，请考虑以下事项：

- 要创建 RDS for Oracle 副本的快照或开启自动备份，请确保手动设置备份保留期。原定设置情况下，自动备份处于开启状态。
- 还原副本备份时，将还原到数据库时间，而不是进行备份的时间。数据库时间指备份中数据的最新应用的事务时间。差异很大，因为副本可能会落后于主数据库数分钟或数小时。

  要查找差异，请使用 describe-db-snapshots 命令。比较 snapshotDatabaseTime（这是副本备份的数据库时间）与 OriginalSnapshotCreateTime 字段（这是主数据库上最新应用的事务）。

RDS for Oracle 副本的 Oracle Data Guard 要求和限制

在创建 RDS for Oracle 副本之前，请注意以下要求和限制：

- 如果您的主数据库实例使用多租户架构的单租户配置，请考虑以下几点：
  - 您必须将 Oracle Database 19c 或更高版本与企业版一起使用。
  - 您的主 CDB 实例必须位于 ACTIVE 生命周期。
  - 您不能将非 CDB 主实例转换为 CDB 实例，也不能在同一操作中转换其副本。而是应删除非 CDB 副本，将主数据库实例转换为 CDB，然后创建新的副本。
- 确保主数据库实例上的登录触发器允许 RDS_DATAGUARD 用户以及 AUTHENTICATED_IDENTITY 值为 RDS_DATAGUARD 或 rdsdb 的任何用户进行访问。此外，触发器不得为 RDS_DATAGUARD 用户设置当前架构。
- 为了避免阻止来自 Data Guard 代理进程的连接，请不要启用受限会话。有关受限会话的更多信息，请参阅 启用和禁用受限制的会话。

RDS for Oracle 副本的其他注意事项

在创建 RDS for Oracle 副本之前，请考虑以下事项：

- 如果您的数据库实例是一个或多个跨区域副本的源，该源数据库将会保留其归档重做日志文件，直到在所有跨区域副本上应用这些日志文件为止。存档重做日志可能会导致存储消耗增加。
- 为避免 RDS 自动化发生中断，系统触发器必须允许特定用户登录主数据库和副本数据库。系统触发器包括 DDL、登录和数据库角色触发器。我们建议您向触发器添加代码，来排除以下示例代码中列出的用户：

```
-- Determine who the user is
SELECT SYS_CONTEXT('USERENV','AUTHENTICATED_IDENTITY') INTO CURRENT_USER FROM DUAL;
-- The following users should always be able to login to either the Primary or
 Replica
IF CURRENT_USER IN ('master_user', 'SYS', 'SYSTEM', 'RDS_DATAGUARD', 'rdsdb') THEN
RETURN;
END IF;
```

- 只读副本支持块更改跟踪，但装载副本不支持。您可以将装载副本更改为只读副本，然后启用块更改跟踪。有关更多信息，请参阅 启用和禁用数据块更改跟踪。
- 当源数据库使用 Secrets Manager 管理主用户凭证时，您无法创建 Oracle 只读副本。

为创建 Oracle 副本做准备

在开始使用副本之前，请执行以下任务。

主题

启用自动备份

您必须在数据库实例上启用自动备份，然后其才能充当源数据库实例。要了解如何执行此过程，请参阅启用自动备份。

启用强制日志记录模式

建议您启用强制日志记录模式。在强制日志记录模式下，即使将 NOLOGGING 与数据定义语言 (DDL) 语句一起使用，Oracle 数据库也会写入重做记录。

启用强制日志记录模式

1. 使用 SQL Developer 等客户端工具登录 Oracle 数据库。
2. 通过运行以下过程启用强制日志记录模式。

```
exec rdsadmin.rdsadmin_util.force_logging(p_enable => true);
```

有关此过程的更多信息，请参阅 设置强制日志记录。

更改日志记录配置

对于 n 个大小为 m 的联机重做日志，RDS 会在主数据库实例和所有副本上自动创建 n +1 个大小为 m 的备用日志。每当您在主服务器上更改日志记录配置时，更改都会自动传播到副本。

如果您更改日志记录配置，请考虑以下指南：

- 我们建议您在将数据库实例设为副本源之前完成更改。RDS for Oracle 还支持在实例成为源后对其进行更新。
- 在更改主数据库实例上的日志记录配置之前，请检查每个副本是否有足够的存储来容纳新的配置。

您可以使用 Amazon RDS 过程 `rdsadmin.rdsadmin_util.add_logfile` 和 `rdsadmin.rdsadmin_util.drop_logfile` 修改数据库实例的日志记录配置。有关更多信息，请参阅添加联机重做日志 和删除联机重做日志。

设置 MAX_STRING_SIZE 参数

创建 Oracle 副本之前，请确保源数据库实例和副本上的 MAX_STRING_SIZE 参数的设置相同。您可以通过将它们与相同的参数组关联来实现。如果您为源和副本使用不同的参数组，则可以将 MAX_STRING_SIZE 设置为相同值。有关设置该参数的更多信息，请参阅为新数据库实例开启扩展数据类型。

规划计算和存储资源

确保源数据库实例及其副本在计算和存储容量方面具有适当的大小，以适合其运行负载。如果副本达到计算、网络或存储资源容量，则副本会停止接收或应用来自其源实例的更改。Amazon RDS for Oracle 不会通过干预来缓解源数据库实例及其副本之间的高副本滞后。您可以单独修改某个副本的存储和 CPU 资源，而不影响其源及其他副本。

在装载模式下创建 RDS for Oracle 副本

默认情况下，Oracle 副本是只读的。要在装载模式下创建副本，请使用控制台、Amazon CLI、或 RDS API。

控制台

从源 Oracle 数据库实例创建装载副本

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。
3. 选择要作为装载副本源的 Oracle 数据库实例。
4. 对于操作，选择创建副本。
5. 对于副本模式，选择装载。
6. 选择要使用的设置。对于数据库实例标识符，输入该只读副本的名称。根据需要，调整其他设置。
7. 对于区域，选择将启动装载副本的区域。
8. 选择您的实例大小和存储类型。我们建议您为只读副本使用与源数据库实例相同的数据库实例类和存储类型。
9. 对于多可用区部署，选择创建备用实例以便在另一个可用区中创建副本的备用来支持装载副本的故障转移。创建您的装载副本作为多可用区数据库实例与源数据库是否为多可用区数据库实例无关。
10. 选择要使用的其他设置。
11. 选择创建副本。

在数据库页面中，装载副本具有角色“副本”。

Amazon CLI

要在装载模式下创建 Oracle 副本，请在 `--replica-mode` 命令 create-db-instance-read-replica 中将 `mounted` 设为 Amazon CLI。

Example

对于 Linux、macOS 或 Unix：

```
aws rds create-db-instance-read-replica \
    --db-instance-identifier myreadreplica \
    --source-db-instance-identifier mydbinstance \
    --replica-mode mounted
```

对于 Windows：

```
aws rds create-db-instance-read-replica ^
    --db-instance-identifier myreadreplica ^
    --source-db-instance-identifier mydbinstance ^
    --replica-mode mounted
```

要将只读副本更改为装载状态，请在 `--replica-mode` 命令 modify-db-instance 中将 `mounted` 设为 Amazon CLI。要将挂载副本置于只读模式，请将 `--replica-mode` 设为 `open-read-only`。

RDS API

要在装载模式下创建 Oracle 副本，请在 RDS API 操作 CreateDBInstanceReadReplica 中指定 `ReplicaMode=mounted`。

修改 RDS for Oracle 副本模式

要更改现有副本的副本模式，请使用 控制台、Amazon CLI 或 RDS API。当您更改为装载模式时，副本将断开所有活动连接。当您更改为只读模式时，Amazon RDS 会初始化 Active Data Guard。

更改操作可能需要几分钟才能完成。在操作过程中，数据库实例状态会更改为正在修改。有关状态更改的更多信息，请参阅查看Amazon RDS 数据库实例状态。

控制台

将 Oracle 副本的副本模式从装载更改为只读

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。

3. 选择装载副本数据库。
4. 选择修改。
5. 对于副本模式，选择 只读。
6. 选择要使更改的其他设置：
7. 选择 Continue (继续)。
8. 对于修改计划，选择立即应用。
9. 选择修改数据库实例。

Amazon CLI

要将只读副本更改为装载模式，请在 `--replica-mode` 命令 modify-db-instance中将 `mounted` 设为 Amazon CLI。要将装载副本更改为只读模式，请将 `--replica-mode` 设为 `open-read-only`。

Example

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier myreadreplica \
    --replica-mode mode
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier myreadreplica ^
    --replica-mode mode
```

RDS API

要将只读副本更改为装载模式，请在 ModifyDBInstance 中设置 `ReplicaMode=mounted`。要将装载副本更改为只读模式，请设置 `ReplicaMode=read-only`。

使用 RDS for Oracle 副本备份

您可以创建和还原 RDS for Oracle 副本的备份。同时支持自动备份和手动快照。有关更多信息，请参阅备份、还原和导出数据。以下各节描述了管理主实例的备份与管理 RDS for Oracle 副本的备份之间的主要区别。

开启 RDS for Oracle 副本备份

Oracle 副本在原定设置情况下开启自动备份。您可以将备份保留期设置为非零正值来开启自动备份。

控制台

立即启用自动备份

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择数据库，然后选择要修改的数据库实例或多可用区数据库集群。
3. 选择 Modify(修改)。
4. 对于备份保留期，请选择非零正值，例如三天。
5. 选择 Continue (继续)。
6. 选择立即应用。
7. 选择修改数据库实例或修改集群以保存更改并启用自动备份。

Amazon CLI

要启用自动备份，请使用 Amazon CLI modify-db-instance 或 modify-db-cluster 命令。

包括以下参数：

- `--db-instance-identifier`（或 `--db-cluster-identifier` 用于可用区数据库集群）
- `--backup-retention-period`
- `--apply-immediately` 或者 `--no-apply-immediately`

在以下示例中，启用了自动备份并将备份保留期设置为三天。更改将立即应用。

Example

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier my_db_instance  \
    --backup-retention-period 3 \
    --apply-immediately
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier my_db_instance  ^
    --backup-retention-period 3 ^
    --apply-immediately
```

RDS API

要启用自动备份，请使用 RDS API ModifyDBInstance 或 ModifyDBCluster 操作以及以下必需参数：

- `DBInstanceIdentifier` 或 `DBClusterIdentifier`
- `BackupRetentionPeriod`

还原 RDS for Oracle 副本备份

您可以还原 Oracle 副本备份，就像还原主实例的备份一样。有关更多信息，请参阅下列内容：

- 还原到数据库实例
- 将 Amazon RDS 的数据库实例还原到指定时间

还原副本备份时的主要注意事项是确定要还原到的时间点。数据库时间指备份中数据的最新应用的事务时间。还原副本备份时，将还原到数据库时间，而不是备份完成的时间。差异很大，因为 RDS for Oracle 副本可能会落后于主数据库几分钟或几小时。因此，副本备份的数据库时间以及还原到的时间点可能比备份创建时间早得多。

要查找数据库时间和创建时间之间的差异，请使用 `describe-db-snapshots` 命令。比较 `SnapshotDatabaseTime`（这是副本备份的数据库时间）与 `OriginalSnapshotCreateTime` 字段（这是主数据库上最新应用的事务）。以下示例显示了两个时间之间的差异：

```
aws rds describe-db-snapshots \
    --db-instance-identifier my-oracle-replica
    --db-snapshot-identifier my-replica-snapshot

{
    "DBSnapshots": [
        {
            "DBSnapshotIdentifier": "my-replica-snapshot",
            "DBInstanceIdentifier": "my-oracle-replica",
```

```
            "SnapshotDatabaseTime": "2022-07-26T17:49:44Z",
            ...
            "OriginalSnapshotCreateTime": "2021-07-26T19:49:44Z"
        }
    ]
}
```

执行 Oracle Data Guard 切换

切换是主数据库和备用数据库之间的角色反转。在切换期间，原始主数据库转换为备用角色，而原始备用数据库转换为主角色。

在 Oracle Data Guard 环境中，主数据库支持一个或多个备用数据库。您可以执行从主数据库到备用数据库的基于切换的托管式角色转换。切换是主数据库和备用数据库之间的角色反转。在切换期间，原始主数据库转换为备用角色，而原始备用数据库转换为主角色。

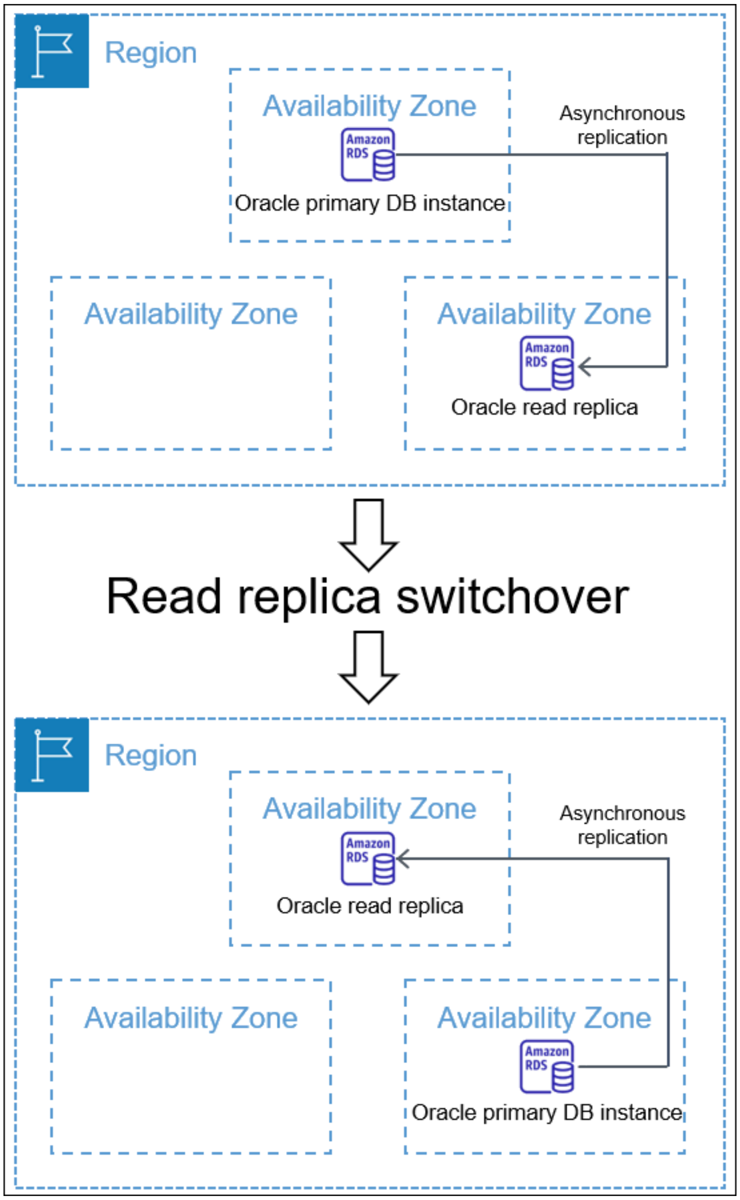
主题

- Oracle Data Guard 切换概述
- Oracle Data Guard 切换的要求
- 启动 Oracle Data Guard 切换
- 监控 Oracle Data Guard 切换

Oracle Data Guard 切换概述

Amazon RDS 支持 Oracle Database 副本的完全托管式、基于切换的角色转换。您只能启动到已挂载或以只读方式打开的备用数据库的切换。

副本可以位于单独的 Amazon Web Services 区域中，也可以位于单个区域的不同可用区（AZ）中。支持所有 Amazon Web Services 区域。



切换与只读副本提升不同。在切换过程中，源和副本数据库实例会更改角色。在提升过程中，只读副本会成为源数据库实例，但源数据库实例不会成为副本。有关更多信息，请参阅 将只读副本提升为独立的数据库实例。

主题

- Oracle Data Guard 切换的优势
- 支持的 Oracle 数据库版本
- Oracle Data Guard 切换的成本
- Oracle Data Guard 切换的工作原理

Oracle Data Guard 切换的优势

就像 RDS for Oracle 只读副本一样，托管式切换依赖于 Oracle Data Guard。该操作旨在实现零数据丢失。Amazon RDS 可自动执行切换的以下各个方面：

- 反转主数据库和指定备用数据库的角色，将新的备用数据库置于与原始备用数据库相同的状态（已挂载或只读）
- 确保数据一致性
- 在转换后维护您的复制配置
- 支持重复性反转，允许您的新备用数据库恢复其原来的主角色

支持的 Oracle 数据库版本

Oracle Database 19c 及更高版本支持 Oracle Data Guard 切换。

Oracle Data Guard 切换的成本

Oracle Data Guard 切换功能不会产生额外费用。Oracle Database 企业版支持挂载模式下的备用数据库。要以只读模式打开备用数据库，您需要 Oracle Active Data Guard 选项。

Oracle Data Guard 切换的工作原理

Oracle Data Guard 切换是一项完全托管式操作。您可以通过发出 CLI 命令 `switchover-read-replica` 启动备用数据库的切换。然后，Amazon RDS 修改您的复制配置中的主角色和备用角色。

原始备用角色和原始主角色是切换前存在的角色。新备用角色和新主角色是切换后存在的角色。旁观者副本是一个副本数据库，它在 Oracle Data Guard 环境中用作备用数据库，但不切换角色。

主题

- [Oracle Data Guard 切换的各个阶段]
- [Oracle Data Guard 切换之后]

Oracle Data Guard 切换的各个阶段

要执行切换，Amazon RDS 必须执行以下步骤：

1. 阻止原始主数据库上的新事务。在切换期间，Amazon RDS 会中断您的 Oracle Data Guard 配置中所有数据库的复制。在切换期间，原始主数据库无法处理写入请求。

2. 将未应用的事务发送到原始备用数据库，并应用它们。
3. 以只读或挂载模式重新启动新的备用数据库。该模式取决于切换前原始备用数据库的打开状态。
4. 以读/写模式打开新的主数据库。

Oracle Data Guard 切换之后

Amazon RDS 切换主数据库和备用数据库的角色。您负责重新连接应用程序并执行任何其他所需的配置。

主题

- [成功标准]
- [与新的主数据库的连接]
- [新的主数据库的配置]

成功标准

当原始备用数据库执行以下操作时，Oracle Data Guard 切换获得成功：

- 转换到其作为新的主数据库的角色
- 完成其重新配置

为了限制停机时间，新的主数据库会尽快变为活动状态。由于 Amazon RDS 异步配置旁观者副本，因此这些副本可能会在原始主数据库之后变为活动状态。

与新的主数据库的连接

切换后，Amazon RDS 不会将您当前的数据库连接传播到新的主数据库。在 Oracle Data Guard 切换完成后，将您的应用程序重新连接到新的主数据库。

新的主数据库的配置

为了切换到新的主数据库，Amazon RDS 将原始备用数据库的模式更改为打开。角色的更改是对数据库的唯一更改。Amazon RDS 未设置多可用区复制等功能。

如果您使用不同的选项切换到跨区域副本，则新的主数据库会保留自己的选项。Amazon RDS 不会迁移原始主数据库上的选项。如果原始主数据库具有 SSL、NNE、OEM 和 OEM_AGENT 等选项，则 Amazon RDS 不会将它们传播到新的主数据库。

Oracle Data Guard 切换的要求

在启动 Oracle Data Guard 切换之前，请确保您的复制环境满足以下要求：

- 原始备用数据库已挂载或以只读方式打开。
- 在原始备用数据库上启用自动备份。
- 原始主数据库和原始备用数据库处于 `available` 状态。
- 原始主数据库和原始备用数据库没有任何待处理的维护操作处于以下状态：`required`、`next window` 或 `in progress`。处于这些状态的操作会阻止切换。要了解如何检查待处理的维护更新的状态，请参阅查看待处理维护更新。

  处于 `available` 状态的待处理维护操作不会阻止切换。RDS for Oracle 经常发布处于状态 `available` 的操作系统（OS）更新。这些待处理的操作系统更新不会阻止切换，除非您将它们安排在下一个维护时段，这会使它们处于 `next window` 状态。

  > **Note**
  >
  > 如果您希望推迟计划的维护操作以便执行切换，请在 RDS 控制台中选择操作，然后选择推迟升级。您还可以通过应用待处理的维护操作或将维护时段移至切换之前的间隔，来防止系统阻止切换。有关更多信息，请参阅 re:Post 文章 How to remove RDS pending maintenance items。

- 原始备用数据库处于复制状态。
- 当主数据库或备用数据库当前处于切换生命周期时，您不要尝试启动切换。如果在切换后正在重新配置副本数据库，Amazon RDS 会阻止您启动另一次切换。

  > **Note**
  >
  > 旁观者副本是 Oracle Data Guard 配置中的副本，它不是切换的目标。在切换期间，旁观者副本可以处于任何状态。

- 原始备用数据库的配置与原始主数据库所需的配置尽可能接近。假设原始主数据库和原始备用数据库具有不同选项的场景。切换完成后，Amazon RDS 不会自动将新的主数据库重新配置为具有与原始主数据库相同的选项。
- 在启动切换之前，您需要配置所需的多可用区部署。Amazon RDS 不将多可用区作为切换的一部分进行管理。多可用区部署保持原样。

假设 db_maz 是多可用区部署中的主数据库，而 db_saz 是单可用区副本。您启动从 db_maz 到 db_saz 的切换。之后，db_maz 是多可用区副本数据库，而 db_saz 是单可用区主数据库。新的主数据库现在不受多可用区部署的保护。

- 在为跨区域切换做准备时，主数据库不使用与复制配置之外的数据库实例相同的选项组。要成功进行跨区域切换，当前主数据库及其只读副本必须是仅有的使用当前主数据库的选项组的数据库实例。否则，Amazon RDS 会阻止切换。

启动 Oracle Data Guard 切换

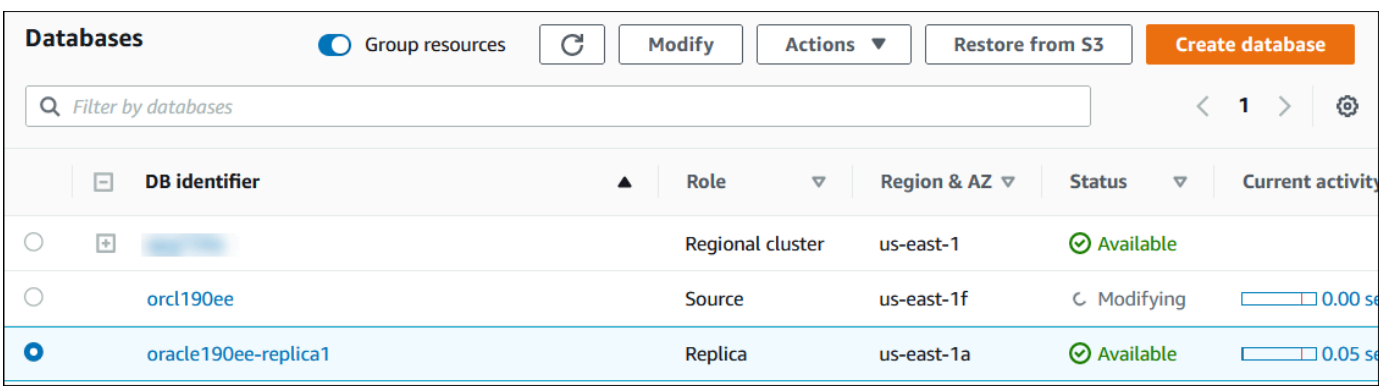
您可以将 RDS for Oracle 只读副本切换为主角色，并将以前的主数据库实例切换为副本角色。

控制台

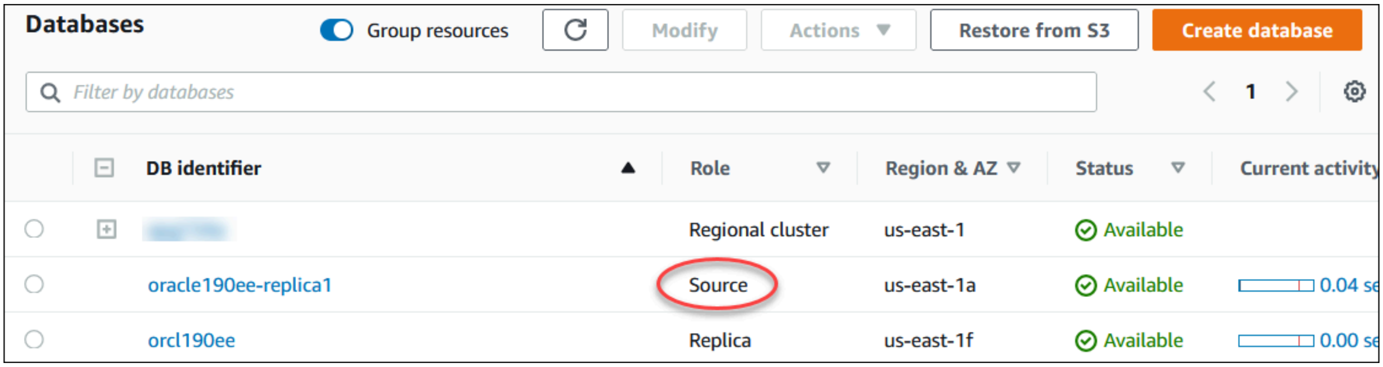
将 Oracle 只读副本切换为主数据库角色

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在 Amazon RDS 控制台中，选择数据库。

   随后会显示数据库窗格。每个只读副本在角色列中显示副本。
3. 选择要切换为主角色的只读副本。
4. 对于 Actions（操作），选择 Switch over replica（切换副本）。
5. 选择 I acknowledge（我确认）。然后选择 Switch over replica（切换副本）。
6. 在 Databases（数据库）页面，监控切换的进度。



   切换完成后，切换目标的角色将从 Replica（副本）更改为 Source（源）。



Amazon CLI

要将 Oracle 副本切换为主数据库角色，请使用 Amazon CLI [switchover-read-replica](https://docs.aws.amazon.com/cli/latest/reference/rds/switchover-read-replica.html) 命令。以下示例使名为 *replica-to-be-made-primary* 的 Oracle 副本成为新的主数据库。

Example

对于 Linux、macOS 或 Unix：

```
aws rds switchover-read-replica \
    --db-instance-identifier replica-to-be-made-primary
```

对于 Windows：

```
aws rds switchover-read-replica ^
    --db-instance-identifier replica-to-be-made-primary
```

RDS API

要将 Oracle 副本切换为主数据库角色，请使用所需参数 `DBInstanceIdentifier` 调用 Amazon RDS API `SwitchoverReadReplica` 操作。此参数指定要承担主数据库角色的 Oracle 副本的名称。

监控 Oracle Data Guard 切换

要检查实例的运行状况，请使用 Amazon CLI 命令 `describe-db-instances`。以下命令检查数据库实例 *orcl2* 的状态。该数据库在切换之前是备用数据库，但在切换之后是新的主数据库。

```
aws rds describe-db-instances \
    --db-instance-identifier orcl2
```

要确认切换成功完成，请查询 V$DATABASE.OPEN_MODE。检查新的主数据库的值是否为 READ WRITE。

```
SELECT OPEN_MODE FROM V$DATABASE;
```

要查找与切换相关的事件，请使用 Amazon CLI 命令 describe-events。以下示例查找有关 *orcl2* 实例的事件。

```
aws rds describe-events \
    --source-identifier orcl2 \
    --source-type db-instance
```

排查 RDS for Oracle 副本的问题

本节介绍了可能出现的复制问题和解决方案。

主题

- 监控 Oracle 复制滞后
- 对添加或修改触发器后 Oracle 复制失败进行故障排查

监控 Oracle 复制滞后

要监控 Amazon CloudWatch 中的复制滞后，请查看 Amazon RDS ReplicaLag 指标。有关复制滞后时间的信息，请参阅监控只读复制和Amazon RDS 的 Amazon CloudWatch 指标。

对于只读副本，如果滞后时间太长，请查询以下视图：

- V$ARCHIVED_LOG – 显示哪些提交已应用到只读副本。
- V$DATAGUARD_STATS – 显示组成 ReplicaLag 指标的各个组成部分的详细分解。
- V$DATAGUARD_STATUS – 显示 Oracle 内部复制过程的日志输出。

对于挂载的副本，如果滞后时间太长，您将无法查询 V$ 视图。而是执行以下操作：

- 检查 CloudWatch 中的 ReplicaLag 指标。
- 在控制台中检查副本的提示日志文件。查找恢复消息中的错误。这些消息包括日志序列号，您可以将其与主序列号进行比较。有关更多信息，请参阅 Amazon RDS for Oracle 数据库日志文件。

对添加或修改触发器后 Oracle 复制失败进行故障排查

如果添加或修改任何触发器，并且之后复制失败，可能是触发器的问题。确保触发器将 RDS 进行复制所需的以下用户账户排除在外：

- 具有管理员权限的用户账户
- `SYS`
- `SYSTEM`
- `RDS_DATAGUARD`
- `rdsdb`

有关更多信息，请参阅 RDS for Oracle 副本的其他注意事项。

向 Oracle 数据库实例添加选项

在 Amazon RDS 中，选项是额外的功能。您可在下面找到可以添加到运行 Oracle 数据库引擎的 Amazon RDS 实例的选项描述。

主题

Oracle DB 选项概述

要为 Oracle 数据库启用选项，可将它们添加到选项组中，然后将选项组关联到数据库实例。有关更多信息，请参阅 使用选项组。

主题

- “Oracle 数据库”选项摘要
- 不同版本支持的选项
- 特定选项的内存要求

“Oracle 数据库”选项摘要

对于 Oracle 数据库实例，您可以添加以下选项。

选项	选项 ID
Amazon S3 集成	S3_INTEGRATION
Oracle Application Express (APEX)	APEX APEX-DEV
Oracle Enterprise Manager	OEM OEM_AGENT
Oracle Java 虚拟机	JVM
Oracle Label Security	OLS
Oracle Locator	LOCATOR
Oracle 本机网络加密	NATIVE_NETWORK_ENCRYPTION
Oracle OLAP	OLAP
Oracle 安全套接字层	SSL
Oracle Spatial	SPATIAL
Oracle SQLT	SQLT
Oracle Statspack	STATSPACK
Oracle 时区	Timezone

选项	选项 ID
Oracle 时区文件自动升级	TIMEZONE_FILE_AUTO UPGRADE
Oracle 透明数据加密	TDE
Oracle UTL_MAIL	UTL_MAIL
Oracle XML DB	XMLDB

不同版本支持的选项

如果不支持该选项，RDS for Oracle 会阻止您将该选项添加到版本中。要了解不同 Oracle 数据库版本中支持哪些 RDS 选项，请使用命令 `aws rds describe-option-group-options`。以下示例列出 Oracle Database 19c 企业版支持的选项。

```
aws rds describe-option-group-options \
    --engine-name oracle-ee \
    --major-engine-version 19
```

有关更多信息，请参阅 Amazon CLI 命令引用中的 describe-option-group-options。

特定选项的内存要求

某些选项需要额外的内存才能在数据库实例上运行。例如，Oracle Enterprise Manager Database Control 使用约 300 MB 的 RAM。如果为小数据库实例启用该选项，则可能会遇到内存限制导致的性能问题。您可以调整 Oracle 参数，以便数据库需要更少的 RAM。或者，您也可以扩展到更大的数据库实例。

Amazon S3 集成

您可以在 RDS for Oracle 数据库实例和 Amazon S3 存储桶之间传输文件。您可以使用 Amazon S3 与 Oracle Database 集成的功能，例如 Oracle Data Pump。例如，您可以将 Data Pump 文件从 Amazon S3 下载到 RDS for Oracle 数据库实例。有关更多信息，请参阅 将数据导入到 Amazon RDS 上的 Oracle。

> **Note**
>
> 数据库实例和 Amazon S3 存储桶必须位于同一 Amazon Web Services 区域。

主题

为与 Amazon S3 集成的 RDS for Oracle 配置 IAM 权限

若要让 RDS for Oracle 与 Amazon S3 集成，您的数据库实例必须可以访问 Amazon S3 存储桶。数据库实例使用的 Amazon VPC 不需要提供 Amazon S3 终端节点的访问权限。

RDS for Oracle 支持在一个账户中的数据库实例与另一个账户中的 Amazon S3 存储桶之间传输文件。如果需要其他步骤，将在以下各节中说明这些步骤。

主题

步骤 1：为 Amazon RDS 角色创建 IAM 策略

在此步骤中，您将创建一个 Amazon Identity and Access Management（IAM）策略，该策略具有在 Amazon S3 存储桶和 RDS 数据库实例之间传输文件所需的权限。此步骤假定您已创建了 S3 桶。

创建策略前，请记下以下信息：

- 存储桶的 Amazon 资源名称（ARN）。
- 您的 Amazon KMS 密钥的 ARN，如果您的存储桶使用 SSE-KMS 或 SSE-S3 加密

> **Note**
>
> RDS for Oracle 数据库实例无法访问使用 SSE-C 加密的 Amazon S3 桶。

有关更多信息，请参阅《Amazon Simple Storage Service 用户指南》中的使用服务器端加密保护数据。

控制台

创建 IAM policy 以允许 Amazon RDS 访问 Amazon S3 桶

1. 打开 IAM 管理控制台。
2. 在 Access management (访问管理) 下，选择 Policies (策略)。
3. 选择创建策略。
4. 在 Visual editor (可视化编辑器) 选项卡上，选择 Choose a service (选择服务)，然后选择 S3。
5. 对于 Actions (操作)，选择 Expand all (全部展开)，然后选择将文件从 Amazon S3 存储桶传输到 Amazon RDS 所需的存储桶权限和对象权限。例如，执行以下操作：

   - 展开 List (列表)，然后选择 ListBucket。
   - 展开 Read (读取)，然后选择 GetObject。
   - 展开 Write（写入），然后选择PutObject 和 DeleteObject。
   - 展开 Permissions management（权限管理），然后选择PutObjectAcl。如果您计划将文件上传到其他账户拥有的存储桶，并且此账户需要完全控制存储桶内容，则需要此权限。

   对象权限是 Amazon S3 中的对象操作的权限。您必须为存储桶中的对象而不是存储桶本身授予这些权限。有关更多信息，请参阅对象操作权限。

6. 选择资源，然后执行以下操作：

   a. 选择特定。

   b. 对于桶，选择添加 ARN。输入桶 ARN。桶名称会自动填入。然后，选择 Add (添加)。

   c. 如果显示了对象资源，请选择添加 ARN 以手动添加资源，或者选择任意。

   > **Note**
   >
   > 您可以将 Amazon 资源名称 (ARN) 设置为更具体的 ARN 值，以允许 Amazon RDS 仅访问 Amazon S3 存储桶中的特定文件或文件夹。有关如何为 Amazon S3 定义访问策略的更多信息，请参阅管理您的 Amazon S3 资源的访问权限。

7. （可选）选择 Add additional permissions (添加其他权限)，向策略中添加资源。例如，执行以下操作：

   a. 如果您的存储桶使用自定义 KMS 密钥加密，请为该服务选择 KMS。

   b. 对于手动操作，请选择以下选项：

      - Encrypt
      - 重新加密起点和重新加密终点
      - Decrypt
      - DescribeKey
      - GenerateDataKey

   c. 对于资源，选择特定。

   d. 对于密钥，选择添加 ARN。输入自定义密钥的 ARN 作为资源，然后选择添加。

      有关更多信息，请参阅 Amazon Simple Storage Service 用户指南中的使用在 Amazon Key Management Service (SSE-KMS) 中存储 KMS 密钥的服务器端加密保护数据。

   e. 如果想要 Amazon RDS 访问其他存储桶，请为这些存储桶添加 ARN。您也可以根据需要授予访问 Amazon S3 中所有存储桶和对象的权限。

8. 选择 Next: Tags (下一步: 标签)，然后选择 Next: Review (下一步: 审核)。

9. 对于 Name (名称)，请为您的 IAM 策略输入名称，例如 `rds-s3-integration-policy`。在创建 IAM 角色与您的数据库实例关联时，需要使用此名称。您也可以添加可选的 Description (描述) 值。

10. 选择创建策略。

Amazon CLI

创建向 Amazon RDS 授予对 Amazon S3 存储桶的访问权限的 Amazon Identity and Access Management（IAM）策略。创建策略后，请记下策略的 ARN。后续步骤需要该 ARN。

根据所需的访问类型，在策略中包括适合的操作：

- `GetObject` – 从 Amazon S3 存储桶向 Amazon RDS 传输文件所必需的。
- `ListBucket` – 从 Amazon S3 存储桶向 Amazon RDS 传输文件所必需的。
- `PutObject` – 从 Amazon RDS 向 Amazon S3 存储桶传输文件所必需的。

以下 Amazon CLI 命令使用这些选项创建一个名为 *rds-s3-integration-policy* 的 IAM 策略。该策略授予对名为 *amzn-s3-demo-bucket* 的存储桶的访问权限。

Example

对于 Linux、macOS 或 Unix：

```
aws iam create-policy \
   --policy-name rds-s3-integration-policy \
   --policy-document '{
     "Version": "2012-10-17",
     "Statement": [
       {
         "Sid": "s3integration",
         "Action": [
           "s3:GetObject",
           "s3:ListBucket",
           "s3:PutObject"
         ],
         "Effect": "Allow",
         "Resource": [
           "arn:aws:s3:::amzn-s3-demo-bucket",
           "arn:aws:s3:::amzn-s3-demo-bucket/*"
         ]
       }
     ]
   }'
```

以下示例包含自定义 KMS 密钥的权限。

```
aws iam create-policy \
```

```
    --policy-name rds-s3-integration-policy \
    --policy-document '{
      "Version": "2012-10-17",
      "Statement": [
        {
          "Sid": "s3integration",
          "Action": [
            "s3:GetObject",
            "s3:ListBucket",
            "s3:PutObject",
            "kms:Decrypt",
            "kms:Encrypt",
            "kms:ReEncrypt*",
            "kms:GenerateDataKey",
            "kms:DescribeKey",
          ],
          "Effect": "Allow",
          "Resource": [
            "arn:aws:s3:::amzn-s3-demo-bucket",
            "arn:aws:s3:::amzn-s3-demo-bucket/*",
            "arn:aws:kms:::your-kms-arn"
          ]
        }
      ]
    }'
```

对于 Windows：

```
aws iam create-policy ^
    --policy-name rds-s3-integration-policy ^
    --policy-document '{
      "Version": "2012-10-17",
      "Statement": [
        {
          "Sid": "s3integration",
          "Action": [
            "s3:GetObject",
            "s3:ListBucket",
            "s3:PutObject"
          ],
          "Effect": "Allow",
          "Resource": [
            "arn:aws:s3:::amzn-s3-demo-bucket",
```

```
          "arn:aws:s3:::amzn-s3-demo-bucket/*"
        ]
      }
    ]
  }'
```

以下示例包含自定义 KMS 密钥的权限。

```
aws iam create-policy ^
   --policy-name rds-s3-integration-policy ^
   --policy-document '{
     "Version": "2012-10-17",
     "Statement": [
       {
         "Sid": "s3integration",
         "Action": [
           "s3:GetObject",
           "s3:ListBucket",
           "s3:PutObject",
           "kms:Decrypt",
           "kms:Encrypt",
           "kms:ReEncrypt",
           "kms:GenerateDataKey",
           "kms:DescribeKey",
         ],
         "Effect": "Allow",
         "Resource": [
           "arn:aws:s3:::amzn-s3-demo-bucket",
           "arn:aws:s3:::amzn-s3-demo-bucket/*",
           "arn:aws:kms:::your-kms-arn"
         ]
       }
     ]
   }'
```

步骤 2：（可选）为 Amazon S3 存储桶创建 IAM 策略

只有在以下情况下才需要此步骤：

- 您计划从一个账户（账户 A）将文件上传到 Amazon S3 存储桶，然后从另一个账户（账户 B）访问它们。
- 账户 B 拥有该存储桶。

- 账户 B 需要完全控制加载到存储桶中的对象。

如果上述条件不适用于您，请跳至步骤 3：为您的数据库实例创建 IAM 角色并附加策略。

要创建存储桶策略，请确保您具备以下各项：

- 账户 A 的账户 ID
- 账户 A 的用户名
- 账户 B 中 Amazon S3 存储桶的 ARN 值

控制台

创建或编辑存储桶策略

1. 登录到 Amazon Web Services Management Console，然后通过以下网址打开 Amazon S3 控制台：https://console.aws.amazon.com/s3/。
2. 在 Buckets (存储桶) 列表中，请选择要为其创建或编辑存储桶策略的存储桶的名称。
3. 选择权限。
4. 在 Bucket policy (存储桶策略) 下，请选择 Edit (编辑)。这将打开编辑存储桶策略页面。
5. 在 Edit bucket policy（编辑存储桶策略）页面上，探索《Amazon S3 用户指南》中的 Policy examples（策略示例），选择 Policy generator（策略生成器）自动生成策略，或者在 Policy（策略）部分编辑 JSON。

   如果选择 Policy generator（策略生成器）则 Amazon 策略生成器将在新的窗口中打开。

   a. 在 Amazon 策略生成器页面，在 Select Type of Policy（选择策略类型）中，请选择 S3 存储桶Policy（S3 存储桶策略）。
   b. 通过在提供的字段中输入信息来添加语句，然后选择 Add Statement（添加语句）。对所有您想添加的语句重复执行此操作。有关这些字段的更多信息，请参阅 IAM 用户指南中的 IAM JSON 策略元素参考。

      > **Note**
      >
      > 为方便起见，Edit bucket policy（编辑存储桶策略）页面会在 Policy（策略）文本字段上方显示当前存储桶的 Bucket ARN （Amazon 资源名称）。您可以复制此 ARN，以便在 Amazon 策略生成器页面上的语句中使用。

c. 添加完语句后，请选择生成策略。

d. 复制生成的策略文本，请选择 Close（关闭），然后返回到 Amazon S3 控制台中的 Edit bucket policy（编辑存储桶策略）页面。

6. 在 Policy（策略）框中，编辑现有策略或从策略生成器粘贴存储桶策略。确保在保存策略之前解决安全警告、错误、一般警告和建议。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Example permissions",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::account-A-ID:account-A-user"
      },
      "Action": [
        "s3:PutObject",
        "s3:PutObjectAcl"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-destination-bucket",
        "arn:aws:s3:::amzn-s3-demo-destination-bucket/*"
      ]
    }
  ]
}
```

7. 请选择 Save changes（保存更改），此操作将让您返回到存储桶权限页面。

步骤 3：为您的数据库实例创建 IAM 角色并附加策略

此步骤假定您已在步骤 1：为 Amazon RDS 角色创建 IAM 策略中创建了 IAM 策略。在此步骤中，您将为 RDS for Oracle 数据库实例创建一个角色，然后将策略附加到该角色。

控制台

创建 IAM 角色以允许 Amazon RDS 访问 Amazon S3 桶

1. 打开 IAM 管理控制台。
2. 在导航窗格中，选择角色。

3. 选择创建角色。
4. 选择 Amazon 服务。
5. 对于其他 Amazon 服务的使用案例：，选择 RDS，然后选择 RDS – 向数据库添加角色。然后选择下一步。
6. 对于权限策略下面的搜索，请输入您在步骤 1：为 Amazon RDS 角色创建 IAM 策略中创建的 IAM policy 的名称，然后当策略在列表中显示时选择该策略。然后选择下一步。
7. 对于角色名称，输入 IAM 角色的名称，例如 `rds-s3-integration-role`。您也可以添加可选的描述值。
8. 选择创建角色。

Amazon CLI

创建一个角色并向该角色附加策略

1. 创建一个让 Amazon RDS 可代表您访问 Amazon S3 存储桶的 IAM 角色。

   我们建议在基于资源的信任关系中使用 `aws:SourceArn` 和 `aws:SourceAccount` 全局条件上下文键，以此限制服务对特定资源的权限。这是防范混淆代理问题最有效的方法。

   您可以使用这两个全局条件上下文键并让 `aws:SourceArn` 值包含账户 ID。在这种情况下，当 `aws:SourceAccount` 值和 `aws:SourceArn` 值中的账户使用相同策略语句时，确保二者使用相同的账户 ID。

   - 如果您想对单个资源进行跨服务访问，请使用 `aws:SourceArn`。
   - 如果您想允许该账户中的任何资源与跨服务使用操作相关联，请使用 `aws:SourceAccount`。

   在信任关系中，请务必使用 `aws:SourceArn` 全局条件上下文键和访问角色资源的完整 Amazon Resource Name (ARN)。

   以下 Amazon CLI 命令会创建名为 *`rds-s3-integration-role`* 的角色来实现此目的。

   Example

   对于 Linux、macOS 或 Unix：

```
aws iam create-role \
   --role-name rds-s3-integration-role \
   --assume-role-policy-document '{
```

```
    "Version": "2012-10-17",
    "Statement": [
      {
        "Effect": "Allow",
        "Principal": {
           "Service": "rds.amazonaws.com"
         },
        "Action": "sts:AssumeRole",
        "Condition": {
            "StringEquals": {
                "aws:SourceAccount": "my_account_ID",
                "aws:SourceArn": "arn:aws:rds:Region:my_account_ID:db:dbname"
            }
        }
      }
    ]
  }'
```

对于 Windows：

```
aws iam create-role ^
   --role-name rds-s3-integration-role ^
   --assume-role-policy-document '{
     "Version": "2012-10-17",
     "Statement": [
       {
         "Effect": "Allow",
         "Principal": {
            "Service": "rds.amazonaws.com"
          },
         "Action": "sts:AssumeRole",
         "Condition": {
             "StringEquals": {
                 "aws:SourceAccount": "my_account_ID",
                 "aws:SourceArn": "arn:aws:rds:Region:my_account_ID:db:dbname"
             }
         }
       }
     ]
   }'
```

有关更多信息，请参阅 IAM 用户指南 中的[创建向 IAM 用户委派权限的角色]。

2. 创建角色之后，记下该角色的 ARN。后续步骤需要该 ARN。
3. 将您创建的策略附在您创建的角色上。

   以下 Amazon CLI 命令将策略附加到名为 *rds-s3-integration-role* 的角色。

   Example

   对于 Linux、macOS 或 Unix：

```
aws iam attach-role-policy \
   --policy-arn your-policy-arn \
   --role-name rds-s3-integration-role
```

   对于 Windows：

```
aws iam attach-role-policy ^
   --policy-arn your-policy-arn ^
   --role-name rds-s3-integration-role
```

   将 *your-policy-arn* 替换为您在上一步中记下的策略 ARN。

步骤 4：将您的 IAM 角色与 RDS for Oracle 数据库实例关联

配置进行 Amazon S3 集成的权限的最后一步是将您的 IAM 角色与数据库实例相关联。请注意以下要求：

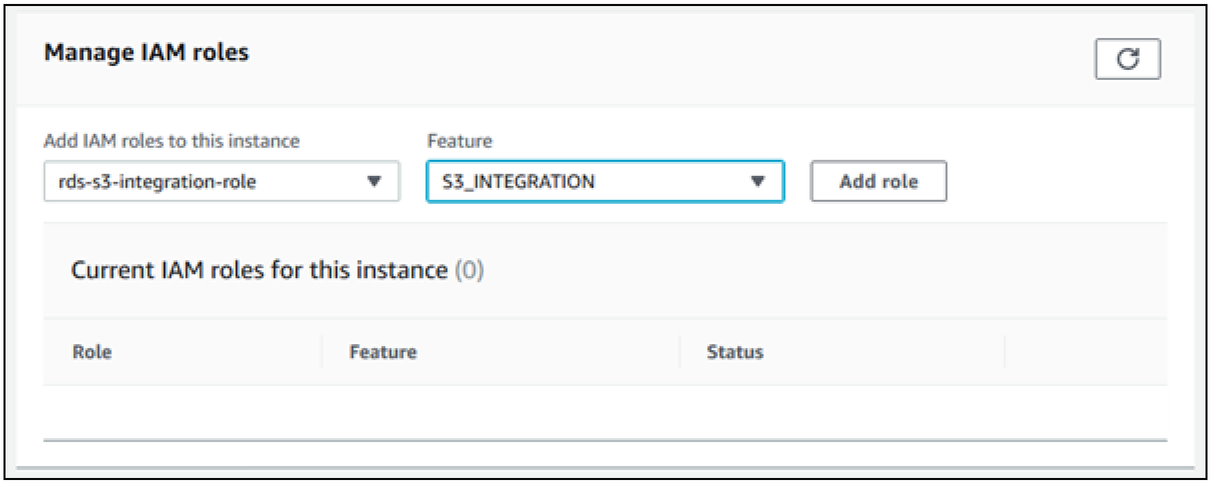
- 您必须具有访问 IAM 角色的权限，并向此角色附加所需的 Amazon S3 权限策略。
- 您一次只能将一个 IAM 角色与 RDS for Oracle 数据库实例关联。
- 数据库实例必须处于可用状态。

控制台

将您的 IAM 角色与 RDS for Oracle 数据库实例关联

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 从导航窗格中选择 Databases (数据库)。
3. 选择 RDS for Oracle 数据库实例名称以显示其详细信息。

4. 在 Connectivity & security（连接性和安全性）选项卡上，向下滚动到页面底部的 Manage IAM roles（管理 IAM 角色）部分。
5. 对于向此实例添加 IAM 角色，选择您在[步骤 3：为您的数据库实例创建 IAM 角色并附加策略]()中创建的角色。
6. 对于 Feature (功能)，选择 S3_INTEGRATION。



7. 选择 Add role (添加角色)。

Amazon CLI

以下 Amazon CLI 命令将角色添加到名为 *mydbinstance* 的 Oracle 数据库实例。

Example

对于 Linux、macOS 或 Unix：

```
aws rds add-role-to-db-instance \
   --db-instance-identifier mydbinstance \
   --feature-name S3_INTEGRATION \
   --role-arn your-role-arn
```

对于 Windows：

```
aws rds add-role-to-db-instance ^
   --db-instance-identifier mydbinstance ^
   --feature-name S3_INTEGRATION ^
   --role-arn your-role-arn
```

将 *your-role-arn* 替换为您在上一步中记下的角色 ARN。必须为 `S3_INTEGRATION` 选项指定 `--feature-name`。

添加 Amazon S3 集成选项

要将 Amazon RDS for Oracle 与 Amazon S3 集成，您的数据库实例必须与包括 `S3_INTEGRATION` 选项的选项组关联。

控制台

为 Amazon S3 集成配置选项组

1. 创建新选项组或识别现有选项组，然后您可以将 `S3_INTEGRATION` 选项添加到其中。

   有关创建选项组的信息，请参阅 创建选项组。
2. 将 `S3_INTEGRATION` 选项添加到该选项组。

   有关向选项组添加选项的信息，请参阅 将选项添加到选项组。
3. 创建新的 RDS for Oracle 数据库实例并将选项组与其关联，或者修改要将选项组与其关联的 RDS for Oracle 数据库实例。

   有关创建数据库实例的信息，请参阅创建 Amazon RDS 数据库实例。

   有关修改 数据库实例的信息，请参阅。修改 Amazon RDS 数据库实例

Amazon CLI

为 Amazon S3 集成配置选项组

1. 创建新选项组或识别现有选项组，然后您可以将 `S3_INTEGRATION` 选项添加到其中。

   有关创建选项组的信息，请参阅 创建选项组。
2. 将 `S3_INTEGRATION` 选项添加到该选项组。

   例如，以下 Amazon CLI 命令将 `S3_INTEGRATION` 选项添加到名为 **myoptiongroup** 的选项组。

   Example

   对于 Linux、macOS 或 Unix：

```
aws rds add-option-to-option-group \
    --option-group-name myoptiongroup \
    --options OptionName=S3_INTEGRATION,OptionVersion=1.0
```

对于 Windows：

```
aws rds add-option-to-option-group ^
    --option-group-name myoptiongroup ^
    --options OptionName=S3_INTEGRATION,OptionVersion=1.0
```

3. 创建新的 RDS for Oracle 数据库实例并将选项组与其关联，或者修改要将选项组与其关联的 RDS for Oracle 数据库实例。

   有关创建数据库实例的信息，请参阅创建 Amazon RDS 数据库实例。

   有关修改 RDS for Oracle 数据库实例的信息，请参阅修改 Amazon RDS 数据库实例。

在 Amazon RDS for Oracle 和 Amazon S3 存储桶之间传输文件

要在 RDS for Oracle 数据库实例和 Amazon S3 桶之间传输文件，您可以使用 Amazon RDS 软件包 `rdsadmin_s3_tasks`。您可以在上传文件时使用 GZIP 压缩文件，并在下载时解压缩它们。

主题

- 文件传输功能的要求和限制
- 将文件从 RDS for Oracle 数据库实例上传到 Amazon S3 存储桶
- 将文件从 Amazon S3 存储桶下载到 Oracle 数据库实例
- 监控文件传输的状态

文件传输功能的要求和限制

在您的数据库实例和 Amazon S3 存储桶之间传输文件之前，请注意以下事项：

- `rdsadmin_s3_tasks` 软件包传输位于单个目录中的文件。您不能在传输中包括子目录。
- Amazon S3 存储桶中的最大对象大小为 5 TB。
- `rdsadmin_s3_tasks` 创建的任务以异步方式运行。
- 您可以从 Data Pump 目录（例如 `DATA_PUMP_DIR`）或用户创建的任何目录上传文件。您不能从 Oracle 后台进程使用的目录（例如 `adump`、`bdump` 或 `trace` 目录）上传文件。

- 对于 download_from_s3，每次过程调用的下载限制为 2000 个文件。如果您需要从 Amazon S3 下载超过 2000 个文件，请将您的下载分成多个单独的操作，每个过程调用不超过 2000 个文件。
- 如果您的下载文件夹中存在文件，并且您尝试下载同名的文件，download_from_s3 将跳过下载。要从下载目录中移除文件，请使用 PL/SQL 过程 UTL_FILE.FREMOVE。

将文件从 RDS for Oracle 数据库实例上传到 Amazon S3 存储桶

要将文件从数据库实例上传到 Amazon S3 存储桶，请使用 `rdsadmin.rdsadmin_s3_tasks.upload_to_s3` 过程。例如，您可以上传 Oracle Recovery Manager（RMAN）备份文件或 Oracle Data Pump 文件。有关使用对象的更多信息，请参阅 Amazon Simple Storage Service 用户指南。有关执行 RMAN 备份的更多信息，请参阅 执行 Oracle 数据库实例的常见 RMAN 任务。

`rdsadmin.rdsadmin_s3_tasks.upload_to_s3` 过程具有以下参数。

参数名称	数据类型	默认值	必需	描述
`p_bucket_name`	VARCHAR2	–	必需	要将文件上传到其中的 Amazon S3 存储桶的名称。
`p_directory_name`	VARCHAR2	–	必需	要从其中上传文件的 Oracle 目录对象的名称。该目录可以是用户创建的任何目录对象或 Data Pump 目录，例如 `DATA_PUMP_DIR` 。您不能从后台进程使用的目录（例如 adump、bdump 和 `trace`）上传文件。 **Note** 您只可以从指定目录上传文件。您不能上传指定目录的子目录中的文件。

参数名称	数据类型	默认值	必需	描述
`p_s3_prefix`	VARCHAR2	–	必需	待上传文件的 Amazon S3 文件名前缀。如果前缀为空，则将所有文件上传到指定 Amazon S3 存储桶的最顶层，且不为文件名添加前缀。 例如，如果前缀为 `folder_1/oradb` ，则文件上传到 `folder_1`。在这个例子中，`oradb` 前缀添加到每个文件。
`p_prefix`	VARCHAR2	–	必需	一个文件名前缀，文件名必须与其匹配才会上传。如果前缀为空，则将所有文件上传到指定目录。
`p_compression_level`	NUMBER	`0`	可选	GZIP 压缩级别。有效值范围为 `0` 至 9： • `0` – 不压缩 • 1 – 最快压缩 • 9 – 最高压缩比
`p_bucket_owner_full_control`	VARCHAR2	–	可选	存储桶的访问控制设置。有效值仅为 null 和 `FULL_CONTROL` 。仅当您从一个账户（账户 A）将文件上传到另一账户（账户 B）拥有的存储桶，并且账户 B 需要完全控制这些文件时，才需要此设置。

`rdsadmin.rdsadmin_s3_tasks.upload_to_s3` 过程的返回值是任务 ID。

下面的示例将 *DATA_PUMP_DIR* 目录中的所有文件上传到名为 *amzn-s3-demo-bucket* 的 Amazon S3 存储桶。文件不会被压缩。

```
SELECT rdsadmin.rdsadmin_s3_tasks.upload_to_s3(
      p_bucket_name    =>  'amzn-s3-demo-bucket',
      p_prefix         =>  '',
      p_s3_prefix      =>  '',
      p_directory_name =>  'DATA_PUMP_DIR')
   AS TASK_ID FROM DUAL;
```

下面的示例将 *db* 目录中前缀为 *DATA_PUMP_DIR* 的所有文件上传到名为 *amzn-s3-demo-bucket* 的 Amazon S3 存储桶。Amazon RDS 将最高级别的 GZIP 压缩应用于这些文件。

```
SELECT rdsadmin.rdsadmin_s3_tasks.upload_to_s3(
      p_bucket_name       =>  'amzn-s3-demo-bucket',
      p_prefix            =>  'db',
      p_s3_prefix         =>  '',
      p_directory_name    =>  'DATA_PUMP_DIR',
      p_compression_level =>  9)
   AS TASK_ID FROM DUAL;
```

下面的示例将 *DATA_PUMP_DIR* 目录中的所有文件上传到名为 *amzn-s3-demo-bucket* 的 Amazon S3 存储桶。文件上传到 `dbfiles` 文件夹。在此示例中，GZIP 压缩级别为 *1*，这是速度最快的压缩级别。

```
SELECT rdsadmin.rdsadmin_s3_tasks.upload_to_s3(
      p_bucket_name       =>  'amzn-s3-demo-bucket',
      p_prefix            =>  '',
      p_s3_prefix         =>  'dbfiles/',
      p_directory_name    =>  'DATA_PUMP_DIR',
      p_compression_level =>  1)
   AS TASK_ID FROM DUAL;
```

下面的示例将 *DATA_PUMP_DIR* 目录中的所有文件上传到名为 *amzn-s3-demo-bucket* 的 Amazon S3 存储桶。文件上传到 `dbfiles` 文件夹，且 `ora` 添加到每个文件名的开头。不应用压缩。

```
SELECT rdsadmin.rdsadmin_s3_tasks.upload_to_s3(
      p_bucket_name    =>  'amzn-s3-demo-bucket',
```

```
      p_prefix         =>  '',
      p_s3_prefix      =>  'dbfiles/ora',
      p_directory_name =>  'DATA_PUMP_DIR')
   AS TASK_ID FROM DUAL;
```

以下示例假定该命令在账户 A 中运行，但账户 B 需要完全控制存储桶内容。命令 `rdsadmin_s3_tasks.upload_to_s3` 将 *s3bucketOwnedByAccountB* 目录中的所有文件传输到名为 *DATA_PUMP_DIR* 的存储桶。访问控制设置为 `FULL_CONTROL`，以便账户 B 可以访问存储桶中的文件。GZIP 压缩级别为 *6*，它可平衡速度和文件大小。

```
SELECT rdsadmin.rdsadmin_s3_tasks.upload_to_s3(
      p_bucket_name               =>  's3bucketOwnedByAccountB',
      p_prefix                    =>  '',
      p_s3_prefix                 =>  '',
      p_directory_name            =>  'DATA_PUMP_DIR',
      p_bucket_owner_full_control =>  'FULL_CONTROL',
      p_compression_level         =>  6)
   AS TASK_ID FROM DUAL;
```

在每个示例中，`SELECT` 语句返回 `VARCHAR2` 数据类型的任务 ID。

您可以通过显示任务的输出文件来查看结果。

```
SELECT text FROM table(rdsadmin.rds_file_util.read_text_file('BDUMP','dbtask-task-id.log'));
```

将 *task-id* 替换为由此过程返回的任务 ID。

> **Note**
>
> 任务以异步方式执行。

将文件从 Amazon S3 存储桶下载到 Oracle 数据库实例

要将文件从 Amazon S3 桶下载到 RDS for Oracle 实例，请使用 Amazon RDS 过程 `rdsadmin.rdsadmin_s3_tasks.download_from_s3`。

`download_from_s3` 过程具有以下参数。

参数名称	数据类型	默认值	必需	描述
p_bucket_name	VARCHAR	–	必需	要从中下载文件的 Amazon S3 存储桶的名称。
p_directory_name	VARCHAR	–	必需	要将文件下载到其中的 Oracle 目录对象的名称。该目录可以是用户创建的任何目录对象或 Data Pump 目录，例如 DATA_PUMP_DIR 。
p_error_on_zero_downloads	VARCHAR	FALSE	可选	一个标志，用于确定当 Amazon S3 桶中没有与前缀匹配的对象时，任务是否引发错误。如果此参数未设置或设置为 FALSE（原定设置），则任务会输出一条消息，指出未找到任何对象，但不会引发异常或失败。如果此参数为 TRUE，则任务会引发异常并失败。 可能无法通过匹配测试的前缀规范的示例包括前缀中的空格（如在 ' import/test9.log' 中）和大小写不匹配（如在 test9.log 和 test9.LOG 中）。
p_s3_prefix	VARCHAR	–	必需	一个文件名前缀，文件名必须与其匹配才会下载。使用空前缀将下载指定 Amazon S3 存储桶中的所有顶级文件，但不包括存储桶中文件夹内的文件。 该过程仅从匹配前缀的第一级文件夹下载 Amazon S3 对象。匹配指定前缀的嵌套目录结构不会下载。 例如，假定 Amazon S3 存储桶拥有文件夹结构 folder_1/folder_2/folder_3 。您可以指定 'folder_1/folder_2/' 前缀。在这种情况下

参数名称	数据类型	默认值	必需	描述
				，只会下载 folder_2 中的文件，不会下载 folder_1 或 folder_3 中的文件。 如果您改为指定 'folder_1/folder_2' 前缀，则将下载 folder_1 中与 'folder_2' 前缀匹配的所有文件，不下载 folder_2 中的文件。
p_decompression_format	VARCHAR	–	可选	解压缩格式。不解压缩时有效值为 NONE；解压缩时有效值为 GZIP。

rdsadmin.rdsadmin_s3_tasks.download_from_s3 过程的返回值是任务 ID。

下面的示例将名为 *amzn-s3-demo-bucket* 的 Amazon S3 桶中的所有文件下载到 *DATA_PUMP_DIR* 目录。这些文件没有被压缩，因此不会应用解压缩。

```
SELECT rdsadmin.rdsadmin_s3_tasks.download_from_s3(
      p_bucket_name    =>  'amzn-s3-demo-bucket',
      p_directory_name =>  'DATA_PUMP_DIR')
   AS TASK_ID FROM DUAL;
```

下面的示例将名为 *db* 的 Amazon S3 存储桶中前缀为 *amzn-s3-demo-bucket* 的所有文件下载到 *DATA_PUMP_DIR* 目录。这些文件是用 GZIP 压缩的，因此将应用解压缩。参数 p_error_on_zero_downloads 开启前缀错误检查，因此，如果前缀与桶中的任何文件都不匹配，则任务会引发异常并失败。

```
SELECT rdsadmin.rdsadmin_s3_tasks.download_from_s3(
      p_bucket_name               =>  'amzn-s3-demo-bucket',
      p_s3_prefix                 =>  'db',
      p_directory_name            =>  'DATA_PUMP_DIR',
      p_decompression_format      =>  'GZIP',
      p_error_on_zero_downloads   =>  'TRUE')
   AS TASK_ID FROM DUAL;
```

下面的示例将名为 *myfolder/* 的 Amazon S3 存储桶中的 *amzn-s3-demo-bucket* 文件夹中的所有文件下载到 *DATA_PUMP_DIR* 目录。使用 p_s3_prefix 参数来指定 Amazon S3 文件夹。上传的文件使用 GZIP 压缩，但在下载期间不会解压缩。

```
SELECT rdsadmin.rdsadmin_s3_tasks.download_from_s3(
      p_bucket_name           =>  'amzn-s3-demo-bucket',
      p_s3_prefix             =>  'myfolder/',
      p_directory_name        =>  'DATA_PUMP_DIR',
      p_decompression_format =>  'NONE')
   AS TASK_ID FROM DUAL;
```

以下示例会将名为 *amzn-s3-demo-bucket* 的 Amazon S3 存储桶中的所有文件 *mydumpfile.dmp* 下载到 *DATA_PUMP_DIR* 目录。不应用解压缩。

```
SELECT rdsadmin.rdsadmin_s3_tasks.download_from_s3(
      p_bucket_name    =>  'amzn-s3-demo-bucket',
      p_s3_prefix      =>  'mydumpfile.dmp',
      p_directory_name =>  'DATA_PUMP_DIR')
   AS TASK_ID FROM DUAL;
```

在每个示例中，SELECT 语句返回 VARCHAR2 数据类型的任务 ID。

您可以通过显示任务的输出文件来查看结果。

```
SELECT text FROM table(rdsadmin.rds_file_util.read_text_file('BDUMP','dbtask-task-
id.log'));
```

将 *task-id* 替换为由此过程返回的任务 ID。

> **Note**
>
> 任务以异步方式执行。
>
> 您可以使用 UTL_FILE.FREMOVE Oracle 过程从目录中删除文件。有关更多信息，请参阅 Oracle 文档中的 FREMOVE 过程。

监控文件传输的状态

文件传输任务开始和结束时会发布 Amazon RDS 事件。事件消息包含文件传输的任务 ID。有关查看事件的信息，请参阅 查看 Amazon RDS 事件。

您可以查看 bdump 文件中的持续任务的状态。bdump 文件位于 `/rdsdbdata/log/trace` 目录下。每个 bdump 文件名称格式如下。

```
dbtask-task-id.log
```

将 *`task-id`* 替换为您想要监控的任务的 ID。

> **Note**
> 任务以异步方式执行。

您可以使用 `rdsadmin.rds_file_util.read_text_file` 存储过程查看 bdump 文件的内容。例如，以下查询将返回 *`dbtask-1234567890123-1234.log`* bdump 文件的内容。

```
SELECT text FROM
 table(rdsadmin.rds_file_util.read_text_file('BDUMP','dbtask-1234567890123-1234.log'));
```

以下示例显示传输失败的日志文件。

```
TASK_ID









--------------------------------------------------------------------------------
1234567890123-1234


TEXT
```

```
--------------------------------------------------------------------------------------------
2023-04-17 18:21:33.993 UTC [INFO ] File #1: Uploading the file /rdsdbdata/datapump/
A123B4CDEF567890G1234567890H1234/sample.dmp to Amazon S3 with bucket name amzn-s3-demo-
bucket and key sample.dmp.
2023-04-17 18:21:34.188 UTC [ERROR] RDS doesn't have permission to write to Amazon S3
 bucket name amzn-s3-demo-bucket and key sample.dmp.
2023-04-17 18:21:34.189 UTC [INFO ] The task failed.
```

排查 Amazon S3 集成问题

有关问题排查提示，请参阅 Amazon re:Post 文章如何排查将 Amazon RDS for Oracle 与 Amazon S3 集成时出现的问题？。

删除 Amazon S3 集成选项

您可以从数据库实例中删除 Amazon S3 集成选项。

要从数据库实例中删除 Amazon S3 集成选项，请执行下列操作之一：

- 要从多个数据库实例中删除 Amazon S3 集成选项，请从数据库实例所属的选项组中删除 S3_INTEGRATION 选项。此更改会影响使用该选项组的所有数据库实例。有关更多信息，请参阅 从选项组中删除选项。

- 要从单个数据库实例中删除 Amazon S3 集成选项，请修改该实例，并指定不包含 S3_INTEGRATION 选项的另一个选项组。您可以指定默认（空）选项组，或指定其他自定义选项组。有关更多信息，请参阅 修改 Amazon RDS 数据库实例。

Oracle Application Express (APEX)

Amazon RDS 通过使用 APEX 和 APEX-DEV 选项支持 Oracle Application Express (APEX)。您可以将 Oracle APEX 部署为运行时环境或基于 Web 的应用程序的完整开发环境。使用 Oracle APEX，您可完全在 Web 浏览器内构建应用程序。有关更多信息，请参阅 Oracle 文档中的 Oracle Application Express。

主题

Oracle APEX 组件

Oracle APEX 包含以下主要组件：

- 存储 Oracle APEX 应用程序和组件元数据的存储库。该存储库包含安装在 Amazon RDS 数据库实例内的表、索引和其他对象。
- 管理与 Oracle APEX 客户端的 HTTP 通信的侦听器。侦听器位于单独的主机上，例如 Amazon EC2 实例、公司的本地服务器或您的台式计算机。该侦听器接受来自 Web 浏览器的传入连接并将这些连接转发至 Amazon RDS 数据库实例进行处理，然后将来自存储库的结果发送回浏览器。

  RDS for Oracle 支持以下类型的侦听器：

  - 对于 Oracle APEX 版本 5.0 和更高版本，请使用 Oracle REST Data Services（ORDS）19.1 和更高版本。建议您使用支持的最新版本 Oracle APEX 和 ORDS。本文档仅出于向后兼容的目的介绍旧版本。
  - 对于 Oracle APEX 版本 4.1.1，可以使用 Oracle APEX Listener 版本 1.1.4。
  - 您可以使用 Oracle HTTP Server 和mod_plsql侦听器。

> **Note**
>
> Amazon RDS 不支持将嵌入式 PL/SQL 网关作为 Oracle APEX 侦听器的 Oracle XML DB HTTP 服务器。对于在 Internet 上运行的应用程序而言，Oracle 一般不建议使用嵌入 PL/SQL 网关。

有关这些侦听器类型的更多信息，请参阅 Oracle 文档中的关于选择 Web 侦听器。

将 APEX 或 APEX-DEV 选项添加到 RDS for Oracle 数据库实例时，Amazon RDS 仅安装 Oracle APEX 存储库。将您的侦听器安装在单独的主机上。

要求和限制

以下主题列出了 Oracle APEX 和 ORDS 的要求和限制。

Oracle APEX 版本要求

APEX 选项将数据库实例类中的存储用于数据库实例。下面是支持的 Oracle APEX 版本和大致的存储要求。

Oracle APEX 版本	存储需求	支持的 Oracle 数据库版本	备注
Oracle APEX 版本 24.2.v1	114 MiB	全部	此版本包括补丁 37885097：PSE BUNDLE FOR APEX 24.2（PSES ON TOP OF 24.2.0）、PATCH_VERSION 4。
Oracle APEX 版本 24.1.v1	112 MiB	全部	此版本包括补丁 36695709：PSE BUNDLE FOR APEX 24.1（PSES ON TOP OF 24.1.0）、PATCH_VERSION 3。如果您需要在 EC2 实例上安装完全相同的 APEX 映像版本，请下载补丁 37544819：24.1.3 PSE BUNDLE FOR APEX 24.1（PSES ON TOP OF 24.1.0）。
Oracle APEX 版本 23.2.v1	110 MiB	全部	此版本包括补丁 35895964：PSE BUNDLE FOR APEX 23.2（PSES ON TOP OF 23.2.0）、P

Oracle APEX 版本	存储需求	支持的 Oracle 数据库版本	备注
			ATCH_VERSION 6。如果您需要在 EC2 实例上安装完全相同的 APEX 映像版本，请下载补丁 37593125：23.2.6 PSE BUNDLE FOR APEX 23.2（PSES ON TOP OF 23.2.0）。
Oracle APEX 版本 23.1.v1	106 MiB	全部	此版本包括补丁 35283657：PSE BUNDLE FOR APEX 23.1（PSES ON TOP OF 23.1.0）、P ATCH_VERSION 2。
Oracle APEX 版本 22.2.v1	106MiB	全部	此版本包括补丁 34628174：PSE BUNDLE FOR APEX 22.2（PSES ON TOP OF 22.2.0）、P ATCH_VERSION 4。
Oracle APEX 版本 22.1.v1	124MiB	全部	此版本包括补丁 34020981：PSE BUNDLE FOR APEX 22.1（PSES ON TOP OF 22.1.0）、P ATCH_VERSION 6。
Oracle APEX 版本 21.2.v1	125 MiB	全部	此版本包括补丁 33420059：PSE BUNDLE FOR APEX 21.2（PSES ON TOP OF 21.2.0）、P ATCH_VERSION 8。
Oracle APEX 版本 21.1.v1	125 MiB	全部	此版本包括补丁 32598392：PSE BUNDLE FOR APEX 21.1、PATCH_VERSION 3。

Oracle APEX 版本 20.2.v1	148 MiB	全部，但 Oracle Database 21c 除外	此版本包括补丁 32006852：PSE BUNDLE FOR APEX 20.2、PATCH_VERSION 2020.11.12。您可以通过运行以下查询来查看补丁编号和日期：
			`SELECT PATCH_VERSION, PATCH_NUMBER`
			`FROM   APEX_PATCHES;`

Oracle APEX 版本	存储需求	支持的 Oracle 数据库版本	备注
Oracle APEX 版本 20.1.v1	173 MiB	全部，但 Oracle Database 21c 除外	此版本包括补丁 30990551：PSE BUNDLE FOR APEX 20.1、PATCH_VERSION 2020.07.15。
Oracle APEX 版本 19.2.v1	149 MiB	全部，但 Oracle Database 21c 除外	
Oracle APEX 版本 19.1.v1	148 MiB	全部，但 Oracle Database 21c 除外	

有关可下载的 Oracle APEX .zip 文件，请参阅 Oracle 网站上的 [Oracle APEX Prior Release Archives]。

Oracle APEX 和 ORDS 的先决条件

请注意使用 Oracle APEX 和 ORDS 的以下先决条件：

- 系统必须使用 Java 运行时环境（JRE）。
- 您的 Oracle 客户端安装必须包括以下内容：
  - 用于执行管理任务的 SQL*Plus 或 SQL Developer
  - 用于配置与 RDS for Oracle 数据库实例的连接的 Oracle Net Services

Oracle APEX 限制

您无法修改 `APEX_version` 用户账户，该账户由 Amazon RDS 管理。因此，您不能对该用户应用数据库配置文件或强制执行密码规则。`APEX_version` 的配置文件和密码设置由 Oracle 和 Amazon 预定义，旨在满足 Amazon RDS 的安全要求。

设置 Oracle APEX 和 Oracle Rest Data Services（ORDS）

以下主题列出了设置 Oracle APEX 和 ORDS 所需的步骤

主题

将 APEX 和 APEX-DEV 选项添加到数据库实例

要将 `APEX` 和 `APEX-DEV` 选项添加到 RDS for Oracle 数据库实例，请执行以下操作：

1. 创建新的选项组，或者复制或修改现有选项组。
2. 将 `APEX` 和 `APEX-DEV` 选项添加到该选项组。
3. 将选项组与数据库实例相关联。

如果添加 `APEX` 和 `APEX-DEV` 选项，则在数据库实例自动重新启动时会出现短暂中断。

> **Note**
>
> 安装 `APEX` 选项时，`APEX_MAIL` 可用。`APEX_MAIL` 包的执行权限授予 `PUBLIC`，因此您不需要 APEX 管理账户即可使用它。

将 APEX 和 APEX-DEV 选项添加到数据库实例

1. 确定您要使用的选项组。您可以创建新的选项组，或使用现有选项组。如果您想使用现有选项组，请跳到下一步。或者，通过以下设置创建自定义数据库选项组：

   a. 对于 Engine，请选择要使用的 Oracle 版本。所有版本均支持 `APEX` 和 `APEX-DEV` 选项。

   b. 对于主引擎版本，选择数据库实例的版本。

   有关更多信息，请参阅“创建选项组”。

2. 将选项添加到该选项组。如果您只希望部署 Oracle APEX 运行时环境，则仅添加 APEX 选项。要部署完整开发环境，则同时添加 APEX 和 APEX-DEV 选项。

   对于版本，选择要使用的 Oracle APEX 版本。

   > **Important**
   >
   > 如果向已经附加到一个或多个数据库实例的现有选项组添加 APEX 或 APEX-DEV 选项，则会发生短暂中断。在此中断期间，将自动重新启动所有数据库实例。

   有关添加选项的更多信息，请参阅 将选项添加到选项组。

3. 将选项组应用到新的或现有的数据库实例：

   - 对于新数据库实例，您可以在启动实例时应用选项组。有关更多信息，请参阅“创建 Amazon RDS 数据库实例”。
   - 对于现有数据库实例，您可以通过修改实例并附加新的选项组来应用选项组。如果您将 APEX 和 APEX-DEV 选项添加到现有数据库实例，则在数据库实例自动重新启动时会出现短暂中断。有关更多信息，请参阅 修改 Amazon RDS 数据库实例。

解锁数据库实例上的公有用户账户

在数据库实例上安装 APEX 或 APEX-DEV 选项后，请确保执行以下操作：

1. 更改 APEX_PUBLIC_USER 账户的密码。
2. 解锁此账户。

您可以使用 Oracle SQL*Plus 命令行实用程序执行该操作。以主用户身份连接到数据库实例，并发出以下命令。使用所选密码替换 new_password。

```
ALTER USER APEX_PUBLIC_USER IDENTIFIED BY new_password;
ALTER USER APEX_PUBLIC_USER ACCOUNT UNLOCK;
```

为 Oracle APEX 配置 RESTful 服务

要在 Oracle APEX 中配置 RESTful 服务（对 Oracle APEX 4.1.1.V1 不需要），请使用 SQL*Plus 以主用户的身份连接到数据库实例。执行此操作后，运行

rdsadmin.rdsadmin_run_apex_rest_config 存储过程。当您运行该存储过程时，请为以下用户提供密码：

- APEX_LISTENER
- APEX_REST_PUBLIC_USER

该存储过程运行 apex_rest_config.sql 脚本，从而为这些用户创建新的数据库账户。

> **Note**
>
> Oracle APEX 版本 4.1.1.v1 无需配置。仅对于此 Oracle APEX 版本，您不需要运行该存储过程。

以下命令运行该存储过程。

```
EXEC rdsadmin.rdsadmin_run_apex_rest_config('apex_listener_password',
 'apex_rest_public_user_password');
```

准备好在单独的主机上安装 ORDS

将 ORDS 安装在单独的主机上，例如 Amazon EC2 实例、公司的本地服务器或您的台式计算机。本节中的示例假设您的主机运行 Linux 且名为 myapexhost.example.com。

在安装 ORDS 之前，您需要创建非特权操作系统用户，然后下载并解压缩 Oracle APEX 安装文件。

为安装 ORDS 做准备

1. 以 myapexhost.example.com 身份登录到 root。
2. 创建一个非特权操作系统用户以控制侦听器安装。以下命令会创建一个名为 apexuser 的新用户。

   ```
   useradd -d /home/apexuser apexuser
   ```

   以下命令会为新用户分配密码。

   ```
   passwd apexuser;
   ```
3. 以 apexuser 身份登录 myapexhost.example.com，并从 Oracle 将 Oracle APEX 安装文件下载到您的 /home/apexuser 目录：

- http://www.oracle.com/technetwork/developer-tools/apex/downloads/index.html
- Oracle Application Express 早期版本存档

4. 在 /home/apexuser 目录中解压缩该文件。

   ```
   unzip apex_version.zip
   ```

   在解压缩该文件后，apex 目录中会有一个 /home/apexuser 目录。
5. 在您仍以 myapexhost.example.com 身份登录 apexuser 时，从 Oracle 网站将 Oracle REST Data Services 文件下载到您的 /home/apexuser 目录：http://www.oracle.com/technetwork/developer-tools/apex-listener/downloads/index.html

设置 Oracle APEX Listener

> **Note**
>
> Oracle APEX Listener 已被弃用。

Amazon RDS for Oracle 继续支持 Oracle APEX 版本 4.1.1 和 Oracle APEX Listener 版本 1.1.4。建议您使用支持的最新版本 Oracle APEX 和 ORDS。

将 Oracle APEX Listener 安装到独立的主机上，例如 Amazon EC2 实例、您公司的本地服务器或您的台式计算机上。假设您的主机名称为 myapexhost.example.com 且运行 Linux。

为安装 Oracle APEX Listener 做准备

在安装 Oracle APEX Listener 之前，您需要创建非特权操作系统用户，然后下载并解压缩 Oracle APEX 安装文件。

为安装 Oracle APEX Listener 做准备

1. 以 myapexhost.example.com 身份登录到 root。
2. 创建一个非特权操作系统用户以控制侦听器安装。以下命令会创建一个名为 apexuser 的新用户。

   ```
   useradd -d /home/apexuser apexuser
   ```

   以下命令会为新用户分配密码。

```
passwd apexuser;
```

3. 以 apexuser 身份登录 myapexhost.example.com，并从 Oracle 将 Oracle APEX 安装文件下载到您的 /home/apexuser 目录：

   - http://www.oracle.com/technetwork/developer-tools/apex/downloads/index.html
   - Oracle Application Express 早期版本存档

4. 在 /home/apexuser 目录中解压缩该文件。

```
unzip apex_<version>.zip
```

   在解压缩该文件后，apex 目录中会有一个 /home/apexuser 目录。

5. 当您仍以 myapexhost.example.com 身份登录 apexuser 时，从 Oracle 网站将 Oracle APEX Listener 文件下载到您的 /home/apexuser 目录。

安装和配置 Oracle APEX Listener

在使用 Oracle APEX 之前，您需要下载 apex.war 文件，使用 Java 安装 Oracle APEX Listener，然后启动侦听器。

安装和配置 Oracle APEX Listener

1. 创建一个基于 Oracle APEX Listener 的新目录并打开侦听器文件。

   运行以下代码：

```
mkdir /home/apexuser/apexlistener
cd /home/apexuser/apexlistener
unzip ../apex_listener.version.zip
```

2. 运行以下代码。

```
java -Dapex.home=./apex -Dapex.images=/home/apexuser/apex/images -Dapex.erase -
jar ./apex.war
```

3. 输入该程序提示的以下信息：

   - APEX Listener Administrator 用户名。默认值为 adminlistener。

- APEX Listener Administrator 密码。
- APEX Listener Manager 用户名。默认值为 managerlistener。
- APEX Listener Administrator 密码。

该程序将输出完成配置所需的 URL，如下所示。

```
INFO: Please complete configuration at: http://localhost:8080/apex/
listenerConfigure
Database is not yet configured
```

4. 让 Oracle APEX Listener 保持运行，以便您可以使用 Oracle Application Express。完成此配置过程后，可在后台运行侦听器。
5. 从 Web 浏览器中转到 Oracle APEX Listener 程序提供的 URL。显示 Oracle Application Express Listener 管理窗口。输入以下信息：
   - 用户名 – `APEX_PUBLIC_USER`
   - 密码 – APEX_PUBLIC_USER 的密码。这是您先前配置 Oracle APEX 存储库时指定的密码。有关更多信息，请参阅 解锁数据库实例上的公有用户账户。
   - 连接类型 – 基本
   - 主机名 – 您的 Amazon RDS 数据库实例的端点，例如 `mydb.f9rbfa893tft.us-east-1.rds.amazonaws.com`。
   - 端口 – 1521
   - SID – Amazon RDS 实例上的数据库名称，例如，`mydb`。
6. 选择 Apply。此时会显示 Oracle APEX 管理窗口。
7. 为 Oracle APEX `admin` 用户设置密码。为此，请使用 SQL*Plus 以主用户身份连接到数据库实例，然后运行以下命令。

```
EXEC rdsadmin.rdsadmin_util.grant_apex_admin_role;
grant APEX_ADMINISTRATOR_ROLE to master;
@/home/apexuser/apex/apxchpwd.sql
```

   使用主用户名替换 *`master`*。在 `apxchpwd.sql` 脚本提示您时，输入新的 `admin` 密码。
8. 返回浏览器中的 Oracle APEX 管理窗口，然后选择管理。接下来选择 Application Express Internal Administration。在提示您输入凭证时，请输入以下信息：
   - 用户名 – `admin`

- 密码 – 使用 `apxchpwd.sql` 脚本设置的密码

选择 Login，然后设置 `admin` 用户的新密码。

您的侦听器现在已可以使用。

配置 Oracle Rest Data Services（ORDS）

以下主题列出了 ORDS 21 和 22 的配置选项：

主题

安装并配置 ORDS 21 及更低版本

现在一切就绪，可以安装和配置用于 Oracle APEX 的 Oracle Rest Data Services (ORDS) 了。对于 Oracle APEX 版本 5.0 及更高版本，请使用 ORDS 版本 19.1 到 21。要了解如何安装 ORDS 22 及更高版本，请参阅安装和配置 ORDS 22 及更高版本。

将侦听器安装在单独的主机上，例如 Amazon EC2 实例、公司的本地服务器或您的台式计算机。对于本节中的示例，我们假定您的主机的名称为 `myapexhost.example.com`，并且您的主机运行的是 Linux。

安装并配置 ORDS 21 及更低版本以与 Oracle APEX 结合使用

1. 转至 Oracle REST Data Services，查看自述文件。确保安装了所需的 Java 版本。
2. 为 ORDS 安装创建一个新目录。

   ```
   mkdir /home/apexuser/ORDS
   cd /home/apexuser/ORDS
   ```

3. 从 Oracle REST data services 下载文件 `ords.`*`version.number`*`.zip`。
4. 将该文件解压缩到 `/home/apexuser/ORDS` 目录中。
5. 如果要将 ORDS 安装到多租户数据库中，请将如下命令行添加到文件 `/home/apexuser/ORDS/params/ords_params.properties`：

```
pdb.disable.lockdown=false
```

6. 为主用户授予所需的权限以安装 ORDS。

   在安装 Oracle APEX 的选项后，为主用户提供所需的权限以安装 ORDS 架构。您可以连接到数据库并运行以下命令以完成该操作。将 *MASTER_USER* 替换为主用户的大写名称。

   > **Important**
   >
   > 输入用户名时，除非您使用区分大小写的标识符创建了用户，否则请使用大写字母。例如，如果您运行 `CREATE USER myuser` 或 `CREATE USER MYUSER`，则数据字典将存储 `MYUSER`。但是，如果您在 `CREATE USER "MyUser"` 中使用双引号，则数据字典将存储 `MyUser`。有关更多信息，请参阅“[向 SYS 对象授予 SELECT 或 EXECUTE 权限](#)”。

   ```
   exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_OBJECTS', 'MASTER_USER',
    'SELECT', true);
   exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_ROLE_PRIVS', 'MASTER_USER',
    'SELECT', true);
   exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_TAB_COLUMNS', 'MASTER_USER',
    'SELECT', true);
   exec rdsadmin.rdsadmin_util.grant_sys_object('USER_CONS_COLUMNS', 'MASTER_USER',
    'SELECT', true);
   exec rdsadmin.rdsadmin_util.grant_sys_object('USER_CONSTRAINTS', 'MASTER_USER',
    'SELECT', true);
   exec rdsadmin.rdsadmin_util.grant_sys_object('USER_OBJECTS', 'MASTER_USER',
    'SELECT', true);
   exec rdsadmin.rdsadmin_util.grant_sys_object('USER_PROCEDURES', 'MASTER_USER',
    'SELECT', true);
   exec rdsadmin.rdsadmin_util.grant_sys_object('USER_TAB_COLUMNS', 'MASTER_USER',
    'SELECT', true);
   exec rdsadmin.rdsadmin_util.grant_sys_object('USER_TABLES', 'MASTER_USER',
    'SELECT', true);
   exec rdsadmin.rdsadmin_util.grant_sys_object('USER_VIEWS', 'MASTER_USER', 'SELECT',
    true);
   exec rdsadmin.rdsadmin_util.grant_sys_object('WPIUTL', 'MASTER_USER', 'EXECUTE',
    true);
   exec rdsadmin.rdsadmin_util.grant_sys_object('DBMS_SESSION', 'MASTER_USER',
    'EXECUTE', true);
   ```

```
exec rdsadmin.rdsadmin_util.grant_sys_object('DBMS_UTILITY', 'MASTER_USER',
 'EXECUTE', true);
```

> Note
>
> 这些命令适用于 ORDS 版本 19.1 和更高版本。

7. 使用下载的 ords.war 文件安装 ORDS 架构。

```
java -jar ords.war install advanced
```

程序会提示您输入以下信息。括号中为默认值。有关更多信息，请参阅 Oracle 文档中的 Oracle REST Data Services 简介。

- 输入用于存储配置数据的位置：

  输入 */apexuser/ORDS*。这是 ORDS 配置文件的位置。

- 指定要使用的数据库连接类型。输入 [1] 基本 [2] TNS [3] 自定义 URL [1] 的编号：

  选择所需的连接类型。

- 输入数据库服务器 [localhost] 的名称：*DB_instance_endpoint*

  选择默认值或输入正确的值。

- 输入数据库侦听器端口 [1521]：*DB_instance_port*

  选择默认值或输入正确的值。

- 输入 1 以指定数据库服务名称，或输入 2 以指定数据库 SID [1]：

  选择 2 以指定数据库 SID。

- 数据库 SID [xe]

  选择默认值或输入正确的值。

- 如果要验证/安装 Oracle REST Data Services 架构，请输入 1，或输入 2 以跳过此步骤 [1]：

  选择 1。此步骤将创建名为 ORDS_PUB_USER 的 Oracle REST Data Services 代理用户。

- 输入 ORDS_PUBLIC_USER 的数据库密码：

  输入密码，然后确认。

- 要求使用管理员权限登录以验证 Oracle REST Data Services 架构。

  输入管理员用户名：*master_user*

  输入 *master_user* 的数据库密码：*master_user_password*

  确认该密码：*master_user_password*

  > **Note**
  > 作为安全最佳实践，请指定除此处所示提示以外的密码。

- 输入 ORDS_METADATA [SYSAUX] 的默认表空间。

  输入 ORDS_METADATA [TEMP] 的临时表空间。

  输入 ORDS_PUBLIC_USER [USERS] 的默认表空间。

  输入 ORDS_PUBLIC_USER [TEMP] 的临时表空间。

- 如果要使用 PL/SQL 网关，请输入 1，或者输入 2 以跳过该步骤。如果使用 Oracle Application Express 或从 mod_plsql 迁移，您必须输入 1 [1]。

  选择默认值。

- 输入 PL/SQL 网关数据库用户名 [APEX_PUBLIC_USER]

  选择默认值。

- 输入 APEX_PUBLIC_USER 的数据库密码：

  输入密码，然后确认。

- 输入 1 以指定 Application Express RESTful Services 数据库用户（APEX_LISTENER、APEX_REST_PUBLIC_USER）的密码，或输入 2 以跳过此步骤 [1]：

  为 APEX 4.1.1.V1 选择 2；为所有其他 APEX 版本选择 1。

- [对于 APEX 4.1.1.v1 不需要] APEX_LISTENER 的数据库密码

  输入密码（如果需要），然后确认。

- [对于 APEX 4.1.1.v1 不需要] APEX_REST_PUBLIC_USER 的数据库密码

  输入密码（如果需要），然后确认。

- 输入一个数字以选择要启用的特征：

  输入 1 以启用所有特征：SQL 开发人员 Web、启用 REST 的 SQL 和数据库 API。
- 如果您希望在独立模式下启动，请输入 1；或输入 2 退出 [1]：

  输入 1。
- 输入 APEX 静态资源位置：

  如果已将 APEX 安装文件解压缩到 `/home/apexuser` 中，请输入 `/home/apexuser/apex/images`。否则，请输入 *`unzip_path`*`/apex/images`，其中 *`unzip_path`* 是解压缩文件的目录。
- 如果使用 HTTP，则输入 1；如果使用 HTTPS，则输入 2 [1]：

  如果您输入 1，请指定 HTTP 端口。如果您输入 2，请指定 HTTPS 端口和 SSL 主机名。HTTPS 选项提示您指定提供证书的方式：
  - 输入 1 以使用自签名证书。
  - 输入 2 以提供您自己的证书。如果输入 2，请指定 SSL 证书的路径和 SSL 证书私有密钥的路径。

8. 为 APEX `admin` 用户设置密码。为此，请使用 SQL*Plus 以主用户身份连接到数据库实例，然后运行以下命令。

   ```
   EXEC rdsadmin.rdsadmin_util.grant_apex_admin_role;
   grant APEX_ADMINISTRATOR_ROLE to master;
   @/home/apexuser/apex/apxchpwd.sql
   ```

   使用主用户名替换 *`master`*。在 `apxchpwd.sql` 脚本提示您时，输入新的 `admin` 密码。
9. 启动 ORDS 侦听器。运行以下代码。

   ```
   java -jar ords.war
   ```

   首次启动 ORDS 时，系统会提示您提供 APEX 静态资源的位置。此映像文件夹位于 APEX 安装目录下的 `/apex/images` 目录中。
10. 返回浏览器中的 Oracle APEX 管理窗口，然后选择管理。接下来选择 Application Express Internal Administration。在提示您输入凭证时，请输入以下信息：

    - 用户名 – `admin`

- 密码 – 使用 `apxchpwd.sql` 脚本设置的密码

  选择 Login，然后设置 `admin` 用户的新密码。

您的侦听器现在已可以使用。

安装和配置 ORDS 22 及更高版本

现在一切就绪，可以安装和配置用于 Oracle APEX 的 Oracle Rest Data Services (ORDS) 了。对于本节中的示例，我们假定单独主机的名称为 `myapexhost.example.com`，并且主机运行的是 Linux。ORDS 22 的说明与先前版本的说明不同。

安装并配置 ORDS 22 及更高版本以与 Oracle APEX 结合使用

1. 转至 [Oracle REST Data Services]，查看您计划下载的 ORDS 版本的自述文件。确保安装了所需的 Java 版本。
2. 为 ORDS 安装创建一个新目录。

   ```
   mkdir /home/apexuser/ORDS
   cd /home/apexuser/ORDS
   ```

3. 从 [Oracle REST Data Services] 下载文件 `ords.`*`version.number`*`.zip` 或 `ords-latest.zip`。
4. 将该文件解压缩到 `/home/apexuser/ORDS` 目录中。
5. 为主用户授予所需的权限以安装 ORDS。

   在安装 `APEX` 选项后，为主用户提供所需的权限以安装 ORDS 架构。您可以登录到数据库并运行以下命令来完成该操作。将 *`MASTER_USER`* 替换为主用户的大写名称。

   > **Important**
   >
   > 输入用户名时，除非您使用区分大小写的标识符创建了用户，否则请使用大写字母。例如，如果您运行 `CREATE USER myuser` 或 `CREATE USER MYUSER`，则数据字典将存储 `MYUSER`。但是，如果您在 `CREATE USER "MyUser"` 中使用双引号，则数据字典将存储 `MyUser`。有关更多信息，请参阅 [向 SYS 对象授予 SELECT 或 EXECUTE 权限]。

```
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_OBJECTS', 'MASTER_USER',
 'SELECT', true);
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_ROLE_PRIVS', 'MASTER_USER',
 'SELECT', true);
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_TAB_COLUMNS', 'MASTER_USER',
 'SELECT', true);
exec rdsadmin.rdsadmin_util.grant_sys_object('USER_CONS_COLUMNS', 'MASTER_USER',
 'SELECT', true);
exec rdsadmin.rdsadmin_util.grant_sys_object('USER_CONSTRAINTS', 'MASTER_USER',
 'SELECT', true);
exec rdsadmin.rdsadmin_util.grant_sys_object('USER_OBJECTS', 'MASTER_USER',
 'SELECT', true);
exec rdsadmin.rdsadmin_util.grant_sys_object('USER_PROCEDURES', 'MASTER_USER',
 'SELECT', true);
exec rdsadmin.rdsadmin_util.grant_sys_object('USER_TAB_COLUMNS', 'MASTER_USER',
 'SELECT', true);
exec rdsadmin.rdsadmin_util.grant_sys_object('USER_TABLES', 'MASTER_USER',
 'SELECT', true);
exec rdsadmin.rdsadmin_util.grant_sys_object('USER_VIEWS', 'MASTER_USER', 'SELECT',
 true);
exec rdsadmin.rdsadmin_util.grant_sys_object('WPIUTL', 'MASTER_USER', 'EXECUTE',
 true);
exec rdsadmin.rdsadmin_util.grant_sys_object('DBMS_SESSION', 'MASTER_USER',
 'EXECUTE', true);
exec rdsadmin.rdsadmin_util.grant_sys_object('DBMS_UTILITY', 'MASTER_USER',
 'EXECUTE', true);

exec rdsadmin.rdsadmin_util.grant_sys_object('DBMS_LOB', 'MASTER_USER', 'EXECUTE',
 true);
exec rdsadmin.rdsadmin_util.grant_sys_object('DBMS_ASSERT', 'MASTER_USER',
 'EXECUTE', true);
exec rdsadmin.rdsadmin_util.grant_sys_object('DBMS_OUTPUT', 'MASTER_USER',
 'EXECUTE', true);
exec rdsadmin.rdsadmin_util.grant_sys_object('DBMS_SCHEDULER', 'MASTER_USER',
 'EXECUTE', true);
exec rdsadmin.rdsadmin_util.grant_sys_object('HTP', 'MASTER_USER', 'EXECUTE',
 true);
exec rdsadmin.rdsadmin_util.grant_sys_object('OWA', 'MASTER_USER', 'EXECUTE',
 true);
exec rdsadmin.rdsadmin_util.grant_sys_object('WPG_DOCLOAD', 'MASTER_USER',
 'EXECUTE', true);
```

```
exec rdsadmin.rdsadmin_util.grant_sys_object('DBMS_CRYPTO', 'MASTER_USER',
 'EXECUTE', true);
exec rdsadmin.rdsadmin_util.grant_sys_object('DBMS_METADATA', 'MASTER_USER',
 'EXECUTE', true);
exec rdsadmin.rdsadmin_util.grant_sys_object('DBMS_SQL', 'MASTER_USER', 'EXECUTE',
 true);
exec rdsadmin.rdsadmin_util.grant_sys_object('UTL_SMTP', 'MASTER_USER', 'EXECUTE',
 true);
exec rdsadmin.rdsadmin_util.grant_sys_object('DBMS_NETWORK_ACL_ADMIN',
 'MASTER_USER', 'EXECUTE', true);
exec rdsadmin.rdsadmin_util.grant_sys_object('SESSION_PRIVS', 'MASTER_USER',
 'SELECT', true);
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_USERS', 'MASTER_USER', 'SELECT',
 true);
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_NETWORK_ACL_PRIVILEGES',
 'MASTER_USER', 'SELECT', true);
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_NETWORK_ACLS', 'MASTER_USER',
 'SELECT', true);
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_REGISTRY', 'MASTER_USER',
 'SELECT', true);
```

> **Note**
>
> 上述命令适用于 ORDS 22 及更高版本。

6. 使用下载的 `ords` 脚本安装 ORDS 架构。指定要包含配置文件和日志文件的目录。Oracle Corporation 建议不要将这些目录放在包含 ORDS 产品软件的目录中。

```
mkdir -p /home/apexuser/ords_config /home/apexuser/ords_logs

/home/apexuser/ORDS/bin/ords \
  --config /home/apexuser/ords_config \
  install --interactive --log-folder /home/apexuser/ords_logs
```

对于运行容器数据库（CDB）架构的数据库实例，请使用 ORDS 23.2 及更高版本，并在安装 ORDS 时传递 `--pdb-skip-disable-lockdown` 参数。

```
/home/apexuser/ORDS/bin/ords \
  --config /home/apexuser/ords_config \
  install --interactive --log-folder /home/apexuser/ords_logs --pdb-skip-disable-
lockdown
```

程序会提示您输入以下信息。括号中为默认值。有关更多信息，请参阅 Oracle 文档中的 Oracle REST Data Services 简介。

- `Choose the type of installation:`

  选择 **2** 可在数据库中安装 ORDS 架构，并在本地 ORDS 配置文件中创建数据库连接池。

- `Specify the database connection type to use. Enter number for [1] Basic [2] TNS [3] Custom URL:`

  选择所需的连接类型。此示例假设您选择 **1**。

- `Enter the name of the database server [localhost]:` ***`DB_instance_endpoint`***

  选择默认值 或输入正确的值。

- `Enter the database listener port [1521]:` ***`DB_instance_port`***

  选择默认值 **1521** 或输入正确的值。

- `Enter the database service name [orcl]:`

  输入 RDS for Oracle 数据库实例所使用的数据库名称。

- `Provide database user name with administrator privileges`

  输入 RDS for Oracle 数据库实例的主用户名。

- `Enter the database password for [username]:`

  输入 RDS for Oracle 数据库实例的主用户密码。

- `Enter the default tablespace for ORDS_METADATA and ORDS_PUBLIC_USER [SYSAUX]:`

- `Enter the temporary tablespace for ORDS_METADATA [TEMP]. Enter the default tablespace for ORDS_PUBLIC_USER [USERS]. Enter the temporary tablespace for ORDS_PUBLIC_USER [TEMP].`

- `Enter a number to select additional feature(s) to enable [1]:`

- `Enter a number to configure and start ORDS in standalone mode [1]:`

  选择 **2** 可在独立模式下立即跳过启动 ORDS。

- `Enter a number to select the protocol [1] HTTP`

- `Enter the HTTP port [8080]:`
- `Enter the APEX static resources location:`

  输入 Oracle APEX 安装文件的路径（`/home/apexuser/apex/images`）。

7. 为 Oracle APEX `admin` 用户设置密码。为此，请使用 SQL*Plus 以主用户身份连接到数据库实例，然后运行以下命令。

```
EXEC rdsadmin.rdsadmin_util.grant_apex_admin_role;
grant APEX_ADMINISTRATOR_ROLE to master;
@/home/apexuser/apex/apxchpwd.sql
```

   使用主用户名替换 *`master`*。在 `apxchpwd.sql` 脚本提示您时，输入新的 `admin` 密码。

8. 使用带有 `serve` 命令的 `ords` 脚本在独立模式下运行 ORDS。对于生产部署，可以考虑使用支持的 Java EE 应用程序服务器，例如 Apache Tomcat 或 Oracle WebLogic 服务器。有关更多信息，请参阅 Oracle 数据库文档中的 Deploying and Monitoring Oracle REST Data Services。

```
/home/apexuser/ORDS/bin/ords \
  --config /home/apexuser/ords_config serve \
  --port 8193 \
  --apex-images /home/apexuser/apex/images
```

   如果 ORDS 正在运行但无法访问 Oracle APEX 安装，您可能会看到以下错误，尤其是在非 CDB 实例上。

```
The procedure named apex_admin could not be accessed, it may not be declared,
 or the user executing this request may not have been granted execute privilege
 on the procedure, or a function specified by security.requestValidationFunction
 configuration property has prevented access.
```

   要修复此错误，请通过运行带有 `config` 命令的 `ords` 脚本来更改 ORDS 使用的请求验证函数。默认情况下，ORDS 使用 `ords_util.authorize_plsql_gateway` 程序包，只有 CDB 实例支持该程序包。对于非 CDB 实例，您可以将此程序包更改为 `wwv_flow_epg_include_modules.authorize` 软件包。有关为您的用例配置正确的请求验证函数的最佳实践，请参阅 Oracle 数据库文档和 Oracle Support。

9. 返回浏览器中的 Oracle APEX 管理窗口，然后选择管理。接下来选择 Application Express Internal Administration。在提示您输入凭证时，请输入以下信息：

- 用户名 – `admin`
- 密码 – 使用 `apxchpwd.sql` 脚本设置的密码

选择 Login，然后设置 `admin` 用户的新密码。

您的侦听器现在已可以使用。

升级和移除 Oracle APEX

要升级和移除 Oracle APEX，请遵循本主题中的说明：

主题

升级 Oracle APEX 版本

> **Important**
>
> 在升级 Oracle APEX 之前，请对数据库实例进行备份。有关更多信息，请参阅为 Amazon RDS 的单可用区数据库实例创建数据库快照和测试 Oracle 数据库升级。

要将 Oracle APEX 与数据库实例一起升级，请执行以下操作：

- 为数据库实例的升级后版本创建一个新的选项组。
- 将 `APEX` 和 `APEX-DEV` 选项的升级后版本添加到新的选项组。确保包括数据库实例使用的任何其他选项。有关更多信息，请参阅"选项组注意事项"。
- 在升级数据库实例时，请为升级后的数据库实例指定该新选项组。

在升级 Oracle APEX 的版本后，早期版本的 Oracle APEX 架构可能仍在您的数据库中。如果您不再需要旧的 Oracle APEX 架构，可在升级后从数据库中将其删除。

如果您升级 Oracle APEX 版本，并且在先前的 Oracle APEX 版本中未配置 RESTful 服务，我们建议您配置 RESTful 服务。有关更多信息，请参阅 为 Oracle APEX 配置 RESTful 服务。

在某些情况下，当您计划对数据库实例进行主要版本升级时，您可能会发现您使用的是与目标数据库版本不兼容的 Oracle APEX 版本。在这些情况下，您可以在升级数据库实例之前升级 Oracle APEX 的版本。先升级 Oracle APEX 可以减少升级数据库实例所需的时间。

> **Note**
>
> 在升级 Oracle APEX 后，安装并配置用于已升级版本的侦听器。有关说明，请参阅设置 Oracle APEX Listener。

移除 APEX 和 APEX-DEV 选项

您可以从数据库实例中移除 APEX 和 APEX-DEV 选项。要从数据库实例中移除这些选项，请执行下列操作之一：

- 要从多个数据库实例中移除 APEX 和 APEX-DEV 选项，请从其所属的选项组中移除这些选项。此更改会影响使用该选项组的所有数据库实例。如果从已附加到多个数据库实例的选项组中移除选项，数据库实例重新启动时会发生短暂中断。

  有关更多信息，请参阅 从选项组中删除选项。
- 要从单个数据库实例中移除 APEX 和 APEX-DEV 选项，请修改数据库实例，并指定不包含这些选项的另一个选项组。您可以指定默认 (空) 选项组，或指定其他自定义选项组。如果移除选项，数据库实例在自动重新启动时会出现短暂中断。

  有关更多信息，请参阅 修改 Amazon RDS 数据库实例。

在从数据库实例中移除 APEX 和 APEX-DEV 选项时，将从数据库中移除 APEX 架构。

Amazon EFS 集成

Amazon Elastic File System（Amazon EFS）提供无服务器的完全弹性文件存储，因此，您无需预调配或管理存储容量和性能，即可共享文件数据。使用 Amazon EFS，您可以创建文件系统，然后通过 NFS 版本 4.0 和 4.1（NFSv4）协议将其安装在您的 VPC 中。然后，您可以像任何其他兼容 POSIX 的文件系统一样使用 EFS 文件系统。有关一般信息，请参阅什么是 Amazon Elastic File System？和 Amazon 博客将 Amazon RDS for Oracle 与 Amazon EFS 集成。

主题

Amazon EFS 集成概述

使用 Amazon EFS，您可以在 RDS for Oracle 数据库实例和 EFS 文件系统之间传输文件。例如，您可以使用 EFS 来支持以下使用案例：

- 在应用程序和多个数据库服务器之间共享文件系统。
- 为与迁移相关的文件（包括可传输的表空间数据文件）创建共享目录。有关更多信息，请参阅 使用 Oracle 可传输表空间进行迁移。
- 存储和共享存档的重做日志文件，无需在服务器上分配额外的存储空间。
- 使用 Oracle 数据库实用程序（例如 `UTL_FILE`）读取和写入文件。

Amazon EFS 集成的优势

当您选择 EFS 文件系统而不是替代的数据传输解决方案时，您将获得以下优势：

- 您可以在 Amazon EFS 和 RDS for Oracle 数据库实例之间传输 Oracle Data Pump 文件。您无需在本地复制这些文件，因为 Data Pump 直接从 EFS 文件系统导入。有关更多信息，请参阅 将数据导入到 Amazon RDS 上的 Oracle。
- 数据迁移比使用数据库链接更快。
- 避免在 RDS for Oracle 数据库实例上分配存储空间来存放文件。
- EFS 文件系统可以自动扩展存储，无需您进行预调配。
- Amazon EFS 集成没有最低费用或设置成本。您仅需按实际用量付费。
- Amazon EFS 集成支持两种形式的加密：传输中数据加密和静态加密。默认情况下，使用 TLS 版本 1.2 启用传输中数据加密。您可以在创建 Amazon EFS 文件系统时启用静态数据加密。有关更多信息，请参阅 Amazon Elastic File System User Guide（《Amazon Elastic File System 用户指南》）中的 Encrypting data at rest（加密静态数据）。

Amazon EFS 集成的要求

确保您满足以下要求：

- 数据库必须运行数据库版本 19.0.0.0.ru-2022-07.rur-2022-07.r1 或更高版本。
- 数据库实例和 EFS 文件系统必须位于同一个 Amazon Web Services 区域、VPC 和 Amazon Web Services 账户中。RDS for Oracle 不支持 EFS 的跨账户和跨区域访问。
- VPC 必须同时启用了 DNS 解析和 DNS 主机名。有关更多信息，请参阅《Amazon Virtual Private Cloud 用户指南》中的 VPC 中的 DNS 属性。
- EFS 文件系统必须使用 Standard 或 Standard-IA 存储类。
- 如果您在 `mount` 命令中使用 DNS 名称，请确保 VPC 配置为使用由 Amazon 提供的 DNS 服务器。不支持自定义 DNS 服务器。
- 必须使用非 RDS 解决方案来备份 EFS 文件系统。RDS for Oracle 不支持 EFS 文件系统的自动备份或手动数据库快照。有关更多信息，请参阅备份您的 Amazon EFS 文件系统。

为 RDS for Oracle 与 Amazon EFS 集成配置网络权限

要让 RDS for Oracle 与 Amazon EFS 集成，请确保您的数据库实例具有网络访问权限，可访问 EFS 文件系统。有关更多信息，请参阅《Amazon Elastic File System 用户指南》中的 控制 NFS 客户端对 Amazon EFS 文件系统的网络访问权限。

主题

- 使用安全组控制网络访问权限

- 使用文件系统策略控制网络访问权限

使用安全组控制网络访问权限

您可以使用 VPC 安全组等网络层安全机制控制数据库实例对 EFS 文件系统的访问权限。要允许访问数据库实例的 EFS 文件系统，请确保您的 EFS 文件系统满足以下要求：

- EFS 挂载目标存在于 RDS for Oracle 数据库实例使用的每个可用区中。

  EFS 挂载目标提供您可以在其中挂载 EFS 文件系统的 NFSv4 端点的 IP 地址。您使用其 DNS 名称挂载文件系统，该名称将解析为数据库实例的可用区使用的 EFS 挂载目标的 IP 地址。

  您可以将不同可用区中的数据库实例配置为使用相同的 EFS 文件系统。对于多可用区，部署中的每个可用区都需要一个挂载点。您可能需要将数据库实例移到其他可用区。出于这些原因，我们建议您在 VPC 的每个可用区中创建一个 EFS 挂载点。默认情况下，当您使用控制台创建新的 EFS 文件系统时，RDS 会为所有可用区创建挂载目标。
- 安全组已连接到挂载目标。
- 安全组有一条入站规则，允许在 TCP/2049（类型 NFS）上使用 RDS for Oracle 数据库实例的网络子网或安全组。

有关更多信息，请参阅《Amazon Elastic File System 用户指南》中的创建 Amazon EFS 文件系统和创建和管理 EFS 挂载目标和安全组。

使用文件系统策略控制网络访问权限

Amazon EFS 与 RDS for Oracle 的集成使用默认（空）EFS 文件系统策略。默认策略不使用 IAM 进行身份验证。相反，它向可以使用挂载目标连接到文件系统的任何匿名客户端授予完全访问权限。当用户配置的文件系统策略不生效时（包括在创建文件系统时），默认策略将生效。有关更多信息，请参阅《Amazon Elastic File System 用户指南》中的默认 EFS 文件系统策略。

要增强所有客户端（包括 RDS for Oracle）的 EFS 文件系统的访问权限，您可以配置 IAM 权限。使用这种方法，您可以创建文件系统策略。有关更多信息，请参阅《Amazon Elastic File System 用户指南》中的创建文件系统策略。

为 RDS for Oracle 与 Amazon EFS 集成配置 IAM 权限

默认情况下，Amazon EFS 集成功能不使用 IAM 角色：`USE_IAM_ROLE` 选项设置为 `FALSE`。要将 RDS for Oracle 与 Amazon EFS 和 IAM 角色集成，您的数据库实例必须具有可以访问 Amazon EFS 文件系统的 IAM 权限。

主题

- 步骤 1：为您的数据库实例创建 IAM 角色并附加策略
- 步骤 2：为您的 Amazon EFS 文件系统创建文件系统策略
- 步骤 3：将您的 IAM 角色与 RDS for Oracle 数据库实例关联

步骤 1：为您的数据库实例创建 IAM 角色并附加策略

在此步骤中，您将为 RDS for Oracle 数据库实例创建一个角色，以允许 Amazon RDS 访问您的 EFS 文件系统。

控制台

创建 IAM 角色以允许 Amazon RDS 访问 EFS 文件系统

1. 打开 IAM 管理控制台。
2. 在导航窗格中，选择角色。
3. 选择 Create role（创建角色）。
4. 对于 Amazon service (亚马逊云科技服务)，选择 RDS。
5. 对于 Select your use case (选择您的用例)，选择 RDS – Add Role to Database (RDS - 将角色添加到数据库)。
6. 选择下一步。
7. 不要添加任何权限策略。选择下一步。
8. 将 Role name (角色名称) 设置为您的 IAM 角色的名称，例如，`rds-efs-integration-role`。您也可以添加可选的描述值。
9. 选择 Create role（创建角色）。

Amazon CLI

要限制服务对特定资源的权限，我们建议在基于资源的信任关系中使用 `aws:SourceArn` 和 `aws:SourceAccount` 全局条件上下文键。这是防范混淆代理问题最有效的方法。

您可以使用这两个全局条件上下文键并让 `aws:SourceArn` 值包含账户 ID。在这种情况下，当 `aws:SourceAccount` 值和 `aws:SourceArn` 值中的账户使用相同策略语句时，确保二者使用相同的账户 ID。

- 如果您想对单个资源进行跨服务访问，请使用 `aws:SourceArn`。

- 如果您想允许该账户中的任何资源与跨服务使用操作相关联，请使用 aws:SourceAccount。

在信任关系中，请务必使用 aws:SourceArn 全局条件上下文键和访问角色资源的完整 Amazon Resource Name (ARN)。

以下 Amazon CLI 命令会创建名为 *rds-efs-integration-role* 的角色来实现此目的。

Example

对于 Linux、macOS 或 Unix：

```
aws iam create-role \
   --role-name rds-efs-integration-role \
   --assume-role-policy-document '{
     "Version": "2012-10-17",
     "Statement": [
       {
         "Effect": "Allow",
         "Principal": {
            "Service": "rds.amazonaws.com"
          },
         "Action": "sts:AssumeRole",
         "Condition": {
             "StringEquals": {
                 "aws:SourceAccount": my_account_ID,
                 "aws:SourceArn": "arn:aws:rds:Region:my_account_ID:db:dbname"
             }
         }
       }
     ]
   }'
```

对于 Windows：

```
aws iam create-role ^
   --role-name rds-efs-integration-role ^
   --assume-role-policy-document '{
     "Version": "2012-10-17",
     "Statement": [
       {
         "Effect": "Allow",
         "Principal": {
```

```
            "Service": "rds.amazonaws.com"
          },
         "Action": "sts:AssumeRole",
         "Condition": {
             "StringEquals": {
                 "aws:SourceAccount": my_account_ID,
                 "aws:SourceArn": "arn:aws:rds:Region:my_account_ID:db:dbname"
             }
         }
       }
     ]
   }'
```

有关更多信息，请参阅《IAM 用户指南》中的创建向 IAM 用户委派权限的角色。

步骤 2：为您的 Amazon EFS 文件系统创建文件系统策略

在此步骤中，您将为 EFS 文件系统创建文件系统策略。

创建或编辑 EFS 文件系统策略

1. 打开 EFS 管理控制台。
2. 选择 File Systems (文件系统)。
3. 在 File systems（文件系统）页面上，选择要为其编辑或创建文件系统策略的文件系统。将显示该文件系统的详细信息页面。
4. 选择 File system policy（文件系统策略）选项卡。

   如果策略为空，则使用默认 EFS 文件系统策略。有关更多信息，请参阅《Amazon Elastic File System 用户指南》中的默认 EFS 文件系统策略。
5. 选择编辑。此时将显示 File system policy (文件系统策略) 页面。
6. 在 Policy editor（策略编辑器）中，输入如下所示的策略，然后选择 Save（保存）。

```
{
    "Version": "2012-10-17",
    "Id": "ExamplePolicy01",
    "Statement": [
        {
            "Sid": "ExampleStatement01",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::123456789012:role/rds-efs-integration-role"
```

```
            },
            "Action": [
                "elasticfilesystem:ClientMount",
                "elasticfilesystem:ClientWrite",
                "elasticfilesystem:ClientRootAccess"
            ],
            "Resource": "arn:aws:elasticfilesystem:us-east-1:123456789012:file-
system/fs-1234567890abcdef0"
        }
    ]
}
```

步骤 3：将您的 IAM 角色与 RDS for Oracle 数据库实例关联

在此步骤中，您将您的 IAM 角色与数据库实例关联。请注意以下要求：

- 您必须具有访问 IAM 角色的权限，并向此角色附加所需的 Amazon EFS 权限策略。
- 您一次只能将一个 IAM 角色与 RDS for Oracle 数据库实例关联。
- 实例的状态必须为 Available（可用）。

有关更多信息，请参阅《Amazon Elastic File System 用户指南》中的 Amazon EFS 的身份和访问权限管理。

控制台

将您的 IAM 角色与 RDS for Oracle 数据库实例关联

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 选择数据库。
3. 如果数据库实例不可用，请选择 Actions (操作)，然后选择 Start (启动)。当实例状态显示 Started (已启动) 时，请转至下一步。
4. 选择 Oracle 数据库实例名称以显示其详细信息。
5. 在 Connectivity & security（连接性和安全性）选项卡上，向下滚动到页面底部的 Manage IAM roles（管理 IAM 角色）部分。
6. 选择要添加到 Add IAM roles to this instance（将 IAM 角色添加到此实例）部分的角色。
7. 对于 Feature（特征），选择 EFS_INTEGRATION。

8. 选择 Add role (添加角色)。

Amazon CLI

以下 Amazon CLI 命令将角色添加到名为 *mydbinstance* 的 Oracle 数据库实例。

Example

对于 Linux、macOS 或 Unix：

```
aws rds add-role-to-db-instance \
   --db-instance-identifier mydbinstance \
   --feature-name EFS_INTEGRATION \
   --role-arn your-role-arn
```

对于 Windows：

```
aws rds add-role-to-db-instance ^
   --db-instance-identifier mydbinstance ^
   --feature-name EFS_INTEGRATION ^
   --role-arn your-role-arn
```

将 *your-role-arn* 替换为您在上一步中记下的角色 ARN。必须为 `EFS_INTEGRATION` 选项指定 `--feature-name`。

添加 EFS_INTEGRATION 选项

要将 Amazon RDS for Oracle 与 Amazon EFS 集成，您的数据库实例必须与包括 `EFS_INTEGRATION` 选项的选项组关联。

属于同一选项组的多个 Oracle 数据库实例共享相同的 EFS 文件系统。不同的数据库实例可以访问相同的数据，但可以通过使用不同的 Oracle 目录来划分访问权限。有关更多信息，请参阅[在 RDS for Oracle 和 Amazon EFS 文件系统之间传输文件](#)。

控制台

为 Amazon EFS 集成配置选项组

1. 创建新选项组或识别现有选项组，然后您可以将 `EFS_INTEGRATION` 选项添加到其中。

   有关创建选项组的信息，请参阅 [创建选项组](#)。

2. 将 `EFS_INTEGRATION` 选项添加到该选项组。您需要指定 `EFS_ID` 文件系统 ID 并设置 `USE_IAM_ROLE` 标志。

   有关更多信息，请参阅 将选项添加到选项组。

3. 通过以下任一方式将选项组与数据库实例关联：

   - 创建新的 Oracle 数据库实例并将选项组与其关联。有关创建数据库实例的信息，请参阅创建 Amazon RDS 数据库实例。
   - 修改 Oracle 数据库实例以将选项组与其关联。有关修改 Oracle 数据库实例的信息，请参阅 修改 Amazon RDS 数据库实例。

Amazon CLI

为 EFS 集成配置选项组

1. 创建新选项组或识别现有选项组，然后您可以将 `EFS_INTEGRATION` 选项添加到其中。

   有关创建选项组的信息，请参阅 创建选项组。

2. 将 `EFS_INTEGRATION` 选项添加到该选项组。

   例如，以下 Amazon CLI 命令将 `EFS_INTEGRATION` 选项添加到名为 **`myoptiongroup`** 的选项组。

   Example

   对于 Linux、macOS 或 Unix：

```
aws rds add-option-to-option-group \
    --option-group-name myoptiongroup \
    --options "OptionName=EFS_INTEGRATION,OptionSettings=\
    [{Name=EFS_ID,Value=fs-1234567890abcdef0},{Name=USE_IAM_ROLE,Value=TRUE}]"
```

   对于 Windows：

```
aws rds add-option-to-option-group ^
    --option-group-name myoptiongroup ^
    --options "OptionName=EFS_INTEGRATION,OptionSettings=^
    [{Name=EFS_ID,Value=fs-1234567890abcdef0},{Name=USE_IAM_ROLE,Value=TRUE}]"
```

3. 通过以下任一方式将选项组与数据库实例关联：

- 创建新的 Oracle 数据库实例并将选项组与其关联。有关创建数据库实例的信息，请参阅创建 Amazon RDS 数据库实例。
- 修改 Oracle 数据库实例以将选项组与其关联。有关修改 Oracle 数据库实例的信息，请参阅 修改 Amazon RDS 数据库实例。

配置 Amazon EFS 文件系统权限

默认情况下，对于新创建的 EFS 文件系统，只有根用户（UID `0`）具有读取、写入和执行权限。为了让其他用户也能修改文件系统，根用户必须明确授予他们访问权限。RDS for Oracle 数据库实例的用户属于 `others` 类别。有关更多信息，请参阅《Amazon Elastic File System 用户指南》中的在网络文件系统（NFS）级别处理用户、组和权限。

要允许您的 RDS for Oracle 数据库实例在 EFS 文件系统上读取和写入文件，请执行以下操作：

- 在您的 Amazon EC2 或本地实例上本地挂载 EFS 文件系统。
- 配置精细权限。

例如，要授予 `other` 用户写入 EFS 文件系统根目录的权限，请在此目录上运行 `chmod 777`。有关更多信息，请参阅《Amazon Elastic File System 用户指南》中的示例 Amazon EFS 文件系统使用案例和权限。

在 RDS for Oracle 和 Amazon EFS 文件系统之间传输文件

要在 RDS for Oracle 实例和 Amazon EFS 文件系统之间传输文件，请至少创建一个 Oracle 目录并配置 EFS 文件系统权限以控制数据库实例访问权限。

主题

- 创建 Oracle 目录
- 向 EFS 文件系统传输数据和从其中传输数据：示例

创建 Oracle 目录

要创建 Oracle 目录，请使用过程 `rdsadmin.rdsadmin_util.create_directory_efs`。 过程具有以下参数。

参数名称	数据类型	默认值	必需	描述
p_directory_name	VARCHAR	–	是	Oracle 目录的名称。
p_path_on_efs	VARCHAR	–	是	EFS 文件系统上的路径。路径名的前缀使用模式 /rdsefs-*fsid*/，其中 *fsid* 是 EFS 文件系统 ID 的占位符。 例如，如果您的 EFS 文件系统名为 fs-1234567890abcdef0 ，并且您在此文件系统上创建了一个名为 mydir 的子目录，则可以指定以下值： `/rdsefs-fs-1234567890abcdef0/mydir`

假设您在 EFS 文件系统 fs-1234567890abcdef0 上创建了一个名为 /datapump1 的子目录。以下示例创建一个 Oracle 目录 DATA_PUMP_DIR_EFS，该目录指向 EFS 文件系统上的 /datapump1 目录。p_path_on_efs 参数的文件系统路径值以字符串 /rdsefs- 为前缀。

```
BEGIN
  rdsadmin.rdsadmin_util.create_directory_efs(
    p_directory_name => 'DATA_PUMP_DIR_EFS',
    p_path_on_efs    => '/rdsefs-fs-1234567890abcdef0/datapump1');
END;
/
```

向 EFS 文件系统传输数据和从其中传输数据：示例

以下示例使用 Oracle Data Pump 将名为 MY_TABLE 的表导出到文件 datapump.dmp。此文件位于 EFS 文件系统上。

```
DECLARE
  v_hdnl NUMBER;
BEGIN
  v_hdnl := DBMS_DATAPUMP.OPEN(operation => 'EXPORT', job_mode => 'TABLE',
 job_name=>null);
  DBMS_DATAPUMP.ADD_FILE(
    handle     => v_hdnl,
```

```
    filename  => 'datapump.dmp',
    directory => 'DATA_PUMP_DIR_EFS',
    filetype  => dbms_datapump.ku$_file_type_dump_file);
  DBMS_DATAPUMP.ADD_FILE(
    handle    => v_hdnl,
    filename  => 'datapump-exp.log',
    directory => 'DATA_PUMP_DIR_EFS',
    filetype  => dbms_datapump.ku$_file_type_log_file);
  DBMS_DATAPUMP.METADATA_FILTER(v_hdnl,'NAME_EXPR','IN (''MY_TABLE'')');
  DBMS_DATAPUMP.START_JOB(v_hdnl);
END;
/
```

以下示例使用 Oracle Data Pump 从文件 `datapump.dmp` 中导出名为 `MY_TABLE` 的表。此文件位于 EFS 文件系统上。

```
DECLARE
  v_hdnl NUMBER;
BEGIN
  v_hdnl := DBMS_DATAPUMP.OPEN(
    operation => 'IMPORT',
    job_mode  => 'TABLE',
    job_name  => null);
  DBMS_DATAPUMP.ADD_FILE(
    handle    => v_hdnl,
    filename  => 'datapump.dmp',
    directory => 'DATA_PUMP_DIR_EFS',
    filetype  => dbms_datapump.ku$_file_type_dump_file );
  DBMS_DATAPUMP.ADD_FILE(
    handle    => v_hdnl,
    filename  => 'datapump-imp.log',
    directory => 'DATA_PUMP_DIR_EFS',
    filetype  => dbms_datapump.ku$_file_type_log_file);
  DBMS_DATAPUMP.METADATA_FILTER(v_hdnl,'NAME_EXPR','IN (''MY_TABLE'')');
  DBMS_DATAPUMP.START_JOB(v_hdnl);
END;
/
```

有关更多信息，请参阅 将数据导入到 Amazon RDS 上的 Oracle。

删除 EFS_INTEGRATION 选项

删除 `EFS_INTEGRATION` 选项的步骤取决于您是从多个数据库实例还是单个实例中删除该选项。

数据库实例数	操作	相关信息
多个	从数据库实例所属的选项组中删除 EFS_INTEGRATION 选项。此更改会影响使用该选项组的所有实例。	从选项组中删除选项
单列排序	修改数据库实例并指定不包含 EFS_INTEGRATION 选项的另一个选项组。您可以指定默认（空）选项组，或指定其他自定义选项组。	修改 Amazon RDS 数据库实例

删除 EFS_INTEGRATION 选项后，您可以选择删除已连接到数据库实例的 EFS 文件系统。

Amazon EFS 集成问题排查

您的 RDS for Oracle 数据库实例监控与 Amazon EFS 文件系统的连接。当监控检测到问题时，它可能会尝试更正问题并在 RDS 控制台中发布事件。有关更多信息，请参阅查看 Amazon RDS 事件。

使用本部分中的信息帮助您诊断和修复在使用 Amazon EFS 集成时的常见问题。

Notification	描述	操作
`The EFS for RDS Oracle instance` *`instance_name`* `isn't available on the primary host. NFS port 2049 of your EFS isn't reachable.`	数据库实例无法与 EFS 文件系统通信。	请确保满足以下条件： • EFS 文件系统存在。 • 连接到 EFS 挂载目标的安全组具有入站规则，以允许在 TCP/2049（类型 NFS）上使用 RDS for Oracle 数据库实例的安全组或网络子网。
`The EFS isn't reachable.`	安装 EFS_INTEGRATION 选项时出错。	请确保满足以下条件： • EFS 文件系统存在。 • 连接到 EFS 挂载目标的安全组具有入站规则，以允许在 TCP/2049（类型 NFS）上使用 RDS for Oracle 数据库实例的安全组或网络子网。

Notification	描述	操作
		• 对您的 VPC 开启 `enableDnsSupport` 属性。 • 您正在您的 VPC 中使用 Amazon 提供的 DNS 服务器。Amazon EFS 集成不适用于自定义 DHCP DNS。
`The associated role with your DB instance wasn't found.`	安装 `EFS_INTEGRATION` 选项时出错。	确保您已将 IAM 角色与 RDS for Oracle 数据库实例关联。
`The associated role with your DB instance wasn't found.`	安装 `EFS_INTEGRATION` 选项时出错。RDS for Oracle 已从数据库快照中还原，`USE_IAM_ROLE` 选项设置为 `TRUE`。	确保您已将 IAM 角色与 RDS for Oracle 数据库实例关联。
`The associated role with your DB instance wasn't found.`	安装 `EFS_INTEGRATION` 选项时出错。RDS for Oracle 是根据一个多合一的 CloudFormation 模板创建的，其 `USE_IAM_ROLE` 选项设置为 `TRUE`。	作为解决方法，请完成以下步骤： 1. 使用 IAM 角色和默认选项组创建数据库实例。 2. 在后续的堆栈更新中，请添加带有 `EFS_INTEGRATION` 选项的自定义选项组。
`PLS-00302: component 'CREATE_DIRECTORY_EFS' must be declared`	当您使用不支持 Amazon EFS 的 RDS for Oracle 版本时，可能会发生此错误。	确保您使用的是 RDS for Oracle 数据库实例版本 19.0.0.0.ru-2022-07.rur-2022-07.r1 或更高版本。

Notification	描述	操作
`Read access of your EFS is denied. Check your file system policy.`	您的数据库实例无法读取 EFS 文件系统。	确保您的 EFS 文件系统允许通过 IAM 角色或在 EFS 文件系统级别进行读取访问。
不适用	您的数据库实例无法写入 EFS 文件系统。	执行以下步骤： 1. 确保您的 EFS 文件系统挂载在 Amazon EC2 实例上。 2. 向您的 RDS 用户授予 `others` 组写入权限。最简单的方法是在 EFS 文件系统的顶部目录上运行 chmod 777 命令。

Oracle Java 虚拟机

Amazon RDS 通过使用 JVM 选项来支持 Oracle Java 虚拟机 (JVM)。Oracle Java 提供了 SQL 架构和函数以在 Oracle 数据库中利用 Oracle Java 功能。有关更多信息，请参阅 Oracle 文档中的 Oracle Database 中的 Java 简介。您可以将 Oracle JVM 与所有版本的 Oracle Database 21c（21.0.0）和 Oracle Database 19c（19.0.0）结合使用。

Oracle JVM 的注意事项

Amazon RDS 中的 Java 实施拥有一组有限的权限。主用户被授予 RDS_JAVA_ADMIN 角色，进而授予 JAVA_ADMIN 角色所获权限的一个子集。若要列出授予 RDS_JAVA_ADMIN 角色的权限，请在您的数据库实例上运行以下查询：

```
SELECT * FROM dba_java_policy
   WHERE grantee IN ('RDS_JAVA_ADMIN', 'PUBLIC')
   AND enabled = 'ENABLED'
   ORDER BY type_name, name, grantee;
```

Oracle JVM 的先决条件

以下是使用 Oracle Java 的先决条件：

- 您的数据库实例必须属于足够大的类。db.t3.small 数据库实例类不支持 Oracle Java。有关更多信息，请参阅 数据库实例类。
- 数据库实例必须已启用自动次要版本升级。此选项使您的数据库实例在次要数据库引擎版本升级可用时自动接收该升级。Amazon RDS 使用该选项将数据库实例更新为最新的 Oracle 补丁集更新（PSU）或版本更新（RU）。有关更多信息，请参阅"修改 Amazon RDS 数据库实例"。

Oracle JVM 的最佳实践

以下是使用 Oracle Java 的最佳实践：

- 为了实现最高安全性，可将 JVM 选项与安全套接字层 (SSL) 结合使用。有关更多信息，请参阅"Oracle 安全套接字层"。
- 配置您的数据库实例以限制网络访问。有关更多信息，请参阅"在 VPC 中访问数据库实例的场景"和"在 VPC 中使用数据库实例"。
- 如果您满足以下条件，请更新 HTTPS 终端节点的配置以支持 TLSv1.2：

- 您可以使用 Oracle Java 虚拟机 (JVM) 通过 TLSv1 或 TLSv1.1 协议连接 HTTPS 终端节点。
- 您的终端节点不支持 TLSv1.2 协议。
- 您尚未将 2021 年 4 月版本更新应用于 Oracle 数据库。

通过更新终端节点配置，您可以确保 JVM 与 HTTPS 终端节点的连接将继续有效。有关 Oracle JRE 和 JDK 中 TLS 更改的更多信息，请参阅 Oracle JRE 和 JDK 加密路线图。

添加 Oracle JVM 选项

下面是将 JVM 选项添加到数据库实例的一般过程：

1. 创建新的选项组，或者复制或修改现有选项组。
2. 将 选项添加到该选项组。
3. 将选项组与数据库实例相关联。

添加 JVM 选项后，将出现短暂中断。在添加 选项后，您无需重新启动数据库实例。在选项组处于活动状态后，Oracle Java 将立即可用。

> **Note**
>
> 在该中断期间，将短暂禁用密码验证功能。您还可能会在中断期间看到与密码验证功能相关的事件。在 Oracle 数据库实例变得可用之前，将再次启用密码验证功能。

将 JVM 选项添加到数据库实例

1. 确定您要使用的选项组。您可以创建新的选项组，或使用现有选项组。如果您想使用现有选项组，请跳到下一步。或者，通过以下设置创建自定义数据库选项组：

   - 对于引擎，选择数据库实例使用的数据库引擎（oracle-ee、oracle-se、oracle-se1 或 oracle-se2）。
   - 对于主引擎版本，选择数据库实例的版本。

   有关更多信息，请参阅“创建选项组”。
2. 将 JVM 选项添加到该选项组。有关添加选项的更多信息，请参阅 将选项添加到选项组。
3. 将选项组应用到新的或现有的数据库实例：

- 对于新数据库实例，请在启动实例时应用选项组。有关更多信息，请参阅“创建 Amazon RDS 数据库实例”。
- 对于现有数据库实例，请修改实例并附加新的选项组以应用选项组。有关更多信息，请参阅“修改 Amazon RDS 数据库实例”。

4. 为用户授予所需的权限。

   默认情况下，Amazon RDS 主用户具有使用 JVM 选项的权限。如果其他用户需要具有这些权限，请在 SQL 客户端中以主用户身份连接到数据库实例，并为用户授予这些权限。

   以下示例为 JVM 用户授予使用 `test_proc` 选项的权限。

```
create user test_proc identified by password;
CALL dbms_java.grant_permission('TEST_PROC',
 'oracle.aurora.security.JServerPermission', 'LoadClassInPackage.*', '');
```

> **Note**
>
> 作为安全最佳实践，请指定除此处所示提示以外的密码。

   在为用户授予这些权限后，以下查询将返回输出。

```
select * from dba_java_policy where grantee='TEST_PROC';
```

> **Note**
>
> Oracle 用户名区分大小写，通常全部为大写字符。

删除 Oracle JVM 选项

您可以从数据库实例中删除 JVM 选项。删除该选项后，将出现短暂中断。删除 JVM 选项后，您无需重新启动数据库实例。

> **Warning**
>
> 如果数据库实例使用的是在选项中启用的数据类型，则删除 JVM 选项会导致数据丢失。请先备份数据，然后再继续。有关更多信息，请参阅“备份、还原和导出数据”。

要从数据库实例中删除 JVM 选项，请执行下列操作之一：

- 从 JVM 选项所属的选项组中删除该选项。此更改会影响使用该选项组的所有数据库实例。有关更多信息，请参阅“从选项组中删除选项”。
- 修改数据库实例并指定不包含 JVM 选项的另一个选项组。此更改会影响单个数据库实例。您可以指定默认 (空) 选项组，或指定其他自定义选项组。有关更多信息，请参阅“修改 Amazon RDS 数据库实例”。

Oracle Enterprise Manager

Amazon RDS 支持 Oracle Enterprise Manager (OEM)。OEM 是 Oracle 的企业信息技术集成管理产品线。

Amazon RDS 在 Oracle Database 19c 非 CDB 或 CDB 上支持 OEM。下表描述了支持的 OEM 选项。

选项	选项 ID	支持的 OEM 版本
OEM Database Express	`OEM`	OEM Database Express 19c
OEM Management Agent	`OEM_AGENT`	OEM Cloud Control for 13c

> **Note**
> 您可以使用 OEM Database 或 OEM Management Agent，但不能同时使用二者。

Oracle Enterprise Manager Database Express

Amazon RDS 可通过使用 OEM 选项支持 Oracle Enterprise Manager Database Express（EM Express）。Amazon RDS 支持使用 CDB 或非 CDB 架构的 EM Express for Oracle Database 19c。

EM Express 是数据库中包含的基于 Web 的数据库管理工具，仅在打开时才可用。该工具支持关键性能管理和基本数据库管理功能。有关更多信息，请参阅 Oracle 数据库文档中的 Introduction to Oracle Enterprise Manager Database Express。

> **Note**
>
> db.t3.small 数据库实例类不支持 EM Express。有关数据库实例类的更多信息，请参阅 RDS for Oracle 数据库实例类。

OEM 选项设置

Amazon RDS 支持 OEM 选项的以下设置。

选项设置	有效值	描述
端口	一个整数值	RDS for Oracle 数据库实例上负责侦听 EM Express 的端口。默认端口为 5500。
安全组	—	对 Port 具有访问权限的安全组。

步骤 1：添加 OEM 选项

将 OEM 选项添加到数据库实例的一般过程如下所示：

1. 创建新的选项组，或者复制或修改现有选项组。
2. 将 选项添加到该选项组。
3. 将选项组与数据库实例相关联。

如果添加 OEM 选项，则在数据库实例自动重新启动时会出现短暂中断。

将 OEM 选项添加到数据库实例

1. 确定您想要使用的选项组。您可以创建新的选项组，或使用现有选项组。如果您想使用现有选项组，请跳到下一步。或者，通过以下设置创建自定义数据库选项组：

   a. 对于引擎，请选择适用于您的数据库实例的 Oracle 版本。

   b. 对于主引擎版本，请选择数据库实例的版本。

   有关更多信息，请参阅“创建选项组”。

2. 将 OEM 选项添加到选项组中，然后配置选项设置。有关添加选项的更多信息，请参阅 将选项添加到选项组。有关各项设置的更多信息，请参阅OEM 选项设置。

   > **Note**
   >
   > 如果在已附加到一个或多个数据库实例的现有选项组中添加 OEM 选项，所有数据库实例自动重新启动时都会发生短暂中断。

3. 将选项组应用到新的或现有的数据库实例：

   - 对于新数据库实例，请在启动实例时应用选项组。有关更多信息，请参阅“创建 Amazon RDS 数据库实例”。
   - 对于现有数据库实例，请修改实例并附加新的选项组以应用选项组。如果添加 OEM 选项，则在数据库实例自动重新启动时会出现短暂中断。有关更多信息，请参阅 修改 Amazon RDS 数据库实例。

> **Note**
>
> 您还可以使用 Amazon CLI 添加 OEM 选项。有关示例，请参阅 将选项添加到选项组。

步骤 2：（仅限 CDB）解锁 DBSNMP 用户账户

如果数据库实例使用 CDB 架构，您必须以 `DBSNMP` 身份登录 EM Express。在 CDB 中，`DBSNMP` 是普通用户。默认情况下，此账户处于锁定状态。如果数据库实例不使用 CDB 架构，请跳过此步骤。

解锁 CDB 实例中的 DBSNMP 用户账户

1. 在 SQL*Plus 或其他 Oracle SQL 应用程序中，以主用户身份登录到数据库实例。

2. 运行以下存储过程来解锁 DBSNMP 账户：

```
EXEC rdsadmin.rdsadmin_util.reset_oem_agent_password('new_password');
```

   如果您收到错误消息，指出该过程不存在，请重启 CDB 实例来自动安装它。有关更多信息，请参阅 重启中的数据库实例。

步骤 3：通过浏览器访问 EM Express

当您从 Web 浏览器访问 EM Express 时，将显示一个登录窗口，提示您输入用户名和密码。

通过浏览器访问 EM Express

1. 确定 Amazon RDS 数据库实例的端点和 EM Express 端口。有关查找 Amazon RDS 数据库实例的终端节点的信息，请参阅 查找 RDS for Oracle 数据库实例的端点。
2. 使用以下格式在浏览器定位器栏中输入 URL。

```
https://endpoint.rds.amazonaws.com:port/em
```

   例如，如果 Amazon RDS 数据库实例的端点为 `mydb.a1bcde234fgh.us-east-1.rds.amazonaws.com`，而 EM Express 端口为 `1158`，则使用以下 URL 访问 EM Express。

```
https://mydb.f9rbfa893tft.us-east-1.rds.amazonaws.com:1158/em
```

3. 当系统提示您输入登录详细信息时，根据数据库架构，执行以下操作之一：

   数据库为非 CDB。

      键入数据库实例的主用户名和主密码。

   数据库为 CDB。

      对于用户输入 `DBSNMP`，并输入 `DBSNMP` 密码。将 `Container` 字段留空。

修改 OEM Database 设置

启用 OEM Database 后，可以修改该选项的安全组设置。

将选项组与数据库实例相关联后，将无法修改 OEM 端口号。要更改数据库实例的 OEM 端口号，可执行以下操作：

1. 创建新的选项组。
2. 将具有新端口号的 OEM 选项添加到新选项组。
3. 删除数据库实例的现有选项组。
4. 将新选项组添加到数据库实例。

有关如何修改选项设置的更多信息，请参阅 修改选项设置。有关各项设置的更多信息，请参阅OEM 选项设置。

运行 OEM Database Express 任务

您可以使用 Amazon RDS 过程运行某些 OEM Database Express 任务。通过运行这些过程，您可以执行下面列出的任务。

> **Note**
> OEM Database Express 任务异步运行。

任务

- 将 OEM Database Express 的网站前端切换到 Adobe Flash
- 将 OEM Database Express 的网站前端切换到 Oracle JET

将 OEM Database Express 的网站前端切换到 Adobe Flash

> **Note**
> 此任务仅适用于 Oracle Database 19c 非 CDB。

从 Oracle Database 19c 开始，Oracle 已弃用以前的 OEM Database Express 用户界面，该界面是基于 Adobe Flash 的。相反，OEM Database Express 现在使用通过 Oracle JET 构建的界面。如果您在使用新界面时遇到困难，可以切换回已弃用的基于 Flash 的界面。可能遇到的困难包括登录 OEM Database Express 后卡在 `Loading` 屏幕上。您可能还会失去基于 Flash 的 OEM Database Express 版本中存在的某些功能。

要将 OEM Database Express 网站前端切换到 Adobe Flash，请运行 Amazon RDS 过程 `rdsadmin.rdsadmin_oem_tasks.em_express_frontend_to_flash`。该过程相当于 `execemx emx` SQL 命令。

安全最佳实践不鼓励使用 Adobe Flash。虽然您可以恢复到基于 Flash 的 OEM Database Express，但我们建议您尽可能使用基于 JET 的 OEM Database Express 网站。如果您恢复使用 Adobe Flash 并希望切换回使用 Oracle JET，请使用 `rdsadmin.rdsadmin_oem_tasks.em_express_frontend_to_jet` 过程。Oracle 数据库升级后，较新版本的 Oracle JET 可能会解决 OEM Database Express 中与 JET 相关的问题。有关切换到 Oracle JET 的更多信息，请参阅将 OEM Database Express 的网站前端切换到 Oracle JET。

> **Note**
>
> 从只读副本的源数据库实例运行此任务也会导致只读副本将其 OEM Database Express 网站前端切换到 Adobe Flash。

以下过程调用将创建一个任务，以将 OEM Database Express 网站切换到 Adobe Flash，并返回任务的 ID。

```
SELECT rdsadmin.rdsadmin_oem_tasks.em_express_frontend_to_flash() as TASK_ID from DUAL;
```

您可以通过显示任务的输出文件来查看结果。

```
SELECT text FROM table(rdsadmin.rds_file_util.read_text_file('BDUMP','dbtask-task-id.log'));
```

将 *`task-id`* 替换为由此过程返回的任务 ID。有关 Amazon RDS 过程 `rdsadmin.rds_file_util.read_text_file` 的更多信息，请参阅读取数据库实例目录中的文件。

您还可以通过在 Logs & events (日志和事件) 部分中搜索 Amazon Web Services Management Console 的相关日志条目，查看 `task-id` 中的任务输出文件的内容。

将 OEM Database Express 的网站前端切换到 Oracle JET

> **Note**
>
> 此任务仅适用于 Oracle Database 19c 非 CDB。

要将 OEM Database Express 网站前端切换到 Oracle JET，请运行 Amazon RDS 过程 `rdsadmin.rdsadmin_oem_tasks.em_express_frontend_to_jet`。该过程相当于 `execemx omx` SQL 命令。

默认情况下，运行 19c 或更高版本的 Oracle 数据库实例的 OEM Database Express 网站使用 Oracle JET。如果您使用了 `rdsadmin.rdsadmin_oem_tasks.em_express_frontend_to_flash` 过程将 OEM Database Express 网站前端切换到 Adobe Flash，则可以切换回 Oracle JET。要执行此操作，请使用 `rdsadmin.rdsadmin_oem_tasks.em_express_frontend_to_jet` 过程。有关切换到 Adobe Flash 的更多信息，请参阅将 OEM Database Express 的网站前端切换到 Adobe Flash。

> **Note**
> 从只读副本的源数据库实例运行此任务也会导致只读副本将其 OEM Database Express 网站前端切换到 Oracle JET。

以下过程调用将创建一个任务以将 OEM Database Express 网站切换到 Oracle JET，并返回任务的 ID。

```
SELECT rdsadmin.rdsadmin_oem_tasks.em_express_frontend_to_jet() as TASK_ID from DUAL;
```

您可以通过显示任务的输出文件来查看结果。

```
SELECT text FROM table(rdsadmin.rds_file_util.read_text_file('BDUMP','dbtask-task-id.log'));
```

将 *`task-id`* 替换为由此过程返回的任务 ID。有关 Amazon RDS 过程 `rdsadmin.rds_file_util.read_text_file` 的更多信息，请参阅读取数据库实例目录中的文件。

您还可以通过在 Logs & events (日志和事件) 部分中搜索 Amazon Web Services Management Console 的相关日志条目，查看 `task-id` 中的任务输出文件的内容。

删除 OEM Database 选项

可以从数据库实例中删除 OEM 选项。如果删除 OEM 选项，实例在自动重新启动时会出现短暂中断。因此，在删除 OEM 选项后，不需要重新启动数据库实例。

要从数据库实例中删除 OEM 选项，请执行下列操作之一：

- 从“OEM ”选项所属的选项组中删除该选项。此更改会影响使用该选项组的所有数据库实例。有关更多信息，请参阅“从选项组中删除选项”。
- 修改数据库实例，并指定不包含“OEM ”选项的另一个选项组。此更改会影响单个数据库实例。您可以指定默认 (空) 选项组，或指定其他自定义选项组。有关更多信息，请参阅“修改 Amazon RDS 数据库实例”。

适用于 Enterprise Manager Cloud Control 的 Oracle Management Agent

Oracle Enterprise Manager (OEM) Management Agent 是一种软件组件，可监控在主机上运行的目标并将该信息传达到中间层 Oracle Management Service (OMS)。Amazon RDS 通过使用 `OEM_AGENT` 选项支持 Management Agent。

有关更多信息，请参阅 Oracle 文档中的 Oracle Enterprise Manager Cloud Control 12c 概述和 Oracle Enterprise Manager Cloud Control 13c 概述。

主题

Management Agent 的要求

以下是使用 Management Agent 的一般要求：

- 数据库实例必须运行 Oracle Database 19c（19.0.0.0）。可以使用 CDB 或非 CDB 架构。
- 您必须使用配置为连接到您的数据库实例的 Oracle Management Service（OMS）。请注意以下 OMS 要求：
  - Management Agent 版本 13.5.0.0.v2 和 13.5.0.0.v3 需要 OMS 版本 13.5.0.23。
  - Management Agent 版本 13.5.0.0.v1 需要 OMS 版本 13.5.0.0。
  - Management Agent 版本 13.4.0.9.v1 和 13.4.0.9.v2 需要 OMS 版本 13.4.0.9 或更高版本及补丁 32198287。
- 在大多数情况下，您必须配置 VPC 以允许 OMS 与数据库实例之间的连接。如果您不熟悉 Amazon Virtual Private Cloud (Amazon VPC)，建议您在继续之前先完成 教程：创建 VPC 以用于数据库实例（仅限 IPv4） 中的步骤。

- 您可以将 Management Agent 与适用于 12c 和 13c 的 Oracle Enterprise Manager Cloud Control 结合使用。确保您的 OEM 版本有足够的存储空间：
  - OEM 13c 版本 5 至少 8.5GiB
  - OEM 13c 版本 4 至少 8.5 GiB
  - OEM 13c 版本 3 至少 8.5 GiB
  - OEM 13c 版本 2 至少 5.5 GiB
  - OEM 13c 版本 1 至少 4.5 GiB
  - OEM 12c 至少 2.5 GiB
- 如果您使用的是 Management Agent 版本 `OEM_AGENT 13.2.0.0.v3` 和 `13.3.0.0.v2`，并且想要使用 TCPS 连接，请按照 Oracle 文档中的配置第三方 CA 证书以便与目标数据库通信中的说明进行操作。另外，按照 Oracle 文档 ID 为 2241358.1 的 Oracle 文档中的说明更新 OMS 上的 JDK。此步骤可确保 OMS 支持数据库支持的所有密码套件。

> **Note**
>
> Management Agent `OEM_AGENT 13.2.0.0.v3`、`13.3.0.0.v2`、`13.4.0.9.v1` 和更高版本支持 Management Agent 和数据库实例之间的 TCPS 连接。

OMS 主机通信先决条件

确保您的 OMS 主机和您的 Amazon RDS 数据库实例可以通信。执行以下操作：

- 当 OMS 主机位于防火墙后时，要从 Management Agent 连接到 OMS 主机，请将数据库实例的 IP 地址添加到防火墙中。确保 OMS 防火墙允许以下网络流量：

  从 OMS 主机到数据库实例

  配置单向防火墙规则，以便支持从 OMS 主机到数据库侦听器端口（默认为 1521）和 OEM Agent 端口（默认为 3872）的流量。

  从数据库实例到 OMS 主机

  配置单向防火墙规则，以便支持从数据库实例到 OMS HTTP 端口（默认为 4903）的流量。
- 要从 OMS 连接到 Management Agent（如果 OMS 具有一个可公开解析的主机名），需将该 OMS 地址添加到安全组。您的安全组必须拥有允许访问数据库侦听器端口和 Management Agent 端口的入站规则。有关创建安全组并添加入站规则的示例，请参阅教程：创建 VPC 以用于数据库实例（仅限 IPv4）。

- 要从 OMS 连接到 Management Agent (如果 OMS 没有可公开解析的主机名)，请使用下列项目之一：
  - 如果 OMS 已在私有 VPC 中的 Amazon Elastic Compute Cloud (Amazon EC2) 实例上托管，则可设置 VPC 对等连接以从 OMS 连接到 Management Agent。有关更多信息，请参阅“VPC 中的数据库实例由另一 VPC 中的 EC2 实例访问”。
  - 如果您的 OMS 已在本地托管，则可设置 VPN 连接以允许从 OMS 访问 Management Agent。有关更多信息，请参阅 VPC 中的数据库实例由客户端应用程序通过互联网访问或 VPN 连接。

Management Agent 的限制

以下是使用 Management Agent 的一些限制：

- 您无法提供自定义 Oracle 管理代理映像。
- 不支持需要主机凭证的管理任务（例如作业执行和数据库修补）。
- 不保证主机指标和进程列表反映实际的系统状态。因此，您不应使用 OEM 来监视根文件系统或挂载点文件系统。有关监控操作系统的更多信息，请参阅 使用增强监控来监控操作系统指标。
- 不支持自动发现。您必须手动添加数据库目标。
- OMS 模块可用性取决于数据库版本。例如，数据库性能诊断和优化模块仅适用于 Oracle Database 企业版。
- Management Agent 会占用额外的内存和计算资源。如果您在启用 `OEM_AGENT` 选项后遇到了性能问题，建议您向上扩展至更大的数据库实例类。有关更多信息，请参阅“数据库实例类”和“修改 Amazon RDS 数据库实例”。
- 在 Amazon RDS 主机上运行 `OEM_AGENT` 的用户没有对警报日志的操作系统访问权限。因此，您无法收集 OEM 中 `DB Alert Log` 和 `DB Alert Log Error Status` 的指标。

Management Agent 的选项设置

Amazon RDS 支持 Management Agent 选项的以下设置。

选项设置	必需	有效值	描述

版本 (`AGENT_VERSION`)	是	`13.5.0.0.v3`
		
		`13.5.0.0.v2`
			Management Agent 软件的版本。支持的最低版本为 `13.1.0.0.v1`。
			
			Amazon CLI 选项名称为 `OptionVersion`。

选项设置	必需	有效值	描述

		`13.5.0.0.v1`
		
		`13.4.0.9.v2`
		
		`13.4.0.9.v1`
		
		`13.3.0.0.v2`
		
		`13.3.0.0.v1`
		
		`13.2.0.0.v3`
		
		`13.2.0.0.v2`
		
		`13.2.0.0.v1`
		
		`13.1.0.0.v1`
			Note
			在 Amazon GovCloud (US)区域中，13.1 版本不可用。

端口 (`AGENT_PORT`)	是	一个整数值	侦听 OMS 主机的数据库实例上的端口。默认值为 3872。您的 OMS 主机必须属于对此端口具有访问权限的安全组。 Amazon CLI 选项名称为 `Port`。

选项设置	必需	有效值	描述
安全组	是	现有安全组	对 Port 具有访问权限的安全组。您的 OMS 主机必须属于此安全组。 Amazon CLI 选项名称为 `VpcSecurityGroupMemberships` 或 `DBSecurityGroupMemberships` 。
OMS_HOST	是	一个字符串值，例如 *my.example.oms*	OMS 的可公开访问的主机名或 IP 地址。 Amazon CLI 选项名称为 `OMS_HOST`。
OMS_PORT	是	一个整数值	OMS 主机上侦听 Management Agent 的 HTTPS 上传端口。 要确定 HTTPS 上传端口，请连接到 OMS 主机，然后运行以下命令 (此操作需要 SYSMAN 密码)： `emctl status oms -details` Amazon CLI 选项名称为 `OMS_PORT`。
AGENT_REGISTRATION_PASSWORD	是	一个字符串值	Management Agent 用于通过 OMS 自行进行身份验证的密码。建议您在启用 `OEM_AGENT` 选项之前先在 OMS 中创建一个永久性密码。使用永久性密码，您可以在多个 Amazon RDS 数据库之中共享一个 Management Agent 选项组。 Amazon CLI 选项名称为 `AGENT_REGISTRATION_PASSWORD` 。
ALLOW_TLS_ONLY	否	`true`、`false`认值）	一个值，该值将 OEM 代理配置为仅在代理作为服务器侦听时支持 TLSv1 协议。不再支持此设置。默认情况下，Management Agent 版本 13.1.0.0.v1 及更高版本支持传输层安全性协议（TLS）。

选项设置	必需	有效值	描述
MINIMUM_TLS_VERSION	否	TLSv1 (默认值)TLSv1.2	一个值，该值指定 OEM 代理在代理作为服务器侦听时支持的最低 TLS 版本。失去支持的代理版本仅支持 TLSv1 设置。
TLS_CIPHER_SUITE	否	请参阅Management Agent 的选项设置。	一个值，该值指定 OEM 代理在代理作为服务器侦听时使用的 TLS 密码套件。

下表列出了 Management Agent 选项支持的 TLS 密码套件。

密码套件	支持的引擎版本	符合 FedRAMP 要求
TLS_RSA_WITH_AES_128_CBC_SHA	全部	否
TLS_RSA_WITH_AES_128_CBC_SHA256	13.1.0.0.v1 及更高版本	否
TLS_RSA_WITH_AES_256_CBC_SHA	13.2.0.0.v3 及更高版本	否
TLS_RSA_WITH_AES_256_CBC_SHA256	13.2.0.0.v3 及更高版本	否
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA	13.2.0.0.v3 及更高版本	是
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA	13.2.0.0.v3 及更高版本	是
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256	13.2.0.0.v3 及更高版本	是
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384	13.2.0.0.v3 及更高版本	是

步骤 1：将 Management Agent 选项添加到数据库实例

要将 Management Agent 选项添加到数据库实例，请执行以下操作：

1. 创建新的选项组，或者复制或修改现有选项组。
2. 将 选项添加到该选项组。
3. 将选项组与数据库实例相关联。

如果遇到错误，请查看我的 Oracle 支持文档以了解有关解决特定问题的信息。

在添加“Management Agent”选项后，您无需重新启动数据库实例。只要选项组处于活动状态，OEM Agent 就处于活动状态。

如果 OMS 主机使用的是不受信任的第三方证书，则 Amazon RDS 将返回以下错误。

```
You successfully installed the OEM_AGENT option. Your OMS host is using an untrusted
 third party certificate.
Configure your OMS host with the trusted certificates from your third party.
```

如果返回此错误，则在更正问题之前，不会启用 Management Agent 选项。有关更正问题的信息，请参阅“我的 Oracle 支持”文档 2202569.1。

控制台

将 Management Agent 选项添加到数据库实例

1. 确定您想要使用的选项组。您可以创建新的选项组，或使用现有选项组。如果您想使用现有选项组，请跳到下一步。或者，通过以下设置创建自定义数据库选项组：

    a. 对于引擎，请选择适用于您的数据库实例的 Oracle 版本。
    b. 对于主引擎版本，请选择数据库实例的版本。

    有关更多信息，请参阅“创建选项组”。
2. 将 OEM_AGENT 选项添加到选项组，然后配置选项设置。有关添加选项的更多信息，请参阅 将选项添加到选项组。有关各项设置的更多信息，请参阅Management Agent 的选项设置。
3. 将选项组应用到新的或现有的数据库实例：

    • 对于新数据库实例，您可以在启动实例时应用选项组。有关更多信息，请参阅“创建 Amazon RDS 数据库实例”。

- 对于现有数据库实例，您可以通过修改实例并附加新的选项组来应用选项组。有关更多信息，请参阅“修改 Amazon RDS 数据库实例”。

Amazon CLI

以下示例使用 Amazon CLI add-option-to-option-group 命令将 OEM_AGENT 选项添加到名为 myoptiongroup 的选项组。

对于 Linux、macOS 或 Unix：

```
aws rds add-option-to-option-group \
    --option-group-name "myoptiongroup" \
    --options
 OptionName=OEM_AGENT,OptionVersion=13.1.0.0.v1,Port=3872,VpcSecurityGroupMemberships=sg-123456
{Name=OMS_PORT,Value=4903},{Name=AGENT_REGISTRATION_PASSWORD,Value=password}] \
    --apply-immediately
```

对于 Windows：

```
aws rds add-option-to-option-group ^
    --option-group-name "myoptiongroup" ^
    --options
 OptionName=OEM_AGENT,OptionVersion=13.1.0.0.v1,Port=3872,VpcSecurityGroupMemberships=sg-123456
{Name=OMS_PORT,Value=4903},{Name=AGENT_REGISTRATION_PASSWORD,Value=password}] ^
    --apply-immediately
```

步骤 2：解锁 DBSNMP 用户账户

Management Agent 使用 DBSNMP 用户账户连接到数据库，并向 Oracle Enterprise Manager 报告问题。在 CDB 中，DBSNMP 是普通用户。Management Agent 和 OEM Database Express 都需要此用户账户。默认情况下，此账户处于锁定状态。解锁此账户的过程会有所不同，具体取决于数据库使用的是非 CDB 还是 CDB 架构。

解锁 DBSNMP 用户账户

1. 在 SQL*Plus 或其他 Oracle SQL 应用程序中，以主用户身份登录到数据库实例。
2. 根据数据库架构，执行以下操作之一：

   数据库为非 CDB。

      运行以下 SQL 语句：

```
ALTER USER dbsnmp IDENTIFIED BY new_password ACCOUNT UNLOCK;
```

数据库为 CDB。

运行以下存储过程来解锁 DBSNMP 账户：

```
EXEC rdsadmin.rdsadmin_util.reset_oem_agent_password('new_password');
```

如果您收到错误消息，指出该过程不存在，请重启 CDB 实例来自动安装它。有关更多信息，请参阅 重启中的数据库实例。

步骤 3：将目标添加到 Management Agent 控制台

要将数据库实例添加为目标，请确保您知道端点和端口。有关查找 Amazon RDS 数据库实例的端点的信息，请参阅 查找 RDS for Oracle 数据库实例的端点。如果数据库使用 CDB 架构，则将 CDB$ROOT 容器单独添加为目标。

将目标添加到 Management Agent 控制台

1. 在 OMS 控制台中，选择 Setup、Add Target、Add Targets Manually。
2. 选择 Add Targets Declaratively by Specifying Target Monitoring Properties。
3. 对于 Target Type，选择 Database Instance。
4. 对于 Monitoring Agent (监控代理)，请选择标识符与 RDS 数据库实例标识符相同的代理。
5. 选择 Add Manually。
6. 输入 Amazon RDS 数据库实例的端点，或者从主机名列表中选择该端点。确保指定的主机名与 Amazon RDS 数据库实例的端点匹配。
7. 指定以下数据库属性：
   - 对于 Target name (目标名称)，输入一个名称。
   - 对于 Database system name (数据库系统名称)，输入一个名称。
   - 对于 Monitor username (监控用户名)，输入 **dbsnmp**。
   - 对于监控密码，输入步骤 2：解锁 DBSNMP 用户账户中的密码。
   - 对于 Role (角色)，输入 normal (正常)。
   - 对于 Oracle home path (Oracle 主路径)，输入 **/oracle**。
   - 对于 Listener Machine name，已显示该代理标识符。

- 对于 Port (端口)，输入数据库端口。RDS 默认端口为 1521。
- 对于 Database name (数据库名称)，输入数据库的名称。如果数据库为 CDB，则此名称为 RDSCDB。

8. 选择 Test Connection。
9. 选择下一步。目标数据库将显示在所监控资源的列表中。

管理 Management Agent

您可以使用 Amazon RDS 过程在 Management Agent 上运行某些 EMCTL 命令。通过运行这些过程，您可以执行下面列出的任务。

> **Note**
>
> 任务以异步方式执行。

任务

- 获取 Management Agent 的状态
- 重新启动 Management Agent
- 列出 Management Agent 监控的目标
- 列出 Management Agent 监控的收集线程
- 清除 Management Agent 的状态
- 让 Management Agent 上载其 OMS
- 为 OMS 执行 ping 操作
- 查看正在执行的任务的状态

获取 Management Agent 的状态

要获取 Management Agent 的状态，请运行 Amazon RDS 过程 `rdsadmin.rdsadmin_oem_agent_tasks.get_status_oem_agent`。该过程相当于 `emctl status agent` 命令。

以下过程创建一个任务来获取管理代理的状态，并返回任务的 ID。

```
SELECT rdsadmin.rdsadmin_oem_agent_tasks.get_status_oem_agent() as TASK_ID from DUAL;
```

要通过显示任务的输出文件来查看结果，请参阅查看正在执行的任务的状态。

重新启动 Management Agent

要重新启动 Management Agent，请运行 Amazon RDS 过程 `rdsadmin.rdsadmin_oem_agent_tasks.restart_oem_agent`。该过程相当于运行 `emctl stop agent` 和 `emctl start agent` 命令。

以下过程创建一个任务来重新启动管理代理并返回任务的 ID。

```
SELECT rdsadmin.rdsadmin_oem_agent_tasks.restart_oem_agent as TASK_ID from DUAL;
```

要通过显示任务的输出文件来查看结果，请参阅查看正在执行的任务的状态。

列出 Management Agent 监控的目标

要列出 Management Agent 监控的目标，请运行 Amazon RDS 过程 `rdsadmin.rdsadmin_oem_agent_tasks.list_targets_oem_agent`。该过程相当于运行 `emctl config agent listtargets` 命令。

以下过程创建一个任务来列出管理代理监控的目标并返回任务的 ID。

```
SELECT rdsadmin.rdsadmin_oem_agent_tasks.list_targets_oem_agent as TASK_ID from DUAL;
```

要通过显示任务的输出文件来查看结果，请参阅查看正在执行的任务的状态。

列出 Management Agent 监控的收集线程

要列出管理代理监控的所有正在运行、就绪和计划的收集线程，请运行 Amazon RDS 过程 `rdsadmin.rdsadmin_oem_agent_tasks.list_clxn_threads_oem_agent`。该过程相当于 `emctl status agent scheduler` 命令。

以下过程创建一个任务来列出收集线程并返回任务的 ID。

```
SELECT rdsadmin.rdsadmin_oem_agent_tasks.list_clxn_threads_oem_agent() as TASK_ID from
 DUAL;
```

要通过显示任务的输出文件来查看结果，请参阅查看正在执行的任务的状态。

清除 Management Agent 的状态

要清除 Management Agent 的状态，请运行 Amazon RDS 过程 `rdsadmin.rdsadmin_oem_agent_tasks.clearstate_oem_agent`。该过程相当于运行 `emctl clearstate agent` 命令。

以下过程创建一个任务来清除管理代理的状态并返回任务的 ID。

```
SELECT rdsadmin.rdsadmin_oem_agent_tasks.clearstate_oem_agent() as TASK_ID from DUAL;
```

要通过显示任务的输出文件来查看结果，请参阅查看正在执行的任务的状态。

让 Management Agent 上载其 OMS

要让 Management Agent 上载与其关联的 Oracle Management Server (OMS)，请运行 Amazon RDS 过程 `rdsadmin.rdsadmin_oem_agent_tasks.upload_oem_agent`。该过程相当于运行 `emclt upload agent` 命令。

以下过程创建一个任务，使 Management Agent 上传其关联的 OMS 并返回任务的 ID。

```
SELECT rdsadmin.rdsadmin_oem_agent_tasks.upload_oem_agent() as TASK_ID from DUAL;
```

要通过显示任务的输出文件来查看结果，请参阅查看正在执行的任务的状态。

为 OMS 执行 ping 操作

要为 Management Agent 的 OMS 执行 ping 操作，请运行 Amazon RDS 过程 `rdsadmin.rdsadmin_oem_agent_tasks.ping_oms_oem_agent`。该过程相当于运行 `emctl pingOMS` 命令。

以下过程创建一个任务来 Ping 管理代理的 OMS 并返回任务的 ID。

```
SELECT rdsadmin.rdsadmin_oem_agent_tasks.ping_oms_oem_agent() as TASK_ID from DUAL;
```

要通过显示任务的输出文件来查看结果，请参阅查看正在执行的任务的状态。

查看正在执行的任务的状态

您可以查看 bdump 文件中的持续任务的状态。bdump 文件位于 `/rdsdbdata/log/trace` 目录下。每个 bdump 文件名称格式如下。

```
dbtask-task-id.log
```

如果要监控任务，请将 *task-id* 替换为要监控的任务的 ID。

要查看 bdump 文件的内容，请运行 Amazon RDS 过程 `rdsadmin.rds_file_util.read_text_file`。以下查询返回 `dbtask-1546988886389-2444.log` bdump 文件的内容。

```
SELECT text FROM
 table(rdsadmin.rds_file_util.read_text_file('BDUMP','dbtask-1546988886389-2444.log'));
```

有关 Amazon RDS 过程 `rdsadmin.rds_file_util.read_text_file` 的更多信息，请参阅读取数据库实例目录中的文件。

删除 Management Agent 选项

可以从数据库实例中删除 OEM Agent。删除 OEM Agent 后，您无需重新启动数据库实例。

要从数据库实例中删除 OEM Agent，请执行下列操作之一：

- 从“OEM Agent”选项所属的选项组中删除该选项。此更改会影响使用该选项组的所有数据库实例。有关更多信息，请参阅“从选项组中删除选项”。
- 修改数据库实例，并指定不包含“OEM Agent”选项的另一个选项组。此更改会影响单个数据库实例。您可以指定默认 (空) 选项组，或指定其他自定义选项组。有关更多信息，请参阅“修改 Amazon RDS 数据库实例”。

Oracle Label Security

Amazon RDS 通过使用 OLS 选项支持 Oracle Database 企业版的 Oracle Label Security。

大多数数据库安全性控制对象级别的访问。Oracle Label Security 提供对针对各个表格行的访问的精细控制。例如，您可以使用 Label Security 通过基于策略的管理模型实施法规合规性。您可以使用 Label Security 策略控制对敏感数据的访问，并限制对具有相应清除级别的用户的访问。有关更多信息，请参阅 Oracle 文档中的 Oracle Label Security 简介。

主题

- Oracle Label Security 的要求
- 使用 Oracle Label Security 时的注意事项
- 添加“Oracle Label Security”选项
- 故障排除

Oracle Label Security 的要求

熟悉 Oracle Label Security 的以下要求：

- 数据库实例必须使用自带许可模式。有关更多信息，请参阅“RDS for Oracle 许可选项”。
- 您必须拥有 Oracle 企业版的有效许可证以及软件更新许可证和支持。
- 您的 Oracle 许可证必须包含 Label Security 选项。

使用 Oracle Label Security 时的注意事项

要使用 Oracle Label Security，请创建可控制对您表格中特定行的访问的策略。有关更多信息，请参阅 Oracle 文档中的创建 Oracle Label Security 策略。

请考虑以下事项：

- Oracle Label Security 是永久性的持续选项。因为该选项是永久性的，所以不能将其从选项组中删除。如果您将 Oracle Label Security 添加到选项组并将其与您的数据库实例关联，则可以稍后将其他选项组与您的数据库实例关联，但该组还必须包含 Oracle Label Security 选项。
- 使用 Label Security 时，您以 `LBAC_DBA` 角色执行所有操作。数据库实例的主用户将被授予 `LBAC_DBA` 角色。您可以向其他用户授予 `LBAC_DBA` 角色，以便他们可以管理 Label Security 策略。

- 请确保将对 OLS_ENFORCEMENT 程序包的访问权限授予需要访问 Oracle Label Security 的任何新用户。要授予对 OLS_ENFORCEMENT 包的访问权限，请以主用户身份连接到数据库实例，然后运行以下 SQL 语句：

```
GRANT ALL ON LBACSYS.OLS_ENFORCEMENT TO username;
```

- 可以通过 Oracle Enterprise Manager（OEM）Cloud Control 配置 Label Security。Amazon RDS 通过 Management Agent 选项支持 OEM Cloud Control。有关更多信息，请参阅 适用于 Enterprise Manager Cloud Control 的 Oracle Management Agent。

添加“Oracle Label Security”选项

将“Oracle Label Security”选项添加到数据库实例的一般过程如下所示：

1. 创建新的选项组，或者复制或修改现有选项组。
2. 将 选项添加到该选项组。

> **Important**
>
> Oracle Label Security 是永久性的持续选项。

3. 将选项组与数据库实例相关联。

在添加“Label Security”选项后，一旦激活该选项组，就会激活 Label Security。

将“Label Security”选项添加到数据库实例

1. 确定您想要使用的选项组。您可以创建新的选项组，或使用现有选项组。如果您想使用现有选项组，请跳到下一步。或者，通过以下设置创建自定义数据库选项组：

    a. 对于 引擎，请选择 oracle-ee。

    b. 对于主引擎版本，选择数据库实例的版本。

    有关更多信息，请参阅“创建选项组”。

2. 将 OLS 选项添加到该选项组。有关添加选项的更多信息，请参阅 将选项添加到选项组。

> **Important**
>
> 如果您向已附加到一个或多个数据库实例的现有选项组添加“Label Security”，则将重新启动所有数据库实例。

3. 将选项组应用到新的或现有的数据库实例：

   - 对于新数据库实例，您可以在启动实例时应用选项组。有关更多信息，请参阅“创建 Amazon RDS 数据库实例”。
   - 对于现有数据库实例，您可以通过修改实例并附加新的选项组来应用选项组。如果您将“Label Security”选项添加到现有数据库实例，则在数据库实例自动重新启动时会出现短暂中断。有关更多信息，请参阅 修改 Amazon RDS 数据库实例。

故障排除

以下是您在使用 Oracle Label Security 时可能遇到的问题。

问题	故障排除建议
当尝试创建策略时，您会看到一条与以下内容类似的错误消息：`insufficient authorization for the SYSDBA package`。	有关 Oracle 的 Label Security 功能的已知问题阻止了用户名为 16 或 24 个字符的用户运行 Label Security 命令。您可以使用其他数量的字符创建新用户，向该新用户授予 LBAC_DBA 角色，以该新用户的身份登录，并以该新用户的身份运行 OLS 命令。有关更多信息，请联系 Oracle Support。

Oracle Locator

Amazon RDS 通过使用 `LOCATOR` 选项支持 Oracle Locator。Oracle Locator 提供支持基于 Internet 和无线服务的应用程序与基于合作伙伴的 GIS 解决方案通常所需的功能。Oracle Locator 是 Oracle Spatial 的有限子集。有关更多信息，请参阅 Oracle 文档中的 Oracle Locator。

> **Important**
>
> 如果您使用 Oracle Locator，则当存在通用漏洞评分系统 (CVSS) 评分为 9+ 的安全漏洞或其他公布的安全漏洞时，Amazon RDS 会自动将数据库实例更新为最新的 Oracle PSU。

Oracle Locator 支持的数据库版本

RDS for Oracle 支持 Oracle Locator for Oracle Database 19c。Oracle Database 21c 不支持 Oracle Locator，但 Oracle Spatial 选项中提供了其功能。以前，Spatial 选项需要额外的许可证。Oracle Locator 是 Oracle Spatial 功能的子集，不需要额外的许可证。2019 年，Oracle 宣布，所有 Oracle Spatial 功能都包含在企业版和标准版 2 许可证中，无需额外费用。因此，Oracle Spatial 选项不再需要额外的许可。有关更多信息，请参阅 Oracle Database Insider 博客中的Machine Learning、Spatial 和 Graph - 无需许可证！。

Oracle Locator 的先决条件

以下是使用 Oracle Locator 的先决条件：

- 数据库实例必须属于足够的类。db.t3.small 数据库实例类不支持 Oracle Locator。有关更多信息，请参阅 RDS for Oracle 数据库实例类。
- 数据库实例必须已启用自动次要版本升级。该选项允许数据库实例在次要数据库引擎版本升级可用时自动接收这些升级，这对于安装 Oracle Java 虚拟机（JVM）的任意选项是必需的。Amazon RDS 使用该选项将数据库实例更新为最新的 Oracle 补丁集更新（PSU）或版本更新（RU）。有关更多信息，请参阅"修改 Amazon RDS 数据库实例"。

Oracle Locator 的最佳实践

以下是使用 Oracle Locator 的最佳实践：

- 为了实现最高安全性，可将 `LOCATOR` 选项与安全套接字层 (SSL) 结合使用。有关更多信息，请参阅"Oracle 安全套接字层"。

- 配置数据库实例以限制对它的访问。有关更多信息，请参阅“在 VPC 中访问数据库实例的场景”和“在 VPC 中使用数据库实例”。

添加 Oracle Locator 选项

下面是将 `LOCATOR` 选项添加到数据库实例的一般过程：

1. 创建新的选项组，或者复制或修改现有选项组。
2. 将 选项添加到该选项组。
3. 将选项组与数据库实例相关联。

如果数据库实例上未 安装 Oracle Java 虚拟机 (JVM)，则添加 `LOCATOR` 选项时会出现短暂中断。如果数据库实例上已安装 Oracle Java 虚拟机 (JVM)，则不会出现中断。在添加 选项后，您无需重新启动数据库实例。只要选项组处于活动状态，Oracle Locator 就可用。

> **Note**
> 在该中断期间，将短暂禁用密码验证功能。您还可能会在中断期间看到与密码验证功能相关的事件。在 Oracle 数据库实例变得可用之前，将再次启用密码验证功能。

将 **`LOCATOR`** 选项添加到数据库实例

1. 确定您要使用的选项组。您可以创建新的选项组，或使用现有选项组。如果您想使用现有选项组，请跳到下一步。或者，通过以下设置创建自定义数据库选项组：

   a. 对于引擎，请选择适用于数据库实例的 Oracle 版本。

   b. 对于主引擎版本，选择数据库实例的版本。

   有关更多信息，请参阅“创建选项组”。
2. 将 LOCATOR 选项添加到该选项组。有关添加选项的更多信息，请参阅 将选项添加到选项组。
3. 将选项组应用到新的或现有的数据库实例：

   - 对于新数据库实例，您可以在启动实例时应用选项组。有关更多信息，请参阅“创建 Amazon RDS 数据库实例”。

- 对于现有数据库实例，您可以通过修改实例并附加新的选项组来应用选项组。有关更多信息，请参阅“修改 Amazon RDS 数据库实例”。

使用 Oracle Locator

在启用 Oracle Locator 选项后，可以开始使用它。您应仅使用 Oracle Locator 功能。请勿使用任何 Oracle Spatial 功能，除非您有 Oracle Spatial 的许可证。

有关 Oracle Locator 支持的功能的列表，请参阅 Oracle 文档中的 Locator 附带的功能。

有关 Oracle Locator 不支持的功能的列表，请参阅 Oracle 文档中的 Locator 未附带的功能。

删除 Oracle Locator 选项

删除使用 `LOCATOR` 选项提供的数据类型的所有对象后，您可以从数据库实例中删除该选项。如果数据库实例上未 安装 Oracle Java 虚拟机 (JVM)，则删除 `LOCATOR` 选项时会出现短暂中断。如果数据库实例上已安装 Oracle Java 虚拟机 (JVM)，则不会出现中断。删除 `LOCATOR` 选项后，您无需重新启动数据库实例。

删除 **LOCATOR** 选项

1. 备份您的数据。

   > **Warning**
   >
   > 如果实例使用作为选项一部分启用的数据类型，并且您删除了 `LOCATOR` 选项，则可能会丢失数据。有关更多信息，请参阅“备份、还原和导出数据”。

2. 检查是否有任何现有对象引用了 `LOCATOR` 选项的数据类型或功能。

   如果存在 `LOCATOR` 选项，则在应用无 `LOCATOR` 选项的新选项组时，实例可能会卡住。您可以使用以下查询标识对象：

```
SELECT OWNER, SEGMENT_NAME, TABLESPACE_NAME, BYTES/1024/1024 mbytes
FROM   DBA_SEGMENTS
WHERE  SEGMENT_TYPE LIKE '%TABLE%'
AND    (OWNER, SEGMENT_NAME) IN
       (SELECT DISTINCT OWNER, TABLE_NAME
        FROM   DBA_TAB_COLUMNS
        WHERE  DATA_TYPE='SDO_GEOMETRY'
```

```
        AND    OWNER <> 'MDSYS')
ORDER BY 1,2,3,4;

SELECT OWNER, TABLE_NAME, COLUMN_NAME
FROM   DBA_TAB_COLUMNS
WHERE  DATA_TYPE = 'SDO_GEOMETRY'
AND    OWNER <> 'MDSYS'
ORDER BY 1,2,3;
```

3. 删除引用了 LOCATOR 选项的数据类型或功能的任何对象。
4. 请执行下列操作之一：
   - 从 LOCATOR 选项所属的选项组中删除该选项。此更改会影响使用该选项组的所有数据库实例。有关更多信息，请参阅“从选项组中删除选项”。
   - 修改数据库实例并指定不包含 LOCATOR 选项的另一个选项组。此更改会影响单个数据库实例。您可以指定默认 (空) 选项组，或指定其他自定义选项组。有关更多信息，请参阅“修改 Amazon RDS 数据库实例”。

Oracle 本机网络加密

Amazon RDS 支持 Oracle 本机网络加密 (NNE)。利用 NATIVE_NETWORK_ENCRYPTION 选项，您可以在数据进出数据库实例时对数据进行加密。Amazon RDS 支持所有版本的 Oracle Database 的 NNE。

详细讨论 Oracle Native Network Encryption 超出了本指南的范围，但在您确定部署解决方案前，应该了解每个算法和密钥的优缺点。有关通过 Oracle 本机网络加密提供的算法和密钥的信息，请参阅 Oracle 文档中的配置网络数据加密。有关Amazon安全性的更多信息，请参阅Amazon安全中心。

> **Note**
> 您可以使用本机网络加密或安全套接字层，但不能同时使用二者。有关更多信息，请参阅 Oracle 安全套接字层。

主题

NATIVE_NETWORK_ENCRYPTION 选项设置

您可以在服务器和客户端上指定加密要求。例如，当数据库实例使用数据库链接连接到另一个数据库时，它可以充当客户端。您可能希望避免在服务器端强制加密。例如，您可能不希望强制所有客户端通信使用加密，因为服务器需要加密。在这种情况下，您可以使用 SQLNET.*CLIENT 选项强制在客户端加密。

Amazon RDS 支持 NATIVE_NETWORK_ENCRYPTION 选项的以下设置。

> **Note**
> 当您使用逗号分隔选项设置的值时，不要在逗号后面放置空格。

选项设置	有效值	默认值	描述
SQLNET.ALLOW_WEAK_CRYPTO_CLIENTS	TRUE, FALSE	TRUE	使用非安全密码的客户端尝试连接到数据库时服务器的行为。如果为 TRUE，即使客户没有使用 2021 年 7 月 PSU 修补也可以连接。 如果设置为 FALSE，客户端只有在使用 2021 年 7 月 PSU 修补后才能连接到数据库。将 SQLNET.ALLOW_WEAK_CRYPTO_CLIENTS 设置为 FALSE 之前，确保满足以下条件： • SQLNET.ENCRYPTION_TYPES_SERVER 和 SQLNET.ENCRYPTION_TYPES_CLIENT 有一种并非 DES、3DES 或 RC4 的匹配加密方法（所有密钥长度）。 • SQLNET.CHECKSUM_TYPES_SERVER 和 SQLNET.CHECKSUM_TYPES_CLIENT 有一种并非 MD5 的匹配安全校验和方法。 • 该客户端使用 2021 年 7 月的 PSU 进行了修补。如果客户端没有进行修补，则客户端将失去连接并收到 ORA-12269 错误。
SQLNET.ALLOW_WEAK_CRYPTO	TRUE, FALSE	TRUE	使用非安全密码的客户端尝试连接到数据库时服务器的行为。以下密码被视为不安全： • DES 加密方法（所有密钥长度） • 3DES 加密方法（所有密钥长度） • RC4 加密方法（所有密钥长度） • MD5 校验和方法

选项设置	有效值	默认值	描述
			如果设置为 TRUE，则客户端可以在使用上述非安全密码时进行连接。 如果设置为 FALSE，则数据库会在客户端可以在使用上述非安全密码时阻止客户端进行连接。将 SQLNET.ALLOW_WEAK_CRYPTO 设置为 FALSE 之前，确保满足以下条件： • SQLNET.ENCRYPTION_TYPES_SERVER 和 SQLNET.ENCRYPTION_TYPES_CLIENT 有一种并非 DES、3DES 或 RC4 的匹配加密方法（所有密钥长度）。 • SQLNET.CHECKSUM_TYPES_SERVER 和 SQLNET.CHECKSUM_TYPES_CLIENT 有一种并非 MD5 的匹配安全校验和方法。 • 该客户端使用 2021 年 7 月的 PSU 进行了修补。如果客户端没有进行修补，则客户端将失去连接并收到 ORA-12269 错误。
SQLNET.CRYPTO_CHECKSUM_CLIENT	Accepted, Rejected, Requested, Required	Requested	当客户端或充当客户端的服务器连接到数据库实例时的数据集成行为。当数据库实例使用数据库链接时，它充当客户端。 Requested 表示数据库实例不需要数据库实例执行校验和。

选项设置	有效值	默认值	描述
SQLNET.CRYPTO_CHECKSUM_SERVER	Accepted, Rejected, Requested, Required	Requested	客户端或充当客户端的服务器连接到数据库实例时的数据集成行为。当数据库实例使用数据库链接时，它充当客户端。 Requested 表示数据库实例不需要客户端执行校验和。
SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENT	SHA256, SHA384, SHA512, SHA1, MD5	SHA256, SHA384, SHA512	校验和算法的列表。 您可以指定一个值，也可以指定逗号分隔的值列表。如果您使用逗号，请不要在逗号后插入空格；否则，您会收到 InvalidParameterValue 错误。 此参数和 SQLNET.CRYPTO_CHECKSUM_TYPES_SERVER 必须有一个通用的密码。
SQLNET.CRYPTO_CHECKSUM_TYPES_SERVER	SHA256, SHA384, SHA512, SHA1, MD5	SHA256, SHA384, SHA512, SHA1, MD5	校验和算法的列表。 您可以指定一个值，也可以指定逗号分隔的值列表。如果您使用逗号，请不要在逗号后插入空格；否则，您会收到 InvalidParameterValue 错误。 此参数和 SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENT 必须有一个通用的密码。
SQLNET.ENCRYPTION_CLIENT	Accepted, Rejected, Requested, Required	Requested	客户端或充当客户端的服务器连接到数据库实例时的客户端加密行为。当数据库实例使用数据库链接时，它充当客户端。 Requested 表示数据库实例不需要给来自服务器的流量加密。

选项设置	有效值	默认值	描述
SQLNET.ENCRYPTION_SERVER	Accepted Rejected Requested , Required	Requested	客户端或充当客户端的服务器连接到数据库实例时的服务器加密行为。当数据库实例使用数据库链接时，它充当客户端。 Requested 表示数据库实例不需要给来自客户端的流量加密。

选项设置	有效值	默认值	描述
SQLNET.ENCRYPTION_ TYPES_CLIENT	RC4_256, AES256, AES192, 3DES168, RC4_128, AES128, 3DES112, RC4_56, DES, RC4_40, DES40	RC4_256, AES256, AES192, 3DES168, RC4_128, AES128, 3DES112, RC4_56, DES, RC4_40, DES40	客户端使用的加密算法的列表。客户端尝试通过按顺序尝试每种算法来对服务器输入进行解密，直到某种算法成功或到达列表末尾。 Amazon RDS 使用 Oracle 中的下列默认列表。RDS 从 RC4_256 开始，然后沿列表按顺序下移。您可以更改顺序或限制数据库实例将接受的算法。 1. RC4_256：RSA RC4 (256 位密钥大小) 2. AES256：AES (256 位密钥大小) 3. AES192：AES (192 位密钥大小) 4. 3DES168：3–密钥三重–DES（112 位有效密钥大小） 5. RC4_128：RSA RC4 (128 位密钥大小) 6. AES128：AES (128 位密钥大小) 7. 3DES112：2-密钥三重-DES（80 位有效密钥大小） 8. RC4_56：RSA RC4（56 位密钥大小） 9. DES：标准 DES（56 位密钥大小） 10. RC4_40：RSA RC4（40 位密钥大小） 11. DES40：DES40（40 位密钥大小） 您可以指定一个值，也可以指定逗号分隔的值列表。如果使用逗号，请不要在逗号后插入空格；否则，您会收到 InvalidParameterValue 错误。

选项设置	有效值	默认值	描述
			此参数和 SQLNET.SQLNET.ENCRYPTION_TYPES_SERVER 必须有一个通用的密码。

选项设置	有效值	默认值	描述
SQLNET.ENCRYPTION_TYPES_SERVER	RC4_256, AES256, AES192, 3DES168, RC4_128, AES128, 3DES112, RC4_56, DES, RC4_40, DES40	RC4_256, AES256, AES192, 3DES168, RC4_128, AES128, 3DES112, RC4_56, DES, RC4_40, DES40	数据库实例使用的加密算法的列表。数据库实例使用每种算法，以便尝试对客户端输入解密，直到某种算法成功或到达列表末尾。 Amazon RDS 使用 Oracle 中的下列默认列表。您可以更改顺序或限制客户端将接受的算法。 1. RC4_256：RSA RC4 (256 位密钥大小) 2. AES256：AES (256 位密钥大小) 3. AES192：AES (192 位密钥大小) 4. 3DES168：3–密钥三重–DES（112 位有效密钥大小） 5. RC4_128：RSA RC4 (128 位密钥大小) 6. AES128：AES (128 位密钥大小) 7. 3DES112：2-密钥三重-DES（80 位有效密钥大小） 8. RC4_56：RSA RC4（56 位密钥大小） 9. DES：标准 DES（56 位密钥大小） 10. RC4_40：RSA RC4（40 位密钥大小） 11. DES40：DES40（40 位密钥大小） 您可以指定一个值，也可以指定逗号分隔的值列表。如果使用逗号，请不要在逗号后插入空格；否则，您会收到 InvalidParameterValue 错误。

选项设置	有效值	默认值	描述
			此参数和 SQLNET.SQLNET.ENCRYPTION_TYPES_SERVER 必须有一个通用的密码。

添加 NATIVE_NETWORK_ENCRYPTION 选项

将 NATIVE_NETWORK_ENCRYPTION 选项添加到数据库实例的一般过程如下：

1. 创建新的选项组，或者复制或修改现有选项组。
2. 将 选项添加到该选项组。
3. 将选项组与数据库实例相关联。

当选项组处于活动状态时，NNE 就处于活动状态。

使用 Amazon Web Services Management Console将 NATIVE_NETWORK_ENCRYPTION 选项添加到数据库实例

1. 对于 Engine，请选择要使用的 Oracle 版本。NNE 在所有版本中都受支持。
2. 对于主引擎版本，选择数据库实例的版本。

   有关更多信息，请参阅 创建选项组。
3. 将 NATIVE_NETWORK_ENCRYPTION 选项添加到该选项组。有关添加选项的更多信息，请参阅将选项添加到选项组。

   > **Note**
   >
   > 添加 NATIVE_NETWORK_ENCRYPTION 选项后，您无需重新启动数据库实例。只要选项组处于活动状态，NNE 就处于活动状态。
4. 将选项组应用到新的或现有的数据库实例：

   - 对于新数据库实例，您可以在启动实例时应用选项组。有关更多信息，请参阅“创建 Amazon RDS 数据库实例”。

- 对于现有数据库实例，您可以通过修改实例并附加新的选项组来应用选项组。添加 NATIVE_NETWORK_ENCRYPTION 选项后，您无需重新启动数据库实例。只要选项组处于活动状态，NNE 就处于活动状态。有关更多信息，请参阅 修改 Amazon RDS 数据库实例。

在 sqlnet.ora 中设置 NNE 值

借助于 Oracle Native Network Encryption，您还可以在服务器端和客户端设置网络加密。客户端是用来连接到数据库实例的计算机。您可以在 slqnet.ora 中指定以下客户端设置：

- `SQLNET.ALLOW_WEAK_CRYPTO`
- `SQLNET.ALLOW_WEAK_CRYPTO_CLIENTS`
- `SQLNET.CRYPTO_CHECKSUM_CLIENT`
- `SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENT`
- `SQLNET.ENCRYPTION_CLIENT`
- `SQLNET.ENCRYPTION_TYPES_CLIENT`

有关信息，请参阅 Oracle 文档中的配置 Oracle 服务器和客户端的网络数据加密和完整性。

有时，数据库实例会拒绝来自应用程序的连接请求。例如，当客户端和服务器上的加密算法不匹配时，可能会发生拒绝。要测试 Oracle Native Network Encryption，请给客户端上的 sqlnet.ora 文件添加下列行：

```
DIAG_ADR_ENABLED=off
TRACE_DIRECTORY_CLIENT=/tmp
TRACE_FILE_CLIENT=nettrace
TRACE_LEVEL_CLIENT=16
```

当尝试连接时，先前的行会在称为 `/tmp/nettrace*` 的客户端上生成跟踪文件。该跟踪文件中包含关于连接的信息。有关使用 Oracle 本机网络加密时的相关连接问题的详细信息，请参阅 Oracle Database 文档中的关于协商加密和完整性。

修改 NATIVE_NETWORK_ENCRYPTION 选项设置

启用 `NATIVE_NETWORK_ENCRYPTION` 选项后，您可以修改其设置。目前，您只能使用 Amazon CLI 或 RDS API 修改 `NATIVE_NETWORK_ENCRYPTION` 选项设置。您无法使用控制台。以下示例修改该选项中的两个设置。

```
aws rds add-option-to-option-group \
    --option-group-name my-option-group \
    --options
 "OptionName=NATIVE_NETWORK_ENCRYPTION,OptionSettings=[{Name=SQLNET.CRYPTO_CHECKSUM_TYPES_SERVE
{Name=SQLNET.CRYPTO_CHECKSUM_TYPES_SERVER,Value=SHA256}]" \
    --apply-immediately
```

要了解如何使用 CLI 修改选项设置，请参阅Amazon CLI。有关各项设置的更多信息，请参阅NATIVE_NETWORK_ENCRYPTION 选项设置。

主题

- 修改 CRYPTO_CHECKSUM_* 值
- 修改 ALLOW_WEAK_CRYPTO* 设置

修改 CRYPTO_CHECKSUM_* 值

如果修改 NATIVE_NETWORK_ENCRYPTION 选项设置，请确保以下选项设置具有至少一个通用密码：

- SQLNET.CRYPTO_CHECKSUM_TYPES_SERVER
- SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENT

以下示例显示修改 SQLNET.CRYPTO_CHECKSUM_TYPES_SERVER 的方案。该配置有效，因为 CRYPTO_CHECKSUM_TYPES_CLIENT 和 CRYPTO_CHECKSUM_TYPES_SERVER 都使用 SHA256。

选项设置	修改前的值	修改后的值
SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENT	**SHA256**, SHA384, SHA512	无更改
SQLNET.CRYPTO_CHECKSUM_TYPES_SERVER	**SHA256**, SHA384, SHA512, SHA1, MD5	SHA1,MD5, **SHA256**

例如，假设需要将 SQLNET.CRYPTO_CHECKSUM_TYPES_SERVER 从默认设置改为 SHA1,MD5。在这种情况下，请确保将 SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENT 设置为 SHA1 或者 MD5。这些算法不包含 SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENT 的默认值中。

修改 ALLOW_WEAK_CRYPTO* 设置

要将 SQLNET.ALLOW_WEAK_CRYPTO* 选项从默认值设置为 FALSE，请确保满足以下条件：

- SQLNET.ENCRYPTION_TYPES_SERVER 和 SQLNET.ENCRYPTION_TYPES_CLIENT 有一种匹配的安全加密方法。如果方法并非 DES、3DES 或 RC4（所有密钥长度），则该方法被视为安全。
- SQLNET.CHECKSUM_TYPES_SERVER 和 SQLNET.CHECKSUM_TYPES_CLIENT 有一种匹配的安全校验和方法。如果方法并非 MD5，则该方法被视为安全。
- 该客户端使用 2021 年 7 月的 PSU 进行了修补。如果客户端没有进行修补，则客户端将失去连接并收到 ORA-12269 错误。

下面的示例显示了示例 NNE 设置。假设您想要将 SQLNET.ENCRYPTION_TYPES_SERVER 和 SQLNET.ENCRYPTION_TYPES_CLIENT 设置为 FALSE，从而阻止不安全的连接。校验和选项设置满足先决条件，因为它们都具有 SHA256。但是，SQLNET.ENCRYPTION_TYPES_CLIENT 和 SQLNET.ENCRYPTION_TYPES_SERVER 使用 DES、3DES 和 RC4 加密方法，这些方法不安全。因此，要将 SQLNET.ALLOW_WEAK_CRYPTO* 选项设置为 FALSE，请首先将 SQLNET.ENCRYPTION_TYPES_SERVER 和 SQLNET.ENCRYPTION_TYPES_CLIENT 设置为安全加密方法，例如 AES256。

选项设置	值
SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENT	SHA256, SHA384, SHA512
SQLNET.CRYPTO_CHECKSUM_TYPES_SERVER	SHA1,MD5,SHA256
SQLNET.ENCRYPTION_TYPES_CLIENT	RC4_256, 3DES168, DES40
SQLNET.ENCRYPTION_TYPES_SERVER	RC4_256, 3DES168, DES40

删除 NATIVE_NETWORK_ENCRYPTION 选项

您可以从数据库实例中删除 NNE。

要从数据库实例中删除 NATIVE_NETWORK_ENCRYPTION 选项，请执行下列操作之一：

- 要从多个数据库实例中删除该选项，请从其所属的选项组中删除 NATIVE_NETWORK_ENCRYPTION 选项。此更改会影响使用该选项组的所有数据库实例。删除 NATIVE_NETWORK_ENCRYPTION 选项后，您无需重新启动数据库实例。有关更多信息，请参阅 从选项组中删除选项。
- 要从单个数据库实例中删除此选项，请修改该数据库实例，并指定另一个不包含 NATIVE_NETWORK_ENCRYPTION 选项的选项组。您可以指定默认 (空) 选项组，或指定其他自定义选项组。删除 NATIVE_NETWORK_ENCRYPTION 选项后，您无需重新启动数据库实例。有关更多信息，请参阅 修改 Amazon RDS 数据库实例。

Oracle OLAP

Amazon RDS 通过使用 OLAP 选项支持 Oracle OLAP。此选项为 Oracle 数据库实例提供在线分析处理 (OLAP)。您可以使用 Oracle OLAP，根据 OLAP 标准创建维度对象和多维数据集来分析大量数据。有关更多信息，请参阅 Oracle 文档。

> **Important**
>
> 如果您使用 Oracle OLAP，则当存在通用漏洞评分系统 (CVSS) 评分为 9+ 的安全漏洞或其他公布的安全漏洞时，Amazon RDS 会自动将数据库实例更新为最新的 Oracle PSU。

Amazon RDS 支持 Enterprise Edition 的 Oracle Database 19c 及更高版本的 Oracle LAP。

Oracle OLAP 的先决条件

以下是使用 Oracle OLAP 的先决条件：

- 您必须拥有 Oracle 提供的 Oracle OLAP 许可证。有关更多信息，请参阅 Oracle 文档中的许可信息。
- 您的数据库实例必须具有足够的实例类。db.t3.small 数据库实例类不支持 Oracle OLAP。有关更多信息，请参阅 RDS for Oracle 数据库实例类。
- 数据库实例必须已启用自动次要版本升级。该选项允许数据库实例在次要数据库引擎版本升级可用时自动接收这些升级，这对于安装 Oracle Java 虚拟机（JVM）的任意选项是必需的。Amazon RDS 使用该选项将数据库实例更新为最新的 Oracle 补丁集更新（PSU）或版本更新（RU）。有关更多信息，请参阅“修改 Amazon RDS 数据库实例”。
- 您的数据库实例不能有名为 OLAPSYS 的用户。如果是这样，OLAP 选项安装将失败。

Oracle OLAP 的最佳实践

以下是使用 Oracle OLAP 的最佳实践：

- 为了实现最高安全性，可将 OLAP 选项与安全套接字层 (SSL) 结合使用。有关更多信息，请参阅“Oracle 安全套接字层”。
- 配置数据库实例以限制对它的访问。有关更多信息，请参阅“在 VPC 中访问数据库实例的场景”和“在 VPC 中使用数据库实例”。

添加 Oracle OLAP 选项

下面是将 OLAP 选项添加到数据库实例的一般过程：

1. 创建新的选项组，或者复制或修改现有选项组。
2. 将 选项添加到该选项组。
3. 将选项组与数据库实例相关联。

如果数据库实例上未 安装 Oracle Java 虚拟机 (JVM)，则添加 OLAP 选项时会出现短暂中断。如果数据库实例上已安装 Oracle Java 虚拟机 (JVM)，则不会出现中断。在添加 选项后，您无需重新启动数据库实例。在选项组处于活动状态后，Oracle OLAP 将立即可用。

将 OLAP 选项添加到数据库实例

1. 确定您要使用的选项组。您可以创建新的选项组，或使用现有选项组。如果您想使用现有选项组，请跳到下一步。或者，通过以下设置创建自定义数据库选项组：

   - 对于引擎，请选择适用于您的数据库实例的 Oracle 版本。
   - 对于主引擎版本，选择数据库实例的版本。

   有关更多信息，请参阅“创建选项组”。
2. 将 OLAP 选项添加到该选项组。有关添加选项的更多信息，请参阅 将选项添加到选项组。
3. 将选项组应用到新的或现有的数据库实例：

   - 对于新数据库实例，请在启动实例时应用选项组。有关更多信息，请参阅“创建 Amazon RDS 数据库实例”。
   - 对于现有数据库实例，请修改实例并附加新的选项组以应用选项组。有关更多信息，请参阅“修改 Amazon RDS 数据库实例”。

使用 Oracle OLAP

在启用 Oracle OLAP 选项后，可以开始使用它。有关 Oracle OLAP 支持的功能的列表，请参阅 Oracle 文档。

删除 Oracle OLAP 选项

删除使用 OLAP 选项提供的数据类型的所有对象后，您可以从数据库实例中删除该选项。如果数据库实例上未 安装 Oracle Java 虚拟机 (JVM)，则删除 OLAP 选项时会出现短暂中断。如果数据库实例上已安装 Oracle Java 虚拟机 (JVM)，则不会出现中断。删除 OLAP 选项后，您无需重新启动数据库实例。

删除 **OLAP** 选项

1. 备份您的数据。

   > **Warning**
   > 如果实例使用作为选项一部分启用的数据类型，并且您删除了 OLAP 选项，则可能会丢失数据。有关更多信息，请参阅"备份、还原和导出数据"。

2. 检查是否有任何现有对象引用了 OLAP 选项的数据类型或功能。
3. 删除引用了 OLAP 选项的数据类型或功能的任何对象。
4. 请执行下列操作之一：
   - 从 OLAP 选项所属的选项组中删除该选项。此更改会影响使用该选项组的所有数据库实例。有关更多信息，请参阅"从选项组中删除选项"。
   - 修改数据库实例并指定不包含 OLAP 选项的另一个选项组。此更改会影响单个数据库实例。您可以指定默认 (空) 选项组，或指定其他自定义选项组。有关更多信息，请参阅"修改 Amazon RDS 数据库实例"。

Oracle 安全套接字层

要为 RDS for Oracle 数据库实例启用 SSL 加密，请向与数据库实例关联的选项组添加 Oracle SSL 选项。根据 Oracle 的要求，Amazon RDS 使用另一个端口连接 SSL。此方法允许数据库实例与 SQL*Plus 之间能够同时进行明文通信和 SSL 加密通信。例如，您可以使用用于明文通信的端口与 VPC 内的其他资源通信，同时使用用于 SSL 加密通信的端口与 VPC 外部的资源通信。

> **Note**
>
> 您可以在同一 RDS for Oracle 数据库实例上使用 SSL 或原生网络加密（NNE），但不能同时使用两者。如果您使用 SSL 加密，则必须禁用任何其他连接加密。有关更多信息，请参阅 Oracle 本机网络加密。

SSL/TLS 和 NNE 不再是 Oracle 高级安全的一部分。在 RDS for Oracle 中，您可以对以下数据库版本的所有许可版本使用 SSL 加密：

- Oracle Database 21c (21.0.0)
- Oracle Database 19c（19.0.0）

主题

Oracle SSL 选项的 TLS 版本

Amazon RDS for Oracle 支持传输层安全性 (TLS) 版本 1.0 和 1.2。在添加新的 Oracle SSL 选项时，请显式将 `SQLNET.SSL_VERSION` 设置为有效值。下面是该选项设置允许的值：

- "1.0" – 客户端只能使用 TLS 版本 1.0 连接到数据库实例。对于现有 Oracle SSL 选项，SQLNET.SSL_VERSION 自动设置为 "1.0"。您可以根据需要更改设置。
- "1.2" – 客户端只能使用 TLS 1.2 连接到数据库实例。
- "1.2 or 1.0" – 客户端只能使用 TLS 1.2 或 1.0 连接到数据库实例。

Oracle SSL 选项的密码套件

Amazon RDS for Oracle 支持多个 SSL 密码套件。默认情况下，Oracle SSL 选项配置为使用 SSL_RSA_WITH_AES_256_CBC_SHA 密码套件。要指定不同的密码套件以在 SSL 连接中使用，请使用 SQLNET.CIPHER_SUITE 选项设置。

可以为 SQLNET.CIPHER_SUITE 指定多个值。如果数据库实例之间具有数据库链接并且您决定更新密码套件，则此技术很有用。

下表总结了所有版本的 Oracle Database 19c 和 21c 中对 RDS for Oracle 的 SSL 支持。

密码套件（SQLNET.CIPHER_SUITE）	TLS 版本支持（SQLNET.SSL_VERSION）	FIPS 支持	符合 FedRAMP 要求
SSL_RSA_WITH_AES_256_CBC_SHA（原定设置）	1.0 和 1.2	是	否
SSL_RSA_WITH_AES_256_CBC_SHA256	1.2	是	否
SSL_RSA_WITH_AES_256_GCM_SHA384	1.2	是	否
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384	1.2	是	是
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	1.2	是	是
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384	1.2	是	是
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256	1.2	是	是

密码套件（SQLNET.CIPHER_SUITE）	TLS 版本支持（SQLNET.SSL_VERSION）	FIPS 支持	符合 FedRAMP 要求
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA	1.2	是	是
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA	1.2	是	是

FIPS 支持

RDS for Oracle 使您能够使用 140-2 联邦信息处理标准（FIPS）。FIPS 140-2 是一项美国政府标准，它定义了密码模块的安全要求。您可以通过针对 Oracle SSL 选项将 `FIPS.SSLFIPS_140` 设置为 `TRUE` 来开启 FIPS 标准。为 SSL 配置 FIPS 140-2 时，加密库会加密客户端与 RDS for Oracle 数据库实例之间的数据。

客户端必须使用符合 FIPS 条件的密码套件。建立连接时，客户端和 RDS for Oracle 数据库实例协商为来回传输消息使用的密码套件。Oracle SSL 选项的密码套件中的表显示了适用于各个 TLS 版本的符合 FIPS 条件的 SSL 密码套件。有关更多信息，请参阅 Oracle 数据库文档中的 Oracle 数据库 FIPS 140-2 设置。

添加 SSL 选项

要使用 SSL，您的 RDS for Oracle 数据库实例必须与包括 `SSL` 选项的选项组关联。

控制台

将 SSL 选项添加到选项组

1. 创建新选项组或识别现有选项组，然后您可以将 `SSL` 选项添加到其中。

   有关创建选项组的信息，请参阅 创建选项组。
2. 将 `SSL` 选项添加到该选项组。

   如果要为 SSL 连接仅使用 FIPS 验证的密码套件，请将选项 `FIPS.SSLFIPS_140` 设置为 `TRUE`。有关 FIPS 标准的信息，请参阅 FIPS 支持。

有关向选项组添加选项的信息，请参阅 将选项添加到选项组。

3. 创建新的 RDS for Oracle 数据库实例并将选项组与其关联，或者修改要将选项组与其关联的 RDS for Oracle 数据库实例。

   有关创建数据库实例的信息，请参阅 创建 Amazon RDS 数据库实例。

   有关修改数据库实例的信息，请参阅 修改 Amazon RDS 数据库实例。

Amazon CLI

将 SSL 选项添加到选项组

1. 创建新选项组或识别现有选项组，然后您可以将 SSL 选项添加到其中。

   有关创建选项组的信息，请参阅 创建选项组。

2. 将 SSL 选项添加到该选项组。

   指定以下选项设置：

   - `Port` – SSL 端口号
   - `VpcSecurityGroupMemberships` – 为其启用此选项的 VPC 安全组
   - `SQLNET.SSL_VERSION` – 客户端可用来连接到数据库实例的 TLS 版本

   例如，以下 Amazon CLI 命令将 SSL 选项添加到名为 `ora-option-group` 的选项组。

   Example

   对于 Linux、macOS 或 Unix：

```
aws rds add-option-to-option-group --option-group-name ora-option-group \
  --options
 'OptionName=SSL,Port=2484,VpcSecurityGroupMemberships="sg-68184619",OptionSettings=[{Name=
```

   对于 Windows：

```
aws rds add-option-to-option-group --option-group-name ora-option-group ^
```

```
  --options
 'OptionName=SSL,Port=2484,VpcSecurityGroupMemberships="sg-68184619",OptionSettings=[{Name=
```

3. 创建新的 RDS for Oracle 数据库实例并将选项组与其关联，或者修改要将选项组与其关联的 RDS for Oracle 数据库实例。

   有关创建数据库实例的信息，请参阅 创建 Amazon RDS 数据库实例。

   有关修改数据库实例的信息，请参阅 修改 Amazon RDS 数据库实例。

配置 SQL*Plus 以将 SSL 用于 RDS for Oracle 数据库实例

在连接到使用 Oracle SSL 选项的 RDS for Oracle 数据库实例之前，您必须先配置 SQL*Plus。

> **Note**
> 若要允许从适当的客户端访问数据库实例，请确保已正确配置安全组。有关更多信息，请参阅"使用安全组控制访问权限"。另外，这些说明适用于 SQL*Plus 和其他直接使用 Oracle 主目录的客户端。有关 JDBC 连接，请参阅设置通过 JDBC 的 SSL 连接。

将 SQL*Plus 配置为使用 SSL 连接到 RDS for Oracle 数据库实例

1. 将 `ORACLE_HOME` 环境变量设置为 Oracle 主目录的位置。

   Oracle 主目录的路径取决于您的安装。以下示例设置 `ORACLE_HOME` 环境变量。

   ```
   prompt>export ORACLE_HOME=/home/user/app/user/product/19.0.0/dbhome_1
   ```

   有关设置 Oracle 环境变量的信息，请参阅 Oracle 文档中的 SQL*Plus 环境变量，另请参阅适用于您的操作系统的 Oracle 安装指南。

2. 将 `$ORACLE_HOME/lib` 追加到 `LD_LIBRARY_PATH` 环境变量。

   下面是一个设置 LD_LIBRARY_PATH 环境变量的示例。

   ```
   prompt>export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$ORACLE_HOME/lib
   ```

3. 为 `$ORACLE_HOME/ssl_wallet` 处的 Oracle wallet 创建一个目录。

下面是一个创建 Oracle wallet 目录的示例。

```
prompt>mkdir $ORACLE_HOME/ssl_wallet
```

4. 下载适用于所有 Amazon Web Services 区域的证书捆绑 .pem 文件并将此文件放在 ssl_wallet 目录中。有关信息，请参阅使用 SSL/TLS 加密与数据库实例或集群的连接。
5. 在 `$ORACLE_HOME/network/admin` 目录中，修改或创建 `tnsnames.ora` 文件并包含下列条目。

```
net_service_name =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS =
        (PROTOCOL = TCPS)
        (HOST = endpoint)
        (PORT = ssl_port_number)
      )
    )
    (CONNECT_DATA =
      (SID = database_name)
    )
    (SECURITY =
      (SSL_SERVER_CERT_DN =
 "C=US,ST=Washington,L=Seattle,O=Amazon.com,OU=RDS,CN=endpoint")
    )
  )
```

6. 在同一目录中，修改或创建 sqlnet.ora 文件并包含下列参数。

> **Note**
>
> 要通过 TLS 安全连接与实体进行通信，Oracle 需要包含身份验证所需证书的钱夹。您可以使用 Oracle 的 ORAPKI 实用工具来创建和维护 Oracle 钱夹，如步骤 7 中所示。有关更多信息，请参阅 Oracle 文档中的使用 ORAPKI 设置 Oracle 钱夹。

```
WALLET_LOCATION = (SOURCE = (METHOD = FILE) (METHOD_DATA = (DIRECTORY =
 $ORACLE_HOME/ssl_wallet)))
SSL_CLIENT_AUTHENTICATION = FALSE
SSL_VERSION = 1.0
```

```
SSL_CIPHER_SUITES = (SSL_RSA_WITH_AES_256_CBC_SHA)
SSL_SERVER_DN_MATCH = ON
```

> **Note**
>
> 您可以将 SSL_VERSION 设置为较高值（如果您的数据库实例支持它）。

7. 运行下列命令以创建 Oracle Wallet。

```
prompt>orapki wallet create -wallet $ORACLE_HOME/ssl_wallet -auto_login_only
```

8. 使用操作系统实用程序将 .pem 捆绑文件中的每个证书解压缩到单独的 .pem 文件中。
9. 使用单独的 `orapki` 命令将每个证书添加到 Wallet 中，将 *certificate-pem-file* 替换为 .pem 文件的绝对文件名。

```
prompt>orapki wallet add -wallet $ORACLE_HOME/ssl_wallet -trusted_cert -cert
      certificate-pem-file -auto_login_only
```

有关更多信息，请参阅 [轮换 SSL/TLS 证书]。

使用 SSL 连接到 RDS for Oracle 数据库实例

在如前所述将 SQL*Plus 配置为使用 SSL 之后，您可以使用 SSL 选项连接到 RDS for Oracle 数据库实例。您也可以选择首先导出指向包含 tnsnames.ora 和 sqlnet.ora 文件的目录的 TNS_ADMIN 值。这样做可确保 SQL*Plus 可始终找到这些文件。以下示例导出 TNS_ADMIN 值。

```
export TNS_ADMIN = ${ORACLE_HOME}/network/admin
```

连接到数据库实例。例如，您可以使用 SQL*Plus，通过 tnsnames.ora 文件中的 *<net_service_name>* 进行连接。

```
sqlplus mydbuser@net_service_name
```

您还可以通过以下命令，使用 SQL*Plus 连接到数据库实例，而无需使用 tnsnames.ora 文件。

```
sqlplus 'mydbuser@(DESCRIPTION = (ADDRESS = (PROTOCOL = TCPS)(HOST = endpoint) (PORT
 = ssl_port_number))(CONNECT_DATA = (SID = database_name)))'
```

您还可在不使用 SSL 的情况下连接到 RDS for Oracle 数据库实例。例如，以下命令通过不带 SSL 加密的明文端口连接到数据库实例。

```
sqlplus 'mydbuser@(DESCRIPTION = (ADDRESS = (PROTOCOL = TCP)(HOST = endpoint) (PORT
 = port_number))(CONNECT_DATA = (SID = database_name)))'
```

如果要关闭传输控制协议 (TCP) 端口访问，请创建一个没有 IP 地址入口的安全组并将此组添加到实例。添加此组将会关闭通过 TCP 端口的连接，同时仍然允许从 SSL 选项安全组允许的范围内的 IP 地址通过 SSL 端口进行指定的连接。

设置通过 JDBC 的 SSL 连接

要使用通过 JDBC 的 SSL 连接，您必须创建密钥存储，信任 Amazon RDS 根 CA 证书并使用下面指定的代码段。

要创建 JKS 格式的 keystore，可以使用以下命令。有关创建 keystore 的更多信息，请参阅 Oracle 文档中的[创建 keystore](#)。有关参考信息，请参阅《Java Platform, Standard Edition Tools Reference》中的 [keytool](#)。

```
keytool -genkey -alias client -validity 365 -keyalg RSA -keystore clientkeystore
```

按照下列步骤信任 Amazon RDS 根 CA 证书。

信任 Amazon RDS 根 CA 证书

1. 下载适用于所有 Amazon Web Services 区域的证书捆绑 .pem 文件并将此文件放在 ssl_wallet 目录中。

   有关下载证书的信息，请参阅 [使用 SSL/TLS 加密与数据库实例或集群的连接](#)。
2. 使用操作系统实用程序将 .pem 文件中的每个证书解压缩到单独的文件中。
3. 使用单独的 `openssl` 命令将每个证书转换为 .der 格式，同时将证书 *certificate-pem-file* 替换为证书 .pem 文件的名称（不带 .pem 扩展名）。

   ```
   openssl x509 -outform der -in certificate-pem-file.pem -out certificate-pem-
   file.der
   ```
4. 使用以下命令将每个证书导入 keystore。

```
keytool -import -alias rds-root -keystore clientkeystore.jks -file certificate-pem-
file.der
```

有关更多信息，请参阅 轮换 SSL/TLS 证书。

5. 确认成功创建了密钥存储。

```
keytool -list -v -keystore clientkeystore.jks
```

在出现提示时，输入密钥存储密码。

以下代码说明如何设置使用 JDBC 的 SSL 连接。

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.util.Properties;

public class OracleSslConnectionTest {
    private static final String DB_SERVER_NAME = "dns-name-provided-by-amazon-rds";
    private static final Integer SSL_PORT = "ssl-option-port-configured-in-option-
group";
    private static final String DB_SID = "oracle-sid";
    private static final String DB_USER = "user-name";
    private static final String DB_PASSWORD = "password";
    // This key store has only the prod root ca.
    private static final String KEY_STORE_FILE_PATH = "file-path-to-keystore";
    private static final String KEY_STORE_PASS = "keystore-password";

    public static void main(String[] args) throws SQLException {
        final Properties properties = new Properties();
        final String connectionString = String.format(
                "jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCPS)(HOST=%s)(PORT=
%d))(CONNECT_DATA=(SID=%s)))",
                DB_SERVER_NAME, SSL_PORT, DB_SID);
        properties.put("user", DB_USER);
        properties.put("password", DB_PASSWORD);
        properties.put("oracle.jdbc.J2EE13Compliant", "true");
        properties.put("javax.net.ssl.trustStore", KEY_STORE_FILE_PATH);
        properties.put("javax.net.ssl.trustStoreType", "JKS");
```

```
        properties.put("javax.net.ssl.trustStorePassword", KEY_STORE_PASS);
        final Connection connection = DriverManager.getConnection(connectionString,
 properties);
        // If no exception, that means handshake has passed, and an SSL connection can
 be opened
    }
}
```

> **Note**
>
> 作为安全最佳实践，请指定除此处所示提示以外的密码。

使用 SSL 连接强制执行 DN 匹配

您可以使用 Oracle 参数 SSL_SERVER_DN_MATCH 强制数据库服务器的可分辨名称 (DN) 与其服务名称匹配。如果您强制执行匹配验证，则 SSL 将确保证书来自服务器。如果您未强制执行匹配验证，则 SSL 将执行检查，但允许连接，而不管是否存在匹配项。如果您未强制执行匹配，则会允许服务器潜在的伪造身份行为。

要强制执行 DN 匹配，请添加 DN 匹配属性并使用下面指定的连接字符串。

将属性添加到客户端连接以强制执行 DN 匹配。

```
properties.put("oracle.net.ssl_server_dn_match", "TRUE");
```

使用 SSL 时使用以下连接字符串强制执行 DN 匹配。

```
final String connectionString = String.format(
    "jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCPS)(HOST=%s)(PORT=%d))" +
    "(CONNECT_DATA=(SID=%s))" +
    "(SECURITY = (SSL_SERVER_CERT_DN =
\"C=US,ST=Washington,L=Seattle,O=Amazon.com,OU=RDS,CN=%s\")))",
    DB_SERVER_NAME, SSL_PORT, DB_SID, DB_SERVER_NAME);
```

排查 SSL 连接问题

您可能会查询数据库并收到 ORA-28860 错误。

```
ORA-28860: Fatal SSL error
```

```
28860. 00000 - "Fatal SSL error"
*Cause: An error occurred during the SSL connection to the peer. It is likely that this
 side sent data which the peer rejected.
*Action: Enable tracing to determine the exact cause of this error.
```

当客户端尝试使用服务器不支持的 TLS 版本进行连接时，就会发生此错误。要避免此错误，请编辑 sqlnet.ora 并将 `SSL_VERSION` 设置为正确的 TLS 版本。有关更多信息，请参阅 My Oracle Support 中的 Oracle 支持文档 2748438.1。

Oracle Spatial

Amazon RDS 通过使用 `SPATIAL` 选项支持 Oracle Spatial。Oracle Spatial 提供了一个 SQL 架构和一些功能，可帮助在 Oracle 数据库中存储、检索、更新和查询空间数据集合。有关更多信息，请参阅 Oracle 文档中的 Spatial 概念。Amazon RDS 在所有受支持发行版的所有版本中均支持 Oracle Spatial。

Spatial 补丁包（SPB）的工作原理

每个季度，RDS for Oracle 都会为每个受支持的主要引擎发布新的次要引擎版本。版本更新（RU）引擎版本通过包含指定季度的 RU 补丁，纳入了 Oracle 提供的错误修复。Spatial 补丁包（SPB）引擎版本包含 RU 补丁以及特定于 Oracle Spatial 的补丁。例如，19.0.0.0.ru-2025-01.spb-1.r1 是一个次要引擎版本，其中包含引擎版本 19.0.0.0.ru-2025-01.rur-2025-01.r1 中的 RU 补丁以及 Spatial 补丁。只有 Oracle Database 19c 才支持 SPB。

尽管 SPB 与 RU 的命名方式不同，但它们的功能相同。RU 使用命名格式 19.0.0.0.ru-2025-01.rur-2025-01.r1。SPB 名称包括文本“spb”，如 19.0.0.0.ru-2025-01.spb-1.r1。通常，SPB 在其相应的季度 RU 发布后的 2-3 周内发布。例如，19.0.0.0.ru-2025-01.spb-1.r1 在 19.0.0.0.ru-2025-01.rur-2025-01.r1 之后发布。

RDS for Oracle 具有单独的途径来自动升级 RU 和 SPB 的次要版本。如果数据库实例使用 RU，则 RDS 会自动将实例升级到 RU。如果数据库实例使用 SPB，则 RDS 会将实例升级到 SPB。当您在数据库实例中添加 Oracle Spatial 选项并启用自动升级时，RDS 会将实例置于 SPB 的升级路径中。

有关 RU 和 SPB 的更多信息，请参阅 Oracle 次要版本升级。有关 Oracle Database 19c 支持的 RU 和 SPB 的列表，请参阅《Amazon RDS for Oracle Release Notes》中的 Amazon RDS for Oracle Database 19c (19.0.0.0)。

使用 Oracle Spatial 的先决条件

以下是使用 Oracle Spatial 的先决条件：

- 确保您的数据库实例具有足够的实例类。db.t3.small 数据库实例类不支持 Oracle Spatial。有关更多信息，请参阅 RDS for Oracle 数据库实例类。
- 确保您的数据库实例已启用自动次要版本升级。该选项允许数据库实例在次要数据库引擎版本升级可用时自动接收这些升级，这对于安装 Oracle Java 虚拟机（JVM）的任意选项是必需的。Amazon RDS 使用该选项将数据库实例更新为最新的 Oracle 补丁集更新（PSU）或版本更新（RU）。有关更多信息，请参阅“修改 Amazon RDS 数据库实例”。

Oracle Spatial 的最佳实践

以下是使用 Oracle Spatial 的最佳实践：

- 为了实现最高安全性，可将 SPATIAL 选项与安全套接字层 (SSL) 结合使用。有关更多信息，请参阅"Oracle 安全套接字层"。
- 配置数据库实例以限制对它的访问。有关更多信息，请参阅"在 VPC 中访问数据库实例的场景"和"在 VPC 中使用数据库实例"。

添加 Oracle Spatial 选项

下面是将 SPATIAL 选项添加到数据库实例的一般过程：

1. 创建新的选项组，或者复制或修改现有选项组。
2. 将 选项添加到该选项组。
3. 将选项组与数据库实例相关联。

如果数据库实例上未 安装 Oracle Java 虚拟机 (JVM)，则添加 SPATIAL 选项时会出现短暂中断。如果数据库实例上已安装 Oracle Java 虚拟机 (JVM)，则不会出现中断。在添加 选项后，您无需重新启动数据库实例。只要选项组处于活动状态，Oracle Spatial 就可用。

> **Note**
>
> 在该中断期间，将短暂禁用密码验证功能。您还可能会在中断期间看到与密码验证功能相关的事件。在 Oracle 数据库实例变得可用之前，将再次启用密码验证功能。

将 **SPATIAL** 选项添加到数据库实例

1. 确定您要使用的选项组。您可以创建新的选项组，或使用现有选项组。如果您想使用现有选项组，请跳到下一步。或者，通过以下设置创建自定义数据库选项组：

   a. 对于引擎，请选择适用于您的数据库实例的 Oracle 版本。

   b. 对于主引擎版本，选择数据库实例的版本。

   有关更多信息，请参阅"创建选项组"。
2. 将 SPATIAL 选项添加到该选项组。有关添加选项的更多信息，请参阅 将选项添加到选项组。

3. 将选项组应用到新的或现有的数据库实例：

   • 对于新数据库实例，您可以在启动实例时应用选项组。有关更多信息，请参阅“创建 Amazon RDS 数据库实例”。

   • 对于现有数据库实例，您可以通过修改实例并附加新的选项组来应用选项组。有关更多信息，请参阅“修改 Amazon RDS 数据库实例”。

删除 Oracle Spatial 选项

删除使用 SPATIAL 选项提供的数据类型的所有对象后，可以从数据库实例中删除该选项。如果数据库实例上未 安装 Oracle Java 虚拟机 (JVM)，则删除 SPATIAL 选项时会出现短暂中断。如果数据库实例上已安装 Oracle Java 虚拟机 (JVM)，则不会出现中断。删除 SPATIAL 选项后，您无需重新启动数据库实例。

删除 **SPATIAL** 选项

1. 备份您的数据。

   > **Warning**
   >
   > 如果实例使用作为选项一部分启用的数据类型，并且您删除了 SPATIAL 选项，则可能会丢失数据。有关更多信息，请参阅“备份、还原和导出数据”。

2. 检查是否有任何现有对象引用了 SPATIAL 选项的数据类型或功能。

   如果存在 SPATIAL 选项，则在应用无 SPATIAL 选项的新选项组时，实例可能会卡住。您可以使用以下查询标识对象：

```
SELECT OWNER, SEGMENT_NAME, TABLESPACE_NAME, BYTES/1024/1024 mbytes
FROM   DBA_SEGMENTS
WHERE  SEGMENT_TYPE LIKE '%TABLE%'
AND    (OWNER, SEGMENT_NAME) IN
       (SELECT DISTINCT OWNER, TABLE_NAME
        FROM   DBA_TAB_COLUMNS
        WHERE  DATA_TYPE='SDO_GEOMETRY'
        AND    OWNER <> 'MDSYS')
ORDER BY 1,2,3,4;

SELECT OWNER, TABLE_NAME, COLUMN_NAME
FROM   DBA_TAB_COLUMNS
```

```
WHERE  DATA_TYPE = 'SDO_GEOMETRY'
AND    OWNER <> 'MDSYS'
ORDER BY 1,2,3;
```

3. 删除引用了 SPATIAL 选项的数据类型或功能的任何对象。
4. 请执行下列操作之一：
   - 从 SPATIAL 选项所属的选项组中删除该选项。此更改会影响使用该选项组的所有数据库实例。有关更多信息，请参阅“从选项组中删除选项”。
   - 修改数据库实例并指定不包含 SPATIAL 选项的另一个选项组。此更改会影响单个数据库实例。您可以指定默认 (空) 选项组，或指定其他自定义选项组。有关更多信息，请参阅“修改 Amazon RDS 数据库实例”。

Oracle SQLT

Amazon RDS 通过使用 SQLT 选项来支持 Oracle SQLTXPLAIN (SQLT)。您可对任一版本的 Oracle Database 19c 及更高版本使用 SQLT。

Oracle `EXPLAIN PLAN` 语句可以确定 SQL 语句的执行计划。它可以验证 Oracle 优化程序是否选择某个执行计划，例如嵌套循环联接。它还可以帮助您了解优化程序的决定，例如为什么选择嵌套循环联接而不是哈希联接。所以 `EXPLAIN PLAN` 可帮助您了解该语句的效果。

SQLT 是一个生成报告的 Oracle 实用程序。报告包括对象统计信息、对象元数据、与优化程序相关的初始化参数，以及数据库管理员可用于调整 SQL 语句以获得最佳性能的其他信息。SQLT 会生成一个 HTML 报告，其中包含报表中所有部分的超链接。

与 Automatic Workload Repository 或 Statspack 报告不同，SQLT 适用于单个 SQL 语句。SQLT 是收集、存储和显示性能数据的 SQL、PL/SQL 和 SQL*Plus 文件的集合。

下面是每个 SQLT 版本支持的 Oracle 版本。

SQLT 版本	Oracle Database 21c	Oracle Database 19c
2018-07-25.v1	支持	支持
2018-03-31.v1	不支持	不支持
2016-04-29.v1	不支持	不支持

下载 SQLT 并了解它的使用说明：

- 登录到 My Oracle Support 账户，然后打开以下文档：
- 下载 SQLT：[文档 215187.1]
- SQLT 使用说明：[文档 1614107.1]
- 有关 SQLT 的常见问题：[文档 1454160.1]
- 有关读取 SQLT 输出的信息：[文档 1456176.1]
- 主报告解释：[文档 1922234.1]

Amazon RDS 不支持以下 SQLT 方法：

- XPLORE
- XHUME

SQLT 的先决条件

以下是使用 SQLT 的先决条件：

- 您必须删除 SQLT 所需的用户和角色 (如果存在)。

  SQLT 选项将在数据库实例上创建以下用户和角色：

  - SQLTXPLAIN 用户
  - SQLTXADMIN 用户
  - SQLT_USER_ROLE 角色

  如果数据库实例具有这些用户或角色中的任意一个，请使用 SQL 客户端登录到数据库实例，并使用以下语句将其删除：

```
DROP USER SQLTXPLAIN CASCADE;
DROP USER SQLTXADMIN CASCADE;
DROP ROLE SQLT_USER_ROLE CASCADE;
```

- 您必须删除 SQLT 所需的表空间 (如果存在)。

  SQLT 选项将在数据库实例上创建以下表空间：

  - RDS_SQLT_TS
  - RDS_TEMP_SQLT_TS

  如果数据库实例有这些表空间，请使用 SQL 客户端登录数据库实例，并将其删除。

SQLT 选项设置

SQLT 可以使用由 Oracle Tuning Pack 和 Oracle Diagnostics Pack 提供的许可功能。Oracle Tuning Pack 包括 SQL Tuning Advisor，Oracle Diagnostics Pack 包括 Automatic Workload Repository。SQLT 设置可允许或禁止从 SQLT 访问这些功能。

Amazon RDS 支持 SQLT 选项的以下设置。

选项设置	有效值	默认值	描述
LICENSE_PACK	T, D, N	N	要使用 SQLT 访问的 Oracle 管理包。输入以下值之一： • T 指示您拥有 Oracle Tuning Pack 和 Oracle Diagnostics Pack 的许可，并且希望从 SQLT 访问 SQL Tuning Advisor 和 Automatic Workload Repository。 • D 指示您拥有 Oracle Diagnostics Pack 的许可，并且希望从 SQLT 访问 Automatic Workload Repository。 • N 指示您没有 Oracle Tuning Pack 和 Oracle Diagnostics Pack 的许可，或者您拥有其中一项或两项的许可，但不希望 SQLT 访问它们。 Note Amazon RDS 不提供这些 Oracle 管理包的许可。如果您指示要使用未包含在数据库实例中的包，则可以对数据库实例使用 SQLT。不过，SQLT 无法访问该包，并且 SQLT 报告不包括该包的数据。例如，如果您指定 T，但数据库实例不包括 Oracle Tuning Pack，则 SQLT 可用于数据库实例，但其生成的报告不包含与 Oracle Tuning Pack 相关的数据。
VERSION	2016-04-29.v1	2016-04-29.v1	要安装的 SQLT 版本。

选项设置	有效值	默认值	描述
	2018-03-31.v1 2018-07-25.v1		**Note** 对于 Oracle Database 19c 和 21c，唯一受支持的版本是 2018-07-25.v1 。此版本是这些发布的默认版本。

添加 SQLT 选项

下面是将 SQLT 选项添加到数据库实例的一般过程：

1. 创建新的选项组，或者复制或修改现有选项组。
2. 向选项组添加 SQLT 选项。
3. 将选项组与数据库实例相关联。

在添加 SQLT 选项后，一旦激活该选项组，就会激活 SQLT。

将 SQLT 选项添加到数据库实例

1. 确定您要使用的选项组。您可以创建新的选项组，或使用现有选项组。如果您想使用现有选项组，请跳到下一步。或者，通过以下设置创建自定义数据库选项组：

   a. 对于 Engine，请选择要使用的 Oracle 版本。所有版本均支持 SQLT 选项。
   b. 对于主引擎版本，选择数据库实例的版本。

   有关更多信息，请参阅“创建选项组”。
2. 将 SQLT 选项添加到该选项组。有关添加选项的更多信息，请参阅 将选项添加到选项组。
3. 将选项组应用到新的或现有的数据库实例：

   • 对于新数据库实例，您可以在启动实例时应用选项组。有关更多信息，请参阅“创建 Amazon RDS 数据库实例”。

   • 对于现有数据库实例，您可以通过修改实例并附加新的选项组来应用选项组。有关更多信息，请参阅“修改 Amazon RDS 数据库实例”。

4. (可选) 使用 SQLT 选项验证每个数据库实例上的 SQLT 安装。

   a. 使用 SQL 客户端以主用户身份连接到数据库实例。

      有关使用 SQL 客户端连接到 Oracle 数据库实例的信息，请参阅连接到 Oracle 数据库实例。

   b. 运行以下查询：

      ```
      SELECT sqltxplain.sqlt$a.get_param('tool_version') sqlt_version FROM DUAL;
      ```

      查询返回 Amazon RDS 上的 SQLT 选项的当前版本。12.1.160429 是 Amazon RDS 上可用的 SQLT 版本的一个例子。

5. 更改由 SQLT 选项创建的用户的密码。

   a. 使用 SQL 客户端以主用户身份连接到数据库实例。

   b. 运行以下 SQL 语句更改 SQLTXADMIN 用户的密码：

      ```
      ALTER USER SQLTXADMIN IDENTIFIED BY new_password ACCOUNT UNLOCK;
      ```

      > **Note**
      >
      > 作为安全最佳实践，请指定除此处所示提示以外的密码。

   c. 运行以下 SQL 语句更改 SQLTXPLAIN 用户的密码：

      ```
      ALTER USER SQLTXPLAIN IDENTIFIED BY new_password ACCOUNT UNLOCK;
      ```

      > **Note**
      >
      > 作为安全最佳实践，请指定除此处所示提示以外的密码。

> **Note**
>
> 升级 SQLT 需要卸载 SQLT 的旧版本然后安装新版本。因此，在升级 SQLT 时，所有 SQLT 元数据都可能会丢失。数据库的主要版本升级也会卸载并重新安装 SQLT。主版本升级的示例是从 Oracle Database 19c 升级到 Oracle Database 21c。

使用 SQLT

SQLT 可与 Oracle SQL*Plus 实用程序一起使用。

使用 SQLT

1. 从 My Oracle Support 网站上的文档 215187.1 下载 SQLT .zip 文件。

   > **Note**
   >
   > 您无法从 My Oracle Support 网站中下载 SQLT 12.1.160429。Oracle 已弃用该旧版本。

2. 解压缩 SQLT .zip 文件。
3. 从命令提示符处，转到文件系统上的 `sqlt/run` 目录。
4. 从命令提示符处，打开 SQL*Plus，并以主用户身份连接到数据库实例。

   有关使用 SQL*Plus 连接到数据库实例的信息，请参阅 连接到 Oracle 数据库实例。
5. 获取 SQL 语句的 SQL ID：

   ```
   SELECT SQL_ID FROM V$SQL WHERE SQL_TEXT='sql_statement';
   ```

   输出类似于以下内容：

   ```
   SQL_ID
   -------------
   chvsmttqjzjkn
   ```

6. 使用 SQLT 分析 SQL 语句：

```
START sqltxtract.sql sql_id sqltxplain_user_password
```

例如，对于 SQL ID chvsmttqjzjkn，输入以下内容：

```
START sqltxtract.sql chvsmttqjzjkn sqltxplain_user_password
```

SQLT 在运行 SQLT 命令的目录中生成 HTML 报告和相关资源作为 .zip 文件。

7. (可选) 要允许应用程序用户使用 SQLT 诊断 SQL 语句，请使用以下语句向每个应用程序用户授予 SQLT_USER_ROLE：

```
GRANT SQLT_USER_ROLE TO application_user_name;
```

> **Note**
>
> Oracle 不建议对 SYS 用户或具有 DBA 角色的用户运行 SQLT。最佳实践是通过向应用程序用户授予 SQLT_USER_ROLE，使用应用程序用户账户运行 SQLT 诊断。

升级 SQLT 选项

对于 Amazon RDS for Oracle，您可以将 SQLT 选项从现有版本升级到更高版本。要升级 SQLT 选项，请完成使用 SQLT中的步骤 1–3 以升级到新的 SQLT 版本。此外，如果您在本节的步骤 7 中为以前版本的 SQLT 授予了权限，请再次为新的 SQLT 版本授予权限。

升级 SQLT 选项将导致旧 SQLT 版本的元数据丢失。删除旧 SQLT 版本的架构和相关对象，并安装新版本的 SQLT。有关 SQLT 最新版本中的更改的更多信息，请参阅 My Oracle Support 网站上的文档 1614201.1。

> **Note**
> 不支持版本降级。

修改 SQLT 设置

在启用 SQLT 后，您可以修改该选项的 `LICENSE_PACK` 和 `VERSION` 设置。

有关如何修改选项设置的更多信息，请参阅 修改选项设置。有关各项设置的更多信息，请参阅SQLT 选项设置。

删除 SQLT 选项

您可以从数据库实例中删除 SQLT。

要从数据库实例中删除 SQLT，请执行下列操作之一：

- 要从多个数据库实例中删除 SQLT，请从数据库实例所属的选项组中删除 SQLT 选项。此更改会影响使用该选项组的所有数据库实例。有关更多信息，请参阅“从选项组中删除选项”。
- 要从单个数据库实例中删除 SQLT，请修改该数据库实例，并指定另一个不包含 SQLT 选项的选项组。您可以指定默认（空）选项组，或指定其他自定义选项组。有关更多信息，请参阅 修改 Amazon RDS 数据库实例。

Oracle Statspack

Oracle Statspack 选项安装并启用 Oracle Statspack 性能统计数据功能。Oracle Statspack 是收集、存储和显示性能数据的 SQL、PL/SQL 和 SQL*Plus 脚本的集合。有关使用 Oracle Statspack 信息，请参阅 Oracle 文档中的 Oracle Statspack。

> **Note**
>
> Oracle 不再支持 Oracle Statspack，并用更高级的 Automatic Workload Repository (AWR) 取而代之。只有已购买 Diagnostics Pack 的 Oracle 企业版客户才能使用 AWR。您可以在 Amazon RDS 上将 Oracle Statspack 与任何 Oracle 数据库引擎一起使用。您无法在 Amazon RDS 只读副本上运行 Oracle Statspack。

设置 Oracle Statspack

要运行 Statspack 脚本，必须添加 Statspack 选项。

设置 Oracle Statspack

1. 在 SQL 客户端中，使用管理账户登录到 Oracle 数据库。
2. 根据是否安装了 Statspack，执行以下操作之一：

   - 如果已安装 Statspack ，并且 `PERFSTAT` 账户与 Statspack 关联，请跳到步骤 4。
   - 如果尚未安装 Statspack，并且 `PERFSTAT` 账户存在，请按如下方式删除账户：

     ```
     DROP USER PERFSTAT CASCADE;
     ```

     否则，尝试添加 Statspack 选项会生成错误和 `RDS-Event-0058`。
3. 将 Statspack 选项添加到选项组。请参阅将选项添加到选项组。

   Amazon RDS 会自动在数据库实例上安装 Statspack 脚本，然后设置 `PERFSTAT` 账户。
4. 使用以下 SQL 语句重置密码，并将 pwd 替换为新密码：

   ```
   ALTER USER PERFSTAT IDENTIFIED BY pwd ACCOUNT UNLOCK;
   ```

   您可以使用 `PERFSTAT` 用户账户登录并运行 Statspack 脚本。
5. 使用以下语句向 `PERFSTAT` 账户授予 `CREATE JOB` 权限：

```
GRANT CREATE JOB TO PERFSTAT;
```

6. 确保填充 PERFSTAT.STATS$IDLE_EVENT 表中的空闲等待事件。

   由于 Oracle 错误 28523746，可能无法填充 PERFSTAT.STATS$IDLE_EVENT 中的空闲等待事件。要确保所有空闲事件都可用，请运行以下语句：

```
INSERT INTO PERFSTAT.STATS$IDLE_EVENT (EVENT)
SELECT NAME FROM V$EVENT_NAME WHERE WAIT_CLASS='Idle'
MINUS
SELECT EVENT FROM PERFSTAT.STATS$IDLE_EVENT;
COMMIT;
```

生成 Statspack 报告

Statspack 报告比较两个快照。

生成 Statspack 报告

1. 在 SQL 客户端中，使用 PERFSTAT 账户登录到 Oracle 数据库。
2. 使用以下任一方法创建快照：

   - 手动创建 Statspack 快照。
   - 创建一个在给定时间间隔后拍摄 Statspack 快照的作业。例如，以下作业每小时创建一个 Statspack 快照：

```
VARIABLE jn NUMBER;
exec dbms_job.submit(:jn, 'statspack.snap;',SYSDATE,'TRUNC(SYSDATE
+1/24,''HH24'')');
COMMIT;
```

3. 使用以下查询查看快照：

```
SELECT SNAP_ID, SNAP_TIME FROM STATS$SNAPSHOT ORDER BY 1;
```

4. 运行 Amazon RDS 过程 `rdsadmin.rds_run_spreport`，用快照 ID 替换 begin_snap 和 end_snap。

```
exec rdsadmin.rds_run_spreport(begin_snap,end_snap);
```

例如，以下命令基于 Statspack 快照 1 和 2 之间的间隔创建报告：

```
exec rdsadmin.rds_run_spreport(1,2);
```

Statspack 报告的文件名包括两个快照的编号。例如，使用 Statspack 快照 1 和 2 创建的报告文件将命名为 `ORCL_spreport_1_2.lst`。

5. 监视输出是否存在错误。

   Oracle Statspack 会在运行报表之前执行检查。因此，您还可以在命令输出中看到错误消息。例如，您可能会尝试基于无效范围生成报告，其中起始 Statspack 快照值大于结束值。在这种情况下，输出显示错误消息，但数据库引擎不会生成错误文件。

```
exec rdsadmin.rds_run_spreport(2,1);
*
ERROR at line 1:
ORA-20000: Invalid snapshot IDs. Find valid ones in perfstat.stats$snapshot.
```

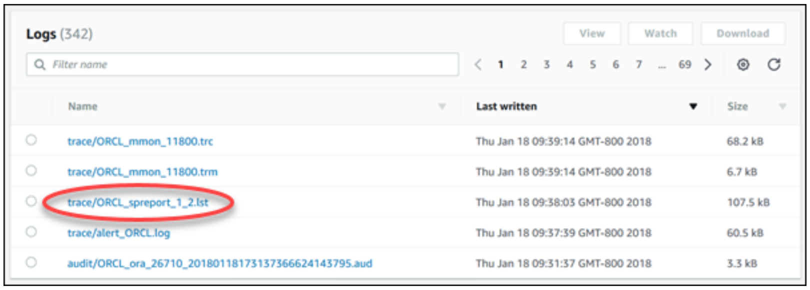
   如果您使用的 Statspack 快照编号无效，则输出将显示错误。例如，如果您尝试为快照 1 和 50 生成报告，但快照 50 不存在，则输出将显示错误。

```
exec rdsadmin.rds_run_spreport(1,50);
*
ERROR at line 1:
ORA-20000: Could not find both snapshot IDs
```

6. (可选)

   要检索报告，请调用跟踪文件过程，如 [使用 Oracle 跟踪文件](#) 中所述。

   或者，从 RDS 控制台下载 Statspack 报告。转到数据库实例详细信息的日志部分，然后选择下载。下面的示例显示了 `trace/ORCL_spreport_1_2.lst`



如果在生成报告时发生错误，则数据库引擎使用与报告相同的命名约定，但扩展名为 `.err`。例如，如果在使用 Statspack 快照 1 和 7 创建报告时出错，报告文件将命名为 `ORCL_spreport_1_7.err`。您可以使用与标准快照报告相同的方法下载错误报告。

删除 Statspack 快照

要删除一系列 Oracle Statspack 快照，请使用以下命令：

```
exec statspack.purge(begin snap, end snap);
```

Oracle 时区

要更改您的 Oracle 数据库实例使用的系统时区，请使用时区选项。例如，您可能需要更改数据库实例的时区，从而与本地环境或旧版应用程序兼容。时区选项在主机级别更改时区。更改时区会影响所有日期列和值 (包括 SYSDATE 和 SYSTIMESTAMP)。

时区选项与 `rdsadmin_util.alter_db_time_zone` 命令不同。`alter_db_time_zone` 命令只能更改某些数据类型的时区。时区选项会更改所有日期列和值的时区。有关 `alter_db_time_zone` 的更多信息，请参阅设置数据库时区。有关升级注意事项的更多信息，请参阅 时区注意事项。

有关设置时区的限制

时区选项是永久、持续的选项。因此，您无法执行以下操作：

- 添加时区选项后，将从选项组中删除该选项。
- 添加选项后，将其从数据库实例中删除。
- 将此选项的时区设置修改为其他时区。

有关设置时区的建议

将时区选项添加到生产数据库之前，我们强烈建议您执行以下操作：

- 为您的数据库实例拍摄快照。如果不小心错误地设置了时区，则必须将数据库实例恢复到其以前的时区设置。有关更多信息，请参阅 为 Amazon RDS 的单可用区数据库实例创建数据库快照。
- 为测试数据库实例添加时区选项。对于使用系统日期添加日期或时间的表，添加时区选项可能引起问题。建议您对测试实例上的数据和应用程序进行分析。这样一来，您便能评测更改生产实例上的时区所产生的影响。

如果数据库实例使用默认选项组，请执行以下步骤：

1. 为您的数据库实例拍摄快照。
2. 向数据库实例添加时区选项。

如果您的数据库实例当前使用非默认选项组，请执行以下步骤：

1. 为您的数据库实例拍摄快照。
2. 创建新的选项组。

3. 向其中添加时区选项，以及当前与现有选项组关联的所有其他选项。

   这样可以防止在启用时区选项时卸载现有选项。
4. 将选项添加到您的数据库实例。

时区选项设置

Amazon RDS 支持时区选项的以下设置。

选项设置	有效值	描述
TIME_ZONE	可用时区之一。完整列表请参阅 可用时区。	数据库实例的新时区。

添加时区选项

完成以下步骤以将时区选项添加到数据库实例：

1. （推荐）为您的数据库实例拍摄快照。
2. 请执行以下任务之一：
   - 从头开始创建一个新选项组。有关更多信息，请参阅 创建选项组。
   - 使用 Amazon CLI 或 API 复制现有选项组。有关更多信息，请参阅 复制选项组。
   - 重复使用现有的非默认选项组。最佳实践是使用当前未与任何数据库实例或快照关联的选项组。
3. 将新选项添加到您在前面的步骤中创建的选项组。
4. 如果当前与您的数据库实例关联的选项组已启用一些选项，请将这些选项添加到您的新选项组。此策略可以防止在启用新选项时卸载现有选项。
5. 将新选项添加到您的数据库实例。

添加时区选项时，数据库实例在自动重新启动时会出现短暂中断。

控制台

将时区选项添加到一个选项组并将该选项组与数据库实例关联

1. 在 RDS 控制台中，选择选项组。

2. 选择要将选项添加到的选项组的名称。
3. 选择添加选项。
4. 对于选项名称，选择时区，然后配置选项设置。
5. 将该选项组与新的或现有的数据库实例关联：
   - 对于新数据库实例，请在启动实例时应用选项组。有关更多信息，请参阅“创建 Amazon RDS 数据库实例”。
   - 对于现有数据库实例，请修改实例并附加新的选项组以应用选项组。如果您将新选项添加到现有数据库实例，则在数据库实例自动重新启动时会出现短暂中断。有关更多信息，请参阅 修改 Amazon RDS 数据库实例。

Amazon CLI

以下示例使用 Amazon CLI add-option-to-option-group 命令将 `Timezone` 选项和 `TIME_ZONE` 选项设置添加到名为 `myoptiongroup` 的选项组。时区设置为 `Africa/Cairo`。

对于 Linux、macOS 或 Unix：

```
aws rds add-option-to-option-group \
    --option-group-name "myoptiongroup" \
    --options "OptionName=Timezone,OptionSettings=[{Name=TIME_ZONE,Value=Africa/
Cairo}]" \
    --apply-immediately
```

对于 Windows：

```
aws rds add-option-to-option-group ^
    --option-group-name "myoptiongroup" ^
    --options "OptionName=Timezone,OptionSettings=[{Name=TIME_ZONE,Value=Africa/
Cairo}]" ^
    --apply-immediately
```

修改时区设置

时区选项是永久、持续的选项。将此选项添加到选项组后不能将其删除。将选项组添加到数据库实例后不能将其删除。您不能将此选项的时区设置修改为其他时区。在未能正确设置时区的情况下，可以使用添加时区选项之前的数据库实例快照进行恢复。

删除时区选项

时区选项是永久、持续的选项。将此选项添加到选项组后不能将其删除。将选项组添加到数据库实例后不能将其删除。要删除时区选项，可以使用添加时区选项之前的数据库实例快照进行恢复。

可用时区

以下值可用于时区选项。

区	时区
非洲	非洲/开罗、非洲/卡萨布兰卡、非洲/哈拉雷、非洲/拉各斯、非洲/罗安达、非洲/蒙罗维亚、非洲/内罗毕、非洲/的黎波里、非洲/温得和克
美洲	美洲/阿拉瓜伊纳、美洲/阿根廷/布宜诺斯艾利斯、美洲/亚森松、美洲/Bogota、美洲/加拉加斯、美洲/芝加哥、美洲/奇瓦瓦、美洲/库亚巴、美洲/丹佛、美洲/底特律、美洲/福塔莱萨、美洲/戈特霍布、美洲/危地马拉、美洲/哈利法克斯、美洲/利马、美洲/洛杉矶、美洲/玛瑙斯、美洲/马塔莫罗斯、美洲/墨西哥城、美洲/蒙特雷、美洲/蒙特维多、美洲/纽约、美洲/菲尼克斯、美洲/圣地亚哥、美洲/圣保罗、美洲/蒂华纳、美洲/多伦多
亚洲	亚洲/阿曼、亚洲/阿什哈巴德、亚洲/巴格达、亚洲/巴库、亚洲/曼谷、亚洲/贝鲁特、亚洲/加尔各答、亚洲/大马士革、亚洲/达卡、亚洲/香港、亚洲/伊尔库茨克、亚洲/雅加达、亚洲/耶路撒冷、亚洲/喀布尔、亚洲/卡拉奇、亚洲/加德满都、亚洲/加尔各答、亚洲/克拉斯诺亚尔斯克、亚洲/马加丹、亚洲/马尼拉、亚洲/马斯喀特、亚洲/新西伯利亚、亚洲/阳光、亚洲/利雅得、亚洲/首尔、亚洲/上海、亚洲/新加坡、亚洲/台北、亚洲/德黑兰、亚洲/东京、亚洲/乌兰巴托、亚洲/符拉迪沃斯托克、亚洲/雅库茨克、亚洲/埃里温
大西洋	大西洋/亚速尔、大西洋/佛得角
澳大利亚	澳大利亚/阿德莱德、澳大利亚/布里斯班、澳大利亚/达尔文、澳大利亚/尤克拉、澳大利亚/霍巴特、澳大利亚/豪勋爵岛、澳大利亚/珀斯、澳大利亚/悉尼
巴西	巴西/迪诺罗尼亚、巴西/东部
加拿大	加拿大/纽芬兰、加拿大/萨斯喀彻温省
ETC	ETC/GMT-3

区	时区
欧洲	欧洲/阿姆斯特丹、欧洲/雅典、欧洲/柏林、欧洲/都柏林、欧洲/赫尔辛基、欧洲/加里宁格勒、欧洲/伦敦、欧洲/马德里、欧洲/莫斯科、欧洲/巴黎、欧洲/布拉格、欧洲/罗马、欧洲/萨拉热窝
太平洋	太平洋/阿皮亚、太平洋/奥克兰、太平洋/查塔姆、太平洋/斐济、太平洋/关岛、太平洋/檀香山、太平洋/圣诞岛、太平洋/玛贵斯、太平洋/萨摩亚、太平洋/汤加塔布岛、太平洋/韦克
US	美国/阿拉斯加、美国/中部、美国/印第安纳东部、美国/东部、美国/太平洋地区
UTC	UTC

Oracle 时区文件自动升级

使用 `TIMEZONE_FILE_AUTOUPGRADE` 选项，您可以将当前时区文件升级到 RDS for Oracle 数据库实例上的最新版本。

主题

Oracle 时区文件概览

Oracle Database 时区文件存储以下信息：

- 相对于协调世界时（UTC）的偏移量
- 夏令时（DST）的过渡时间
- 标准时间和 DST 的缩写

Oracle Database 提供多个版本的时区文件。在本地环境中创建 Oracle 数据库时，可以选择时区文件版本。有关更多信息，请参阅《Oracle Database 全球化支持指南》中的选择时区文件。

如果 DST 的规则发生变化，Oracle 将发布新的时区文件。Oracle 发布这些新的时区文件与每季度版本更新（RU）和版本更新修订（RUR）的时间表无关。时区文件位于数据库主机上的目录 `$ORACLE_HOME/oracore/zoneinfo/` 中。时区文件名使用的格式为 DstV*version*，如 DSTV35 所示。

时区文件如何影响数据传输

在 Oracle 数据库中，`TIMESTAMP WITH TIME ZONE` 数据类型存储时间戳和时区数据。`TIMESTAMP WITH TIME ZONE` 数据类型的数据使用关联时区文件版本中的规则。这样，当您更新时区文件时，现有 `TIMESTAMP WITH TIME ZONE` 数据会受到影响。

在使用不同版本时区文件的数据库之间传输数据时，可能会出现问题。例如，如果您从时区文件版本比目标数据库更高的源数据库导入数据，数据库会发出 ORA-39405 错误。以前，您必须使用以下任一方法解决此错误：

- 使用所需的时区文件创建 RDS for Oracle 数据库实例，从源数据库导出数据，然后将其导入到新数据库中。
- 使用 Amazon DMS 或逻辑复制来迁移数据。

使用 TIMEZONE_FILE_AUTOUPGRADE 选项进行自动更新

当附加到 RDS for Oracle 数据库实例的选项组包括 `TIMEZONE_FILE_AUTOUPGRADE` 选项时，RDS 将自动更新您的时区文件。通过确保 Oracle 数据库使用相同的时区文件版本，可以避免在不同环境之间移动数据时采用耗时的手动方法。容器数据库（CDB）和非 CDB 均支持 `TIMEZONE_FILE_AUTOUPGRADE` 选项。

当您向选项组添加 `TIMEZONE_FILE_AUTOUPGRADE` 选项时，您可以选择是立即添加此选项，还是在维护时段添加此选项。数据库实例应用新选项后，RDS 会检查它是否可以安装更新的 DSTv*version* 文件。目标 DSTv*version* 取决于以下内容：

- 数据库实例当前正在运行的次要引擎版本
- 您要将数据库实例升级到的次要引擎版本

例如，您当前的时区文件版本可能是 DSTv33。当 RDS 将更新应用到选项组时，它可能会确定 DSTv34 在数据库实例文件系统上当前可用。然后，RDS 自动将您的时区文件更新到 DSTv34。

要在支持的 RDS 版本更新中查找可用的 DST 版本，请查看适用于 Oracle 的 Amazon Relational Database Service（Amazon RDS）发布说明中的补丁。例如，版本 19.0.0.0.ru-2022-10.rur-2022-10.r1 列出补丁 34533061: RDBMS - DSTV39 UPDATE - TZDATA2022C。

更新时区文件的策略

升级数据库引擎和将 `TIMEZONE_FILE_AUTOUPGRADE` 选项添加到选项组是单独的操作。如果有较新的时区文件可用，则添加 `TIMEZONE_FILE_AUTOUPGRADE` 选项会启动对时区文件的更新。您可以立即或在下一个维护时段运行以下命令（仅显示相关选项）：

- 仅使用以下 RDS CLI 命令升级数据库引擎：

```
modify-db-instance --engine-version name ...
```

- 仅使用以下 CLI 命令添加 `TIMEZONE_FILE_AUTOUPGRADE` 选项：

```
add-option-to-option-group --option-group-name name --options
 OptionName=TIMEZONE_FILE_AUTOUPGRADE ...
```

- 使用以下 CLI 命令升级您的数据库引擎并向您的实例添加新的选项组：

```
modify-db-instance --engine-version name --option-group-name name ...
```

更新策略取决于是要一起升级数据库和时区文件，还是只执行其中一个操作。请记住，如果您更新选项组，然后在单独的 API 操作中升级数据库引擎，则在升级数据库引擎时，当前可能正在进行时区文件更新。

本部分中的示例假定以下内容：

- 您尚未将 `TIMEZONE_FILE_AUTOUPGRADE` 添加到当前与您的数据库实例关联的选项组。
- 您的数据库实例使用数据库版本 19.0.0.0.ru-2019-07.rur-2019-07.r1 和时区文件 DSTv33。
- 您的数据库实例文件系统包含文件 DSTV34。
- 版本更新 19.0.0.0.ru-2022-10.rur-2022-10.r1 包含 DSTv35。

要更新时区文件，您可以使用以下策略。

主题

- [更新时区文件而不升级引擎]
- [升级时区文件和数据库引擎版本]
- [升级数据库引擎版本而不更新时区文件]

更新时区文件而不升级引擎

在这种情况下，您的数据库使用 DSTv33，但 DSTv34 在您的数据库实例文件系统上可用。您希望将数据库实例使用的时区文件从 DSTv33 更新到 DSTv34，但您不想将您的引擎升级到新的次要版本，其中包括 DSTv35。

在 add-option-to-option-group 命令中，将 TIMEZONE_FILE_AUTOUPGRADE 添加到您的数据库实例使用的选项组。指定是立即添加此选项，还是将其推迟到维护时段。应用 TIMEZONE_FILE_AUTOUPGRADE 选项后，RDS 将执行以下操作：

1. 检查是否有新的 DST 版本。
2. 确定 DSTv34 在文件系统上是否可用。
3. 立即更新时区文件。

升级时区文件和数据库引擎版本

在这种情况下，您的数据库使用 DSTv33，但 DSTv34 在您的数据库实例文件系统上可用。您希望将数据库引擎升级到包含 DSTv35 的次要版本 19.0.0.0.ru-2022-10.rur-2022-10.r1，并在引擎升级期间将时区文件更新到 DSTv5。这样，您的目标是跳过 DSTv34 并将时区文件直接更新到 DSTv35。

要同时升级引擎和时区文件，请使用 --option-group-name 和 --engine-version 选项运行 modify-db-instance。您可以立即运行此命令，也可以将其推迟到维护时段。在 In --option-group-name 中指定包含 TIMEZONE_FILE_AUTOUPGRADE 选项的选项组。例如：

```
aws rds modify-db-instance
    --db-instance-identifier my-instance \
    --engine-version new-version \
    ----option-group-name og-with-timezone-file-autoupgrade \
    --apply-immediately
```

RDS 开始将引擎升级到 19.0.0.0.ru-2022-10.rur-2022-10.r1。应用 TIMEZONE_FILE_AUTOUPGRADE 选项后，RDS 会检查是否有新的 DST 版本，看到 DSTv35 在 19.0.0.0.ru-2022-10.rur-2022-10.r1 中可用，并立即开始更新到 DSTv35。

要立即升级您的引擎，然后升级您的时区文件，请按顺序执行操作：

1. 仅使用以下 CLI 命令升级数据库引擎：

```
aws rds modify-db-instance \
    --db-instance-identifier my-instance \
    --engine-version new-version \
    --apply-immediately
```

2. 使用以下 CLI 命令将 TIMEZONE_FILE_AUTOUPGRADE 选项添加到附加到您的实例的选项组：

```
aws rds add-option-to-option-group \
```

```
    --option-group-name og-in-use-by-your-instance \
    --options OptionName=TIMEZONE_FILE_AUTOUPGRADE \
    --apply-immediately
```

升级数据库引擎版本而不更新时区文件

在这种情况下，您的数据库使用 DSTv33，但 DSTv34 在您的数据库实例文件系统上可用。您希望将数据库引擎升级到版本 19.0.0.0.ru-2022-10.rur-2022-10.r1（其中包含 DSTv35），但保留时区文件 DSTv33。您可能出于以下原因来选择该策略：

- 您的数据不使用 TIMESTAMP WITH TIME ZONE 数据类型。
- 您的数据使用 TIMESTAMP WITH TIME ZONE 数据类型，但您的数据不受时区更改的影响。
- 您想推迟更新时区文件，因为您无法容忍额外的停机时间。

您的策略取决于以下哪些可能性是确实存在的：

- 您的数据库实例未与包含 TIMEZONE_FILE_AUTOUPGRADE 的选项组关联。在 modify-db-instance 命令中，不要指定新的选项组，这样 RDS 就不会更新您的时区文件。
- 您的数据库实例目前与一个包含 TIMEZONE_FILE_AUTOUPGRADE 的选项组关联。在单个 modify-db-instance 命令中，将您的数据库实例与不包含 TIMEZONE_FILE_AUTOUPGRADE 的选项组关联，并将您的数据库引擎升级到 19.0.0.0.ru-2022-10.rur-2022-10.r1。

时区文件更新期间的停机时间

当 RDS 更新时区文件时，使用 TIMESTAMP WITH TIME ZONE 的现有数据可能会发生变化。在这种情况下，您首要的考虑因素是停机时间。

> **Warning**
>
> 如果您添加 TIMEZONE_FILE_AUTOUPGRADE 选项，则您引擎升级的停机时间可能要延长。更新大型数据库的时区数据可能需要数小时甚至数天。

时区文件更新的长度取决于如下因素：

- 数据库中的 TIMESTAMP WITH TIME ZONE 数据量
- 数据库实例配置

- 数据库实例类
- 存储配置
- 数据库配置
- 数据库参数设置

执行以下操作时，可能会产生额外的停机时间：

- 当数据库实例使用过时的时区文件时，将此选项添加到选项组
- 当新引擎版本包含时区文件的新版本时，升级 Oracle 数据库引擎

> **Note**
> 在时区文件更新期间，RDS for Oracle 调用 `PURGE DBA_RECYCLEBIN`。

准备更新时区文件

时区文件升级有两个不同的阶段：准备和升级。虽然准备步骤并非必需，但强烈建议您执行此步骤。在此步骤中，您将了解哪些数据将受到运行 PL/SQL 过程 `DBMS_DST.FIND_AFFECTED_TABLES` 的影响。有关准备窗口的更多信息，请参阅 Oracle 数据库文档中的使用时区数据升级时区文件和时间戳。

准备更新时区文件

1. 使用 SQL 客户端连接到您的 Oracle 数据库。
2. 确定当前使用的时区文件版本。

   ```
   SELECT * FROM V$TIMEZONE_FILE;
   ```

3. 确定数据库实例上可用的最新时区文件版本。

   ```
   SELECT DBMS_DST.GET_LATEST_TIMEZONE_VERSION FROM DUAL;
   ```

4. 确定具有类型为 `TIMESTAMP WITH LOCAL TIME ZONE` 或 `TIMESTAMP WITH TIME ZONE` 的列的表总大小。

   ```
   SELECT SUM(BYTES)/1024/1024/1024 "Total_size_w_TSTZ_columns_GB"
   FROM   DBA_SEGMENTS
   WHERE  SEGMENT_TYPE LIKE 'TABLE%'
   ```

```
AND    (OWNER, SEGMENT_NAME) IN
         (SELECT OWNER, TABLE_NAME
          FROM   DBA_TAB_COLUMNS
          WHERE  DATA_TYPE LIKE 'TIMESTAMP%TIME ZONE');
```

5. 确定具有类型为 TIMESTAMP WITH LOCAL TIME ZONE 或 TIMESTAMP WITH TIME ZONE 的列的段的名称和大小。

```
SELECT OWNER, SEGMENT_NAME, SUM(BYTES)/1024/1024/1024
 "SEGMENT_SIZE_W_TSTZ_COLUMNS_GB"
FROM   DBA_SEGMENTS
WHERE  SEGMENT_TYPE LIKE 'TABLE%'
AND    (OWNER, SEGMENT_NAME) IN
         (SELECT OWNER, TABLE_NAME
          FROM   DBA_TAB_COLUMNS
          WHERE  DATA_TYPE LIKE 'TIMESTAMP%TIME ZONE')
GROUP BY OWNER, SEGMENT_NAME;
```

6. 运行准备步骤。

   - 过程 DBMS_DST.CREATE_AFFECTED_TABLE 可创建表来存储任何受影响的数据。将此表的名称传递给 DBMS_DST.FIND_AFFECTED_TABLES 过程。有关更多信息，请参阅 Oracle 数据库文档中的 CREATE_AFFECTED_TABLE 过程。
   - 此过程 CREATE_ERROR_TABLE 创建用于记录错误的表。有关更多信息，请参阅 Oracle 数据库文档中的 CREATE_ERROR_TABLE 过程。

   以下示例创建受影响的数据和错误表，并查找所有受影响的表。

```
EXEC DBMS_DST.CREATE_ERROR_TABLE('my_error_table')
EXEC DBMS_DST.CREATE_AFFECTED_TABLE('my_affected_table')

EXEC DBMS_DST.BEGIN_PREPARE(new_version);
EXEC DBMS_DST.FIND_AFFECTED_TABLES('my_affected_table', TRUE, 'my_error_table');
EXEC DBMS_DST.END_PREPARE;

SELECT * FROM my_affected_table;
SELECT * FROM my_error_table;
```

7. 查询受影响表和错误表。

```
SELECT * FROM my_affected_table;
```

```
SELECT * FROM my_error_table;
```

添加时区文件自动升级选项

当您向选项组添加选项时，该选项组将处于以下状态之一：

- 一个现有选项组当前已附加到至少一个数据库实例。添加该选项时，所有使用此选项组的数据库实例都会自动重新启动。这会导致短暂中断。
- 现有选项组未附加到任何数据库实例。您计划添加该选项，然后将现有选项组与现有数据库实例或新数据库实例相关联。
- 您创建一个新的选项组并添加该选项。您计划将新的选项组与现有数据库实例或新的数据库实例关联。

控制台

将时区文件自动升级选项添加到数据库实例

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择选项组。
3. 确定您想要使用的选项组。您可以创建新的选项组，或使用现有选项组。如果您想使用现有选项组，请跳到下一步。或者，通过以下设置创建自定义数据库选项组：

   a. 对于 Engine（引擎），选择适用于您的数据库实例的 Oracle Database 版本。
   b. 对于主引擎版本，请选择数据库实例的版本。

   有关更多信息，请参阅 创建选项组。
4. 选择要修改的选项组，然后选择 Add Option (添加选项)。
5. 在添加选项窗口中，执行以下操作：

   a. 选择 TIMEZONE_FILE_AUTOUPGRADE。
   b. 要在添加选项后在所有关联数据库实例上启用该选项，对于立即应用，请选择是。如果选择否（默认），则会在下一个维护时段为每个关联数据库实例启用此选项。
6. 根据需要设置完毕后，选择 Add Option (添加选项)。

Amazon CLI

以下示例使用 Amazon CLI add-option-to-option-group 命令将 TIMEZONE_FILE_AUTOUPGRADE 选项添加到名为 myoptiongroup 的选项组。

对于 Linux、macOS 或 Unix：

```
aws rds add-option-to-option-group \
    --option-group-name "myoptiongroup" \
    --options "OptionName=TIMEZONE_FILE_AUTOUPGRADE" \
    --apply-immediately
```

对于 Windows：

```
aws rds add-option-to-option-group ^
    --option-group-name "myoptiongroup" ^
    --options "OptionName=TIMEZONE_FILE_AUTOUPGRADE" ^
    --apply-immediately
```

更新时区文件后检查数据

建议您在更新时区文件后检查数据。在准备步骤过程中，RDS for Oracle 会自动创建以下表：

- rdsadmin.rds_dst_affected_tables – 列出包含受更新影响的数据的表
- rdsadmin.rds_dst_error_table – 列出更新过程中生成的错误

这些表独立于您在准备时段中创建的任何表。要查看更新的结果，请按如下方式查询表。

```
SELECT * FROM rdsadmin.rds_dst_affected_tables;
SELECT * FROM rdsadmin.rds_dst_error_table;
```

有关受影响数据和错误表的架构的更多信息，请参阅 Oracle 文档中的 FIND_AFFECTED_TABLES 过程。

Oracle 透明数据加密

Amazon RDS 支持 Oracle 透明数据加密 (TDE)，这是 Oracle 企业版中提供的 Oracle 高级安全选项中的功能。该功能在将数据写入到存储之前自动对数据进行加密，并在从存储读取数据时自动对数据进行解密。仅自带许可（BYOL）模式支持此选项。

TDE 在需要加密敏感数据以防第三方获取数据文件和备份的情况下非常有用。当您需要遵守安全相关法规时，TDE 也非常有用。

有关 Oracle 数据库中 TDE 的详细说明超出了本指南的范围。有关信息，请参阅以下 Oracle 数据库资源：

- Oracle 数据库文档中的 Introduction to Transparent Data Encryption
- Oracle 数据库文档中的 Oracle 高级安全性
- Oracle 高级安全透明数据加密最佳实践，这是 Oracle 白皮书

有关将 TDE 和 RDS for Oracle 结合使用的更多信息，请参阅以下博客：

- Oracle Database Encryption Options on Amazon RDS
- Migrate a cross-account TDE-enabled Amazon RDS for Oracle DB instance with reduced downtime using Amazon DMS

TDE 加密模式

Oracle 透明数据加密支持两种加密方式，即 TDE 表空间加密和 TDE 列加密。TDE 表空间加密用于加密整个应用程序表。TDE 列加密用于加密单个包含敏感数据的数据元素。此外，您还可以应用同时使用 TDE 表空间和列加密的混合加密解决方案。

> **Note**
>
> Amazon RDS 为数据库实例管理 Oracle Wallet 和 TDE 主密钥。您无需使用命令 `ALTER SYSTEM set encryption key` 设置加密密钥。

启用 TDE 选项之后，可以使用以下命令检查 Oracle Wallet 的状态：

```
SELECT * FROM v$encryption_wallet;
```

要创建加密的表空间，请使用以下命令：

```
CREATE TABLESPACE encrypt_ts ENCRYPTION DEFAULT STORAGE (ENCRYPT);
```

要指定加密算法，请使用以下命令：

```
CREATE TABLESPACE encrypt_ts ENCRYPTION USING 'AES256' DEFAULT STORAGE (ENCRYPT);
```

前面用于加密表空间的语句与您在本地 Oracle 数据库上使用的语句相同。

TDE 选项的限制

TDE 选项是持久和永久的。在将您的数据库实例与启用了 TDE 选项的选项组关联后，您无法执行以下操作：

- 在当前关联的选项组中禁用 TDE 选项。
- 将您的数据库实例与不包含 TDE 的选项组相关联。
- 共享使用 TDE 选项的数据库快照。有关共享数据库快照的更多信息，请参阅 共享 Amazon RDS 的数据库快照。

有关持久和永久选项的更多信息，请参阅持久性和永久性选项。

确定您的数据库实例是否使用 TDE

您可能需要确定您的数据库实例是否与启用了 TDE 选项的选项组相关联。要查看与数据库实例关联的选项组，请使用 RDS 控制台、Amazon CLI 命令 describe-db-instance 或 API 操作 DescribeDBInstances。

添加 TDE 选项

要将 TDE 选项添加到数据库实例，请完成以下步骤：

1. （推荐）为您的数据库实例拍摄快照。
2. 请执行以下任务之一：
   - 从头开始创建一个新选项组。有关更多信息，请参阅 创建选项组。
   - 使用 Amazon CLI 或 API 复制现有选项组。有关更多信息，请参阅 复制选项组。
   - 重复使用现有的非默认选项组。最佳实践是使用当前未与任何数据库实例或快照关联的选项组。
3. 将新选项添加到您在前面的步骤中创建的选项组。

4. 如果当前与您的数据库实例关联的选项组已启用一些选项，请将这些选项添加到您的新选项组。此策略可以防止在启用新选项时卸载现有选项。
5. 将新选项添加到您的数据库实例。

控制台

将 TDE 选项添加到一个选项组并将该选项组与您的数据库实例关联

1. 在 RDS 控制台中，选择选项组。
2. 选择要将选项添加到的选项组的名称。
3. 选择添加选项。
4. 对于选项名称，选择 TDE，然后配置选项设置。
5. 选择添加选项。

   > **Important**
   > 如果在当前已附加到一个或多个数据库实例的选项组中添加 TDE 选项，则所有数据库实例自动重新启动时都会发生短暂中断。

   有关添加选项的更多信息，请参阅 将选项添加到选项组。
6. 将该选项组与新的或现有的数据库实例关联：
   - 对于新数据库实例，请在启动实例时应用选项组。有关更多信息，请参阅“创建 Amazon RDS 数据库实例”。
   - 对于现有数据库实例，请修改实例并附加新的选项组以应用选项组。数据库实例不会作为此操作的一部分重启。有关更多信息，请参阅 修改 Amazon RDS 数据库实例。

Amazon CLI

在以下示例中，您使用 Amazon CLI add-option-to-option-group 命令将 TDE 选项添加到名为 `myoptiongroup` 的选项组。有关更多信息，请参阅开始使用：Flink 1.13.2。

对于 Linux、macOS 或 Unix：

```
aws rds add-option-to-option-group \
    --option-group-name "myoptiongroup" \
```

```
    --options "OptionName=TDE" \
    --apply-immediately
```

对于 Windows：

```
aws rds add-option-to-option-group ^
    --option-group-name "myoptiongroup" ^
    --options "OptionName=TDE" ^
    --apply-immediately
```

将您的数据复制到不包含 TDE 选项的数据库实例

您无法从数据库实例中删除 TDE 选项或将其与不包括 TDE 选项的选项组关联。要将数据迁移到不包含 TDE 选项的实例，请执行以下操作：

1. 解密数据库实例上的数据。
2. 将数据复制到未与已启用 `TDE` 的选项组关联的新数据库实例。
3. 删除原始数据库实例。

您可以将上一个数据库实例的名称用于新实例。

将 TDE 与 Oracle Data Pump 结合使用时的注意事项

可以使用 Oracle Data Pump 导入或导出加密转储文件。Amazon RDS 支持 Oracle Data Pump 的密码加密模式 (`ENCRYPTION_MODE=PASSWORD`)。Amazon RDS 不支持 Oracle Data Pump 的透明加密模式 (`ENCRYPTION_MODE=TRANSPARENT`)。有关更多信息，请参阅 使用 Oracle Data Pump 导入。

Oracle UTL_MAIL

Amazon RDS 通过使用 UTL_MAIL 选项和 SMTP 服务器支持 Oracle UTL_MAIL。您可以通过使用 UTL_MAIL 包直接从您的数据库发送电子邮件。Amazon RDS 支持适用于以下版本 Oracle 的 UTL_MAIL：

- Oracle Database 21c (21.0.0.0)，所有版本
- Oracle Database 19c (19.0.0.0)，所有版本

以下是使用 UTL_MAIL 的一些限制：

- UTL_MAIL 不支持传输层安全性 (TLS)，因此，不会对电子邮件进行加密。

  要通过创建和上传自定义 Oracle wallet 来安全地连接到远程 SSL/TLS 资源，请按照 使用证书和 Oracle Wallet 配置 UTL_HTTP 访问 中的说明操作。

  Wallet 所需的具体证书因服务而异。对于 Amazon 服务，通常可在 Amazon Trust Services 存储库中了解到相关信息。
- UTL_MAIL 不支持对 SMTP 服务器进行身份验证。
- 您在一封电子邮件中只能发送一个附件。
- 您无法发送 32 K 以上的附件。
- 您只能使用 ASCII 和广义二进制编码的十进制交换码 (EBCDIC) 字符编码。
- SMTP 端口 (25) 根据弹性网络接口所有者的策略进行限制。

当您启用 UTL_MAIL 时，仅向数据库实例的主用户授予执行权限。如果需要，主用户可向其他用户授予执行权限，以便他们能够使用 UTL_MAIL。

> **Important**
>
> 建议您启用 Oracle 的内置审核功能以跟踪 UTL_MAIL 过程的使用情况。

Oracle UTL_MAIL 的先决条件

以下是使用 Oracle UTL_MAIL 的先决条件：

- 一个或多个 SMTP 服务器，及对应的 IP 地址或公有或私有域名服务器 (DNS) 名称。有关通过自定义 DNS 服务器解析的私有 DNS 名称的详细信息，请参阅 设置自定义 DNS 服务器。

添加“Oracle UTL_MAIL”选项

将“Oracle UTL_MAIL”选项添加到数据库实例的一般过程如下所示：

1. 创建新的选项组，或者复制或修改现有选项组。
2. 将 选项添加到该选项组。
3. 将选项组与数据库实例相关联。

在添加“UTL_MAIL”选项后，一旦激活该选项组，就会激活 UTL_MAIL。

将“UTL_MAIL”选项添加到数据库实例

1. 确定您想要使用的选项组。您可以创建新的选项组，或使用现有选项组。如果您想使用现有选项组，请跳到下一步。或者，通过以下设置创建自定义数据库选项组：

   a. 对于引擎，请选择要使用的 Oracle 版本。
   b. 对于主引擎版本，选择数据库实例的版本。

   有关更多信息，请参阅“创建选项组”。
2. 将 UTL_MAIL 选项添加到该选项组。有关添加选项的更多信息，请参阅 将选项添加到选项组。
3. 将选项组应用到新的或现有的数据库实例：

   - 对于新数据库实例，您可以在启动实例时应用选项组。有关更多信息，请参阅“创建 Amazon RDS 数据库实例”。
   - 对于现有数据库实例，您可以通过修改实例并附加新的选项组来应用选项组。有关更多信息，请参阅“修改 Amazon RDS 数据库实例”。

使用 Oracle UTL_MAIL

在启用“UTL_MAIL”选项后，您必须先配置 SMTP 服务器，然后才能开始使用它。

通过将 SMTP_OUT_SERVER 参数设置为有效的 IP 地址或公有 DNS 名称来配置 SMTP 服务器。对于 SMTP_OUT_SERVER 参数，您可以指定多个服务器的地址的逗号分隔的列表。如果第一个服务器不可用，则 UTL_MAIL 会尝试下一个服务器，依此类推。

您可以通过使用数据库参数组设置数据库实例的默认 SMTP_OUT_SERVER。您可以通过在数据库实例的数据库上运行以下代码来设置会话的 SMTP_OUT_SERVER 参数。

```
ALTER SESSION SET smtp_out_server = mailserver.domain.com:25;
```

在启用“UTL_MAIL”选项并配置 SMTP_OUT_SERVER 后，您可以通过使用 SEND 过程来发送邮件。有关更多信息，请参阅 Oracle 文档中的 UTL_MAIL。

删除“Oracle UTL_MAIL”选项

您可以从数据库实例中删除“Oracle UTL_MAIL”。

要从数据库实例中删除“UTL_MAIL”，请执行下列操作之一：

- 要从多个数据库实例中删除“UTL_MAIL”，请从其所属的选项组中删除“UTL_MAIL”选项。此更改会影响使用该选项组的所有数据库实例。有关更多信息，请参阅“从选项组中删除选项”。
- 要从单个数据库实例中删除“UTL_MAIL”选项，请修改该数据库实例，并指定另一个不包含“UTL_MAIL”选项的选项组。您可以指定默认 (空) 选项组，或指定其他自定义选项组。有关更多信息，请参阅“修改 Amazon RDS 数据库实例”。

故障排除

以下是您在将 UTL_MAIL 用于 Amazon RDS 时可能遇到的问题。

- 限制。SMTP 端口 (25) 根据弹性网络接口所有者的策略进行限制。如果您可以使用 UTL_MAIL 成功发送电子邮件，并且看到错误 `ORA-29278: SMTP transient error: 421 Service not available`，则您可能被限制。如果您在发送电子邮件时遇到限制，我们建议您实施回退算法。有关回退算法的更多信息，请参阅 Amazon中的错误重试和指数回退以及如何处理“throttling – Maximum sending rate exceeded”错误。

  您可以请求删除此限制。有关更多信息，请参阅如何从 EC2 实例删除端口 25 上的限制？。

Oracle XML DB

Oracle XML DB 向数据库实例添加本机 XML 支持。利用 XML DB，可以存储和检索结构化或非结构化 XML 以及关系数据。RDS for Oracle 不支持 XML DB 协议服务器。

XML DB 预安装在 Oracle Database 12c 及更高版本上。因此，您无需使用选项组来显式安装 XML DB 作为附加特征。

要了解如何配置和使用 XML DB，请参阅 Oracle 数据库文档中的 Oracle XML DB Developer's Guide。

升级 RDS for Oracle 数据库引擎

如果 Amazon RDS 支持新版本的 Oracle 数据库，您可以将数据库实例升级到新版本。有关 Amazon RDS 上可用的 Oracle 版本的信息，请参阅 Amazon RDS for Oracle 发布说明。

> **Important**
>
> 不再支持 RDS for Oracle Databases 11g、12c 和 18c。如果您维护 Oracle Database 11g、12c 或 18c 快照，可以将其升级到更高版本。有关更多信息，请参阅 升级 Oracle 数据库快照。

主题

RDS for Oracle 引擎升级概述

升级 RDS for Oracle 数据库实例之前，请自行熟悉以下概念。

主题

主要和次要版本升级

主要版本是 Oracle 数据库的主要版本，每 1-2 年发布一次。Oracle Database 19c 和 Oracle Database 21c 是主要版本。

每个季度，RDS for Oracle 都会为每个受支持的主要引擎发布新的次要引擎版本。版本更新（RU）引擎版本通过包含指定季度的 RU 补丁，纳入了 Oracle 提供的错误修复。例如，21.0.0.0.ru-2024-10.rur-2024-10.r1 是 Oracle Database 21c 的次要版本，其中包含 2024 年 10 月的 RU。

Spatial 补丁包（SPB）引擎版本包含 RU 补丁和特定于 Oracle Spatial 的补丁。例如，19.0.0.0.ru-2025-01.spb-1.r1 是一个次要引擎版本，其中包含引擎版本 19.0.0.0.ru-2025-01.rur-2025-01.r1 中的 RU 补丁以及 Spatial 补丁。通常，RDS for Oracle 会在相应 RU 发布后 2-3 周内发布 SPB。有关 RU 和 SPB 之间差异的解释，请参阅版本更新（RU）和 Spatial 补丁包（SPB）。有关支持的 RU 和 SPB 的信息，请参阅 Release notes for Amazon Relational Database Service (Amazon RDS) for Oracle。

RDS for Oracle 支持对数据库实例进行以下升级。

升级类型	应用程序兼容性	升级方法	示例升级途径
主要版本	主版本升级可能引入与现有应用程序不兼容的更改。	仅限手动	从 Oracle Database 19c 到 Oracle Database 21c
次要版本	次要版本升级 仅包含与现有应用程序向后兼容的更改。	自动或手动	从 21.0.0.0.ru-2023-07.rur-2022-07.r1 到 21.0.0.0.ru-2023-10.rur-2022-10.r1

> **Important**
>
> 当您更新数据库引擎时，发生中断。中断时长取决于引擎版本和数据库实例大小。在将升级应用到生产数据库之前，确保彻底地测试任何升级，验证您的应用程序是否正常工作。有关更多信息，请参阅 测试 Oracle 数据库升级。

RDS for Oracle 的支持日期和强制升级

RDS for Oracle 的数据库版本具有预期的支持日期。当 RDS for Oracle 数据库引擎的主要或次要版本接近其支持终止日期时，RDS 将开始强制升级，也成为被迫升级。RDS 发布以下信息：

- 建议您开始手动将已弃用版本上的实例升级到支持的版本
- 一个日期，在此日期之后，您将无法再在不受支持的版本上创建实例
- RDS 开始在维护时段内自动将您的实例升级到受支持版本的日期
- RDS 开始在维护时段之外自动将您的实例升级到受支持版本的日期

> **Important**
>
> 强制升级可能会对 Amazon CloudFormation 堆栈产生意外的后果。如果您依靠 RDS 来自动升级数据库实例，则可能会遇到与 Amazon CloudFormation 有关的问题。

本节包含以下主题：

主题

RDS for Oracle 主要版本的支持日期

RDS for Oracle 主要版本至少会在相应的 Oracle Database 发行版的终止支持日期之前保持可用状态。您可以参照下列日期规划您的测试和升级周期。这些日期表示可能需要升级到较新版本的最早日期。如果 Amazon 对某个 RDS for Oracle 版本的支持时间比原定时间更长，我们会计划更新此表格以反映较晚的日期。

> **Note**
>
> 您还可以运行 describe-db-major-engine-versions Amazon CLI 命令或者使用 DescribeDBMajorEngineVersions RDS API 操作，来查看 Oracle 数据库的主要版本。

Oracle Database 主要发行版	升级到较新版本的预计日期
Oracle Database 19c	2029 年 12 月 31 日，提供 BYOL 卓越支持（扩展支持免收费用） 2032 年 12 月 31 日，提供 BYOL 扩展支持（额外费用）或无限许可协议 2029 年 12 月 31 日，附带许可（LI）
Oracle Database 21c	2027 年 7 月 31 日（不适用于扩展支持）

RDS 会在您需要升级到较新的主要版本前至少 12 个月通知您。通知介绍升级过程，包括重要里程碑的时间安排、对数据库实例的影响以及建议的操作。我们建议您在将数据库升级到主要版本之前，使用新的 RDS for Oracle 版本彻底测试您的应用程序。

在该预先通知期限过后，对后续主要版本的自动升级可能会应用于任何仍在运行较久版本的 RDS for Oracle 数据库实例。如果此类情况发生，自动升级将在设定的维护时段内启动。

有关更多信息，请参阅 My Oracle Support 中的 Release Schedule of Current Database Releases。

RDS for Oracle 次要版本的支持日期

Oracle Database 19c 的以下次要版本正接近终止支持日期：

- 19.0.0.0.ru-2019-07.rur-2019-07.r1
- 19.0.0.0.ru-2019-10.rur-2019-10.r1
- 19.0.0.0.ru-2020-01.rur-2020-01.r1

Oracle Database 21c 的以下次要版本正接近终止支持日期：

- 21.0.0.0.ru-2022-01.rur-2022-01.r1
- 21.0.0.0.ru-2022-04.rur-2022-04.r1
- 21.0.0.0.ru-2022-07.rur-2022-07.r1
- 21.0.0.0.ru-2022-10.rur-2022-10.r1
- 21.0.0.0.ru-2023-01.rur-2023-01.r1
- 21.0.0.0.ru-2023-01.rur-2023-01.r2

下表显示了先前已弃用版本的强制次要版本升级时间表。

日期	针对已弃用版本的操作
2025/07/02	您无法再创建运行已弃用版本的数据库实例。要么手动将数据库实例升级到最新版本更新（RU），要么等待 RDS 自动升级这些实例。
08/01/25 – 09/15/25	RDS 在计划的维护时段强制将数据库实例升级到最新的 RU。RDS 还会升级从已弃用版本的快照还原的所有实例。
2025/09/15	RDS 开始强制将数据库实例升级到最新的 RU，而不考虑计划的维护时段。强制升级将继续进行，直到所有实例都运行受支持的次要版本。

如果您不希望 RDS 自动升级正在运行已弃用版本的实例，请执行以下操作：

- 手动将 Oracle Database 19c 和 21c 次要版本升级到最新的 RU 或未计划进行强制升级的任何 RU。例如，您可以将 21.0.0.0.ru-2023-01.rur-2023-01.r2 升级到 21.0.0.0.ru-2023-04.rur-2023-04.r2，或将 19.0.0.0.ru-2019-07.rur-2019-07.r1 升级到 19.0.0.0.ru-2025-01.rur-2025-01.r2。
- 升级使用已弃用次要版本的数据库快照，然后还原快照。

有关次要版本的更多信息，请参阅 Release notes for Amazon Relational Database Service (Amazon RDS) for Oracle。

Oracle 引擎版本管理

利用数据库引擎版本管理，您可以控制修补和升级数据库引擎的时间和方式。您可以灵活地保持与数据库引擎补丁版本的兼容性。您还可以测试 RDS for Oracle 的新补丁版本，以确保在将这些补丁部署到生产中之前，它们能够与您的应用程序配合工作。此外，您可以根据自己的条款和时间表升级版本。

> **Note**
>
> Amazon RDS 使用 Amazon RDS 特定的数据库引擎版本定期汇总官方 Oracle 数据库补丁。要查看某个 Amazon RDS Oracle 特定引擎版本中包含的 Oracle 补丁的列表，请转到 Amazon RDS for Oracle 发布说明。

在引擎升级期间自动拍摄快照

升级 Oracle 数据库实例时，快照可防止出现升级问题。如果数据库实例的备份保留期大于 0，Amazon RDS 在升级过程中拍摄以下数据库快照：

1. 在进行任何升级更改之前数据库实例的快照。如果升级失败，您可以还原此快照，以创建运行旧版本的数据库实例。
2. 升级完成后的数据库实例快照。

> **Note**
> 要更改您的备份保留期，请参阅 修改 Amazon RDS 数据库实例。

升级后，您无法恢复到以前的引擎版本。但是，您可以通过恢复升级前的快照来创建新的 Oracle 数据库实例。

多可用区部署中的 Oracle 升级

如果数据库实例在多可用区部署中，Amazon RDS 会同时升级主要副本和备用副本。如果不需要操作系统更新，则主升级和备用升级同时进行。在升级完成之前，这些实例不可用。

如果多可用区部署中需要操作系统更新，Amazon RDS 会在您请求数据库升级时应用更新。Amazon RDS 执行以下步骤：

1. 更新当前备用数据库实例上的操作系统。
2. 将主数据库实例失效转移到备用数据库实例。
3. 升级以前是备用实例的新主数据库实例上的数据库版本。主数据库在升级期间不可用。
4. 更新以前是主数据库实例的新备用数据库实例上的操作系统。
5. 升级新的备用数据库实例上的数据库版本。
6. 将新的主数据库实例失效转移回原始主数据库实例，并且将新的备用数据库实例失效转移回原始备用数据库实例。因此，Amazon RDS 会将复制配置返回到其原始状态。

只读副本的 Oracle 升级

源数据库实例及其所有只读副本的 Oracle 数据库引擎版本必须相同。Amazon RDS 在以下阶段执行升级：

1. 升级源数据库实例。只读副本在此阶段可用。
2. 并行升级只读副本，而不考虑副本维护时段。源数据库在此阶段可用。

对于跨区域只读副本的主要版本升级，Amazon RDS 执行其他操作：

- 自动为目标版本生成选项组
- 将所有选项和选项设置从原始选项组复制到新选项组
- 将升级的跨区域只读副本与新选项组关联

Oracle 主要版本升级

要执行主要版本升级，请手动修改数据库实例。主要版本升级不会自动进行。

> **Important**
>
> 在将升级应用到生产数据库之前，确保彻底测试任何升级，验证您的应用程序是否正常工作。有关更多信息，请参阅 测试 Oracle 数据库升级。

主题

- 主要版本升级支持的版本
- 主要版本升级支持的实例类
- 在主要版本升级之前收集统计信息
- 允许主要版本升级

主要版本升级支持的版本

Amazon RDS 支持以下主要版本升级。

当前版本	支持升级
使用 CDB 架构的 19.0.0.0	21.0.0.0

Oracle 数据库的主要版本升级必须升级到在同一个月或更晚发布的版本更新 (RU)。任何 Oracle Database 版本都不支持主要版本降级。

主要版本升级支持的实例类

您当前的 Oracle 数据库实例可能在您要升级到的版本不支持的数据库实例类上运行。在这种情况下，在升级之前，请将数据库实例迁移到受支持的数据库实例类。有关 Amazon RDS for Oracle 的每个版本支持的数据库实例类的更多信息，请参阅 数据库实例类。

在主要版本升级之前收集统计信息

在执行主要版本升级前，Oracle 建议您对您要升级的数据库实例收集优化器统计数据。此操作可以减少升级过程中的数据库实例停机时间。

要收集优化器统计数据，请以主用户身份连接到数据库实例，然后运行以下 `DBMS_STATS.GATHER_DICTIONARY_STATS` 过程，如以下示例所示。

```
EXEC DBMS_STATS.GATHER_DICTIONARY_STATS;
```

有关更多信息，请参阅 Oracle 文档中的 GATHER_DICTIONARY_STATS 过程。

允许主要版本升级

主要引擎版本升级可能与您的应用程序不兼容。升级是不可逆的。如果为 EngineVersion 参数指定的主要版本与当前主要版本不同，则必须允许主要版本升级。

如果使用 CLI 命令 modify-db-instance 升级主要版本，请指定 `--allow-major-version-upgrade`。由于此设置不具备持久性，因此只要执行主要升级，就必须指定 `--allow-major-version-upgrade`。此参数对次要引擎版本的升级没有影响。有关更多信息，请参阅“升级数据库实例引擎版本”。

如果您使用控制台升级主要版本，则无需选择允许升级的选项。相反，控制台会显示一条警告，指出主要升级是不可逆的。

Oracle 次要版本升级

在 RDS for Oracle 中，次要版本升级是对主要数据库引擎版本的更新。在 RDS 中，次要引擎版本可以是版本更新（RU）或 Spatial 补丁包（SPB）。例如，如果数据库实例运行的是主要版本 Oracle Database 21c 和次要版本 21.0.0.0.ru-2024-10.rur-2024-10.r1，则可以将数据库引擎升级到次要版本 21.0.0.0.ru-2025-01.rur-2025-01.r1。RDS for Oracle 不支持次要版本降级。

您可以手动或自动将数据库引擎升级到次要版本。要了解如何手动升级，请参阅手动升级引擎版本。要了解如何配置自动升级，请参阅自动升级次要引擎版本。无论是手动升级还是自动升级，次要版本升级都需要停机。在计划升级时，请考虑此停机。

> **Important**
>
> 在将升级应用到生产数据库之前，确保彻底地测试任何升级，验证您的应用程序是否正常工作。有关更多信息，请参阅 测试 Oracle 数据库升级。

主题

版本更新（RU）和 Spatial 补丁包（SPB）

在 RDS 中，版本更新（RU）是每季度发布的次要引擎版本，其中包括 Oracle Database 的安全修复、错误修复和新功能。Spatial 补丁包（SPB）是 RU 引擎版本，其中包含专为 Oracle Spatial 选项设计的补丁。例如，名为 19.0.0.0.ru-2025-01.spb-1.r1 的 SPB 包括相应 RU 19.0.0.0.ru-2025-01.rur-2025-01.r1 中的所有补丁，以及特定于 Spatial 的补丁。只有 Oracle Database 19c 才支持 SPB。

当实例配置为自动升级次要版本时，RU 和 SPB 位于单独的升级途径上。通常，SPB 在其相应的 RU 发布后的 2-3 周内发布。下表显示 Oracle Database 19c 的示例次要版本。

标准 RU 升级途径	SPB 升级途径
19.0.0.0.ru-2025-01.rur-2025-01.r1	19.0.0.0.ru-2025-01.spb-1.r1
19.0.0.0.ru-2025-04.rur-2025-04.r1	19.0.0.0.ru-2025-04.spb-1.r1
19.0.0.0.ru-2025-07.rur-2025-07.r1	19.0.0.0.ru-2025-07.spb-1.r1
19.0.0.0.ru-2025-10.rur-2025-10.r1	19.0.0.0.ru-2025-10.spb-1.r1

如果数据库实例配置为自动升级，则实例处于与当前版本对应的升级途径上。
例如，如果数据库实例运行版本 19.0.0.0.ru-2025-01.rur-2025-01.r1，则当发布

19.0.0.0.ru-2025-04.rur-2025-04.r1 时，实例会自动升级到此 RU。同样，如果数据库实例运行 19.0.0.0.ru-2025-01.spb-1.r1，那么当发布 19.0.0.0.ru-2025-04.spb-1.r1 时，实例会自动升级到此 SPB。运行 19.0.0.0.ru-2025-01.rur-2025-01.r1（为 RU）的实例不会自动升级到 19.0.0.0.ru-2025-04.spb-1.r1（这是位于单独升级途径上的 SPB）。

即使实例不使用 Spatial，也可以将数据库实例升级到 SPB，但是 Spatial 补丁仅适用于 Oracle Spatial。可以手动从 RU 升级到相同或更高引擎版本的 SPB。例如，可以将实例从 19.0.0.0.ru-2025-01.rur-2025-01.r1 升级到以下任一引擎版本：

- 19.0.0.0.ru-2025-01.spb-1.r1
- 19.0.0.0.ru-2025-04.spb-1.r1

只有当 RU 是更高的引擎版本时，您才能将实例从 SPB 升级到 RU。例如，可以从 SPB 版本 19.0.0.0.ru-2025-04.spb-1.r1 升级到更高的 RU 版本 19.0.0.0.ru-2025-07.rur-2025-07.r1，但不能升级到相同的 RU 版本 19.0.0.0.ru-2025-04.rur-2025-04.r1。

如果数据库实例配置为自动次要版本升级，并且您手动从 RU 升级到 SPB 或从 SPB 升级到 RU，则自动升级途径会发生变化。假设您手动从 RU 版本 19.0.0.0.ru-2025-01.rur-2025-01.r1 版本升级到 SPB 版本 19.0.0.0.ru-2025-01.spb-1.r1。您的下一个自动次要版本升级将是升级到 SPB 版本 19.0.0.0.ru-2025-04.spb-1.r1。

由于 SPB 和 RU 功能相同，因此，用于将实例升级到 RU 和 SPB 的 RDS API 是相同的。以下命令演示如何升级到 RU 和 SPB。

```
aws rds modify-db-instance \
    --db-instance-identifier mydbinstance \
    --engine-version 19.0.0.0.ru-2025-01.rur-2025-01.r1

aws rds modify-db-instance \
    --db-instance-identifier mydbinstance \
    --engine-version 19.0.0.0.ru-2025-01.spb-1.r1
```

有关 Oracle Spatial 选项的更多信息，请参阅 Spatial 补丁包（SPB）的工作原理。有关 Oracle Database 19c 支持的 RU 和 SPB，请参阅 Amazon RDS for Oracle Database 19c (19.0.0.0)。

开启 Oracle 的自动次要版本升级

在自动次要版本升级过程中，RDS 无需手动干预，即可将最新的可用次要版本应用到您的 Oracle 数据库。在以下情况下，Amazon RDS for Oracle 数据库实例会计划在下一个维护时段期间升级：

- 您的数据库实例开启了自动次要版本升级选项。
- 您的数据库实例尚未运行最新的次要数据库引擎版本。
- 您的数据库实例尚未计划待定的升级。

要了解如何开启自动升级，请参阅自动升级次要引擎版本。

RDS for Oracle 中的自动次要版本升级通知

RDS 在开始计划自动升级之前会提前发布通知。您可以在数据库详细信息页面的维护和备份选项卡中找到通知。消息具有以下格式：

```
An automatic minor version upgrade to engine version will become available
 on availability-date and will be applied during a subsequent maintenance window.
```

提前通知中的 *`availability-date`* 是 RDS 开始为您的 Amazon Web Services 区域中的数据库实例计划升级的日期。这不是数据库实例升级计划发生的日期。例如，如果 *`availability-date`* 是 3 月 1 日，则 RDS 可能会将您的升级计划在 4 月 14 日。

您也可以使用 Amazon CLI 中的 `describe-pending-maintenance-actions` 命令获取升级可用日期，如以下示例所示：

```
aws rds describe-pending-maintenance-actions

{
    "PendingMaintenanceActions": [
        {
            "ResourceIdentifier": "arn:aws:rds:us-east-1:123456789012:db:orclinst1",
            "PendingMaintenanceActionDetails": [
                {
                    "Action": "db-upgrade",
                    "Description": "Automatic minor version upgrade to
 21.0.0.0.ru-2024-07.rur-2024-07.r1",
                    "CurrentApplyDate": "2024-12-02T08:10:00Z",
                    "OptInStatus": "next-maintenance"
                }
            ]
        }, ...
```

下表介绍了每种类型的待处理维护操作消息的选项。

待处理维护操作消息	当显示消息时	是否有资格在下一个维护时段应用？	是否有资格立即应用？	是否有资格撤销选择加入？
自动次要版本升级到 *engine-version* 将在 *availability-date* 提供，并应在随后的维护时段内应用。	计划自动升级前 4-6 周。	支持	是	是
自动次要版本升级到 *engine-version*	在 *availability-date* 当天或之后。RDS 会在数据库实例的下一个维护时段自动应用此升级。	支持	是	否

有关 describe-pending-maintenance-actions 的更多信息，请参阅《Amazon CLI 命令参考》。

当 RDS 在 RDS for Oracle 中计划自动次要版本升级时

当自动升级的可用日期到来时，RDS 开始计划升级。对于大多数 Amazon Web Services 区域，RDS 计划在推出最新的每季度 RU 的大概四到六周后升级到此 RU。预定日期因 Amazon Web Services 区域 和其他因素而异。有关 RU 和 RUR 的更多信息，请参阅 Amazon RDS for Oracle 版本注释。

当 RDS 计划升级时，数据库详细信息页面的维护和备份选项卡中会显示以下通知：

```
Automatic minor version upgrade to engine-version
```

前面的消息表明 RDS 已计划在下一个维护时段升级您的数据库引擎。

有时，在 RDS 应用先前的次要版本之前，就会有新的次要版本可用。例如，您的实例运行的是 *minor-version-1*，而 *minor-version-2* 和 *minor-version-3* 均可作为升级目标。在这种情况下，为了避免数据库实例出现不必要的停机时间，RDS 会计划自动将次要版本升级到最新版本，跳过对先前版本的升级。在此示例中，RDS 将您的实例从 *minor-version-1* 直接升级到 *minor-version-3*。

为确保按特定频率升级次要版本，您可以手动升级您的实例，而不是使用自动升级机制。要将升级安排在下一个维护时段，请在使用 `modify-db-instance` 升级到次要版本时指定 `--no-apply-immediately`。要立即升级，请改为指定 `--apply-immediately`。有关更多信息，请参阅 手动升级引擎版本。

管理 RDS for Oracle 中的自动次要版本升级

如果新的次要版本推出，您可以手动将数据库实例升级到此版本。以下示例立即升级名为 `orclinst1` 的数据库实例：

```
aws rds apply-pending-maintenance-action \
    --resource-identifier arn:aws:rds:us-east-1:123456789012:db:orclinst1 \
    --apply-action db-upgrade \
    --opt-in-type immediate
```

要选择退出尚未计划的自动次要版本升级，请将 `--opt-in-type` 设置为 `undo-opt-in`，如下例所示：

```
aws rds apply-pending-maintenance-action \
    --resource-identifier arn:aws:rds:us-east-1:123456789012:db:orclinst1 \
    --apply-action db-upgrade \
    --opt-in-type undo-opt-in
```

如果 RDS 已经为您的数据库实例计划了升级，则无法使用 `apply-pending-maintenance-action` 来取消升级。但是，您可以修改数据库实例并关闭自动次要升级功能，这随后会取消升级计划。

要了解如何关闭自动次要版本升级，请参阅自动升级次要引擎版本。有关 apply-pending-maintenance-action 的更多信息，请参阅《Amazon CLI 命令参考》。

Oracle 数据库升级注意事项

在升级 Oracle 实例之前，请查看以下信息。

主题

- 时区注意事项
- Spatial 补丁包（SPB）注意事项

Oracle 多租户注意事项

下表介绍了不同版本中支持的 Oracle 数据库架构。

Oracle Database 版本	RDS 支持状态	架构
Oracle Database 21c	支持	仅 CDB
Oracle Database 19c	支持	CDB 或非 CDB

下表描述了支持和不支持的升级途径。

升级途径	是否支持？
CDB 升级至 CDB	是
非 CDB 升级至 CDB	否，但您可以将非 CDB 转换为 CDB，然后对其进行升级
CDB 升级至非 CDB	否

有关 RDS for Oracle 中 Oracle 多租户的更多信息，请参阅CDB 架构的单租户配置。

选项组注意事项

如果您的数据库实例使用自定义选项组，Amazon RDS 有时会无法自动分配新的选项组。例如，在升级到新的主要版本时，会发生这种情况。在这种情况下，请在升级时指定新的选项组。我们建议您创建新的选项组，然后向该组添加与现有自定义选项组相同的选项。

有关更多信息，请参阅 创建选项组 或 复制选项组。

如果数据库实例使用包含 `APEX` 和 `APEX-DEV` 选项的自定义选项组，您有时可以减少升级用时。为此，请在升级数据库实例的同时升级 Oracle APEX 版本。有关更多信息，请参阅 升级 Oracle APEX 版本。

参数组注意事项

如果数据库实例使用自定义参数组，则有时 Amazon RDS 无法自动向数据库实例分配新的参数组。例如，在升级到新的主要版本时，会发生这种情况。在这种情况下，请确保在升级时指定一个新的参数组。我们建议您创建新参数组，然后配置您现有的自定义参数组中的那些参数。

有关更多信息，请参阅 在 Amazon RDS 中创建数据库参数组 或 在 Amazon RDS 中复制数据库参数组。

时区注意事项

您可以使用时区选项更改您的 Oracle 数据库实例使用的系统时区。例如，您可能需要更改数据库实例的时区，从而与本地环境或旧版应用程序兼容。时区选项在主机级别更改时区。Amazon RDS for Oracle 全年自动更新系统时区。有关更改系统时区的更多信息，请参阅 Oracle 时区。

创建 Oracle 数据库实例时，数据库会自动设置数据库时区。数据库时区也称为夏令时 (DST) 时区。数据库时区不同于系统时区。

在 Oracle 数据库版本之间，补丁集或单个补丁可能包含新的 DST 版本。这些补丁反映不同时区区域的转换规则变化。例如，政府可能会更改 DST 生效的时间。对 DST 规则的更改可能会影响 `TIMESTAMP WITH TIME ZONE` 数据类型的现有数据。

如果升级 RDS for Oracle 数据库实例，Amazon RDS 不会自动升级数据库时区文件。要自动升级时区文件，您可以在引擎版本升级期间或之后与数据库实例关联的选项组中添加 `TIMEZONE_FILE_AUTOUPGRADE` 选项。有关更多信息，请参阅 Oracle 时区文件自动升级。

或者，要手动升级数据库时区文件，请创建具有所需 DST 补丁的新 Oracle 数据库实例。但是，我们建议您使用 `TIMEZONE_FILE_AUTOUPGRADE` 选项来升级数据库时区文件。

升级时区文件后，将数据从您的当前实例迁移到新实例。您可以使用多种技术迁移数据，包括：

- Amazon Database Migration Service
- Oracle GoldenGate
- Oracle Data Pump
- 原始导出/导入（不再支持一般用途）

> **Note**
>
> 使用 Oracle Data Pump 迁移数据时，当目标时区版本低于源时区版本时，此实用程序会引发错误 ORA-39405。

有关更多信息，请参阅 Oracle 文档中的 TIMESTAMP WITH TIMEZONE 限制。

Spatial 补丁包（SPB）注意事项

在 RDS for Oracle 中，版本更新（RU）是次要引擎版本，其中包括 Oracle Database 的安全修复、错误修复和新功能。Spatial 补丁包（SPB）是次要引擎版本，其中也包含专为 Oracle Spatial 选项设计的补丁。例如，19.0.0.0.ru-2025-01.spb-1.r1 是一个次要引擎版本，其中包含引擎版本 19.0.0.0.ru-2025-01.rur-2025-01.r1 中的 RU 补丁以及 Spatial 补丁。

将数据库升级到 SPB 时，请考虑以下事项：

- 只有 Oracle Database 19c 才支持 SPB。
- 通常，SPB 在其相应的季度 RU 发布后的 2-3 周内发布。
- 即使数据库实例不使用 Oracle Spatial 选项，也可以将该实例升级到 SPB，但引擎版本中的 Spatial 补丁仅适用于 Oracle Spatial。您可以在 SPB 上创建新实例，稍后再安装 Oracle Spatial 选项。
- 如果您为数据库实例启用自动次要版本升级，则升级途径取决于您的实例当前使用的是 SPB 还是 RU。如果您的实例使用 SPB，则 RDS 会自动将您的实例升级到最新的 SPB。如果您的实例使用 RU，则 RDS 会自动将您的实例升级到最新的 RU。
- 只有当 SPB 的引擎版本与当前 RU 相同或更高时，才能手动将数据库实例从 RU 升级到 SPB。
- 只有当 RU 为更高版本时，才能手动将数据库实例从 SPB 升级到 RU。

测试 Oracle 数据库升级

在将数据库实例上升级到主要版本前，应该彻底地测试数据库以及访问该数据库的所有应用程序，了解是否与新版本兼容。我们建议您使用以下过程。

测试主要版本升级

1. 检查新版数据库引擎的 Oracle 升级文档以了解是否存在可能会影响数据库或应用程序的兼容性问题。有关更多信息，请参阅 Oracle 文档中的数据库升级指南。

2. 如果数据库实例使用自定义选项组，则创建与您正升级到的新版本兼容的新选项组。有关更多信息，请参阅“选项组注意事项”。
3. 如果数据库实例使用自定义参数组，则创建与您正升级到的新版本兼容的新参数组。有关更多信息，请参阅“参数组注意事项”。
4. 为要升级的数据库实例创建数据库快照。有关更多信息，请参阅“为 Amazon RDS 的单可用区数据库实例创建数据库快照”。
5. 恢复数据库快照，以便创建一个新的测试数据库实例。有关更多信息，请参阅“还原到数据库实例”。
6. 使用下面介绍的一种方法来修改此新测试数据库实例，以将其升级到新版本：
   - 控制台
   - Amazon CLI
   - RDS API
7. 执行测试：
   - 对升级的数据库实例运行所需次数的质量保证测试以确保数据库和应用程序可正常使用新版本。
   - 实施任何必需的新测试，评估您在步骤 1 中发现的任何兼容性问题的影响。
   - 测试存储的所有过程、函数和触发器。
   - 将应用程序的测试版本指向所升级的数据库实例。验证各应用程序可与新版本一起正常使用。
   - 计算升级的实例使用的存储容量以决定升级是否需要额外的存储容量。为了在生产中支持新版本，您可能需要选择较大的实例类。有关更多信息，请参阅“数据库实例类”。
8. 如果所有测试都通过，请升级您的生产数据库实例。我们建议您在允许对数据库实例进行写入操作之前确认数据库实例是否正常工作。

升级 RDS for Oracle 数据库实例版本

要手动升级 RDS for Oracle 数据库实例的数据库引擎版本，可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API。有关 RDS 数据库升级的一般信息，请参阅升级 RDS for Oracle 数据库实例版本。要实现有效的升级目标，请使用 Amazon CLI describe-db-engine-versions 命令。

控制台

使用控制台升级 RDS for Oracle 数据库实例的引擎版本

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)，然后选择要升级的数据库实例。
3. 选择修改。
4. 对于数据库引擎版本，请选择更高的数据库版本。
5. 选择继续，查看修改摘要。请确保您了解数据库版本升级的含义。无法将升级的数据库实例转换回之前的版本。在继续操作之前，请确保已经使用新版本测试了数据库和应用程序。
6. 决定何时安排升级数据库实例。要立即应用更改，请选择立即应用。选择此选项在某些情况下可能导致中断。有关更多信息，请参阅“使用计划修改设置”。
7. 在确认页面上，检查您的更改。如果更改正确无误，请选择 Modify DB Instance（修改数据库实例）保存更改。

   或者，选择 Back (返回) 编辑您的更改，或者选择 Cancel (取消) 取消更改。

Amazon CLI

要升级 RDS for Oracle 数据库实例的引擎版本，可以使用 CLI modify-db-instance 命令。指定以下参数：

- `--db-instance-identifier` – RDS for Oracle 数据库实例的名称。
- `--engine-version` – 数据库引擎要升级到的版本号。

  有关有效的引擎版本的信息，请使用 Amazon CLI describe-db-engine-versions 命令。
- `--allow-major-version-upgrade` – 升级数据库引擎版本。
- `--no-apply-immediately` – 在下一维护时段内应用更改。要立即应用更改，请使用 `--apply-immediately`。

Example

以下示例将名为 `myorainst` 的 CDB 实例从其当前版本 `19.0.0.0.ru-2024-01.rur-2024-01.r1` 升级到版本 `21.0.0.0.ru-2024-04.rur-2024-04.r1`。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier myorainst \
    --engine-version 21.0.0.0.ru-2024-04.rur-2024-04.r1 \
    --allow-major-version-upgrade \
    --no-apply-immediately
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier myorainst ^
    --engine-version 21.0.0.0.ru-2024-04.rur-2024-04.r1 ^
    --allow-major-version-upgrade ^
    --no-apply-immediately
```

RDS API

要升级 RDS for Oracle DB 数据库实例，可执行 ModifyDBInstance 操作。指定以下参数：

- `DBInstanceIdentifier` – 数据库实例的名称，例如 *myorainst*。
- `EngineVersion` – 数据库引擎要升级到的版本号。有关有效的引擎版本的信息，请使用 DescribeDBEngineVersions 操作。
- `AllowMajorVersionUpgrade` – 是否允许主版本升级。为此，请将该值设置为 `true`。
- `ApplyImmediately` – 是立即应用更改还是在下一个维护时段内应用更改。要立即应用更改，请将该值设置为 `true`。要在下一个维护时段内应用更改，请将该值设置为 `false`。

升级 Oracle 数据库快照

在 Amazon RDS 中升级 Oracle 数据库快照，可确保数据库保持安全、兼容并得到全面支持。随着较旧 Oracle 版本的补丁支持结束，您可以升级与这些版本关联的任何手动数据库快照，以避免潜在的漏洞或服务限制。有关更多信息，请参阅 Oracle 引擎版本管理。

Amazon RDS 支持在所有 Amazon Web Services 区域中升级快照。

控制台

升级 Oracle 数据库快照

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Snapshots，然后选择您希望升级的数据库快照。
3. 对于 Actions (操作)，选择 Upgrade snapshot (升级快照)。此时会显示 Upgrade snapshot (升级快照) 页面。
4. 选择要将快照升级到的新引擎版本。
5. (可选) 对于选项组，选择所升级数据库快照的选项组。升级数据库实例时的选项组注意事项在升级数据库快照时同样适用。有关更多信息，请参阅"选项组注意事项"。
6. 选择保存更改以保存您的更改。

   在升级过程中，将为此数据库快照禁用所有快照操作。此外，数据库快照状态由 available 变为 upgrading，完成后变为 active。如果由于快照损坏导致数据库快照不能升级，状态将变为 unavailable。此状态的快照无法恢复。

   > **Note**
   >
   > 如果数据库快照升级失败，则快照将回滚到具有原始版本的原始状态。

Amazon CLI

要使用 Amazon CLI 升级 Oracle 数据库快照，请使用以下参数调用 modify-db-snapshot 命令：

- `--db-snapshot-identifier` – 数据库快照的名称。
- `--engine-version` – 要将快照升级到的版本。

您可能还需要包含以下参数。升级数据库实例时的选项组注意事项在升级数据库快照时同样适用。有关更多信息，请参阅"选项组注意事项"。

- `--option-group-name` – 所升级数据库快照的选项组。

Example

以下示例是升级数据库快照的示例。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-snapshot \
    --db-snapshot-identifier mydbsnapshot \
    --engine-version 19.0.0.0.ru-2020-10.rur-2020-10.r1 \
    --option-group-name default:oracle-se2-19
```

对于 Windows：

```
aws rds modify-db-snapshot ^
    --db-snapshot-identifier mydbsnapshot ^
    --engine-version 19.0.0.0.ru-2020-10.rur-2020-10.r1 ^
    --option-group-name default:oracle-se2-19
```

RDS API

要使用 Amazon RDS API 升级 Oracle 数据库快照，请使用以下参数调用 ModifyDBSnapshot 操作：

- `DBSnapshotIdentifier` – 数据库快照的名称。
- `EngineVersion` – 要将快照升级到的版本。

您可能还需要包含 `OptionGroupName` 参数。升级数据库实例时的选项组注意事项在升级数据库快照时同样适用。有关更多信息，请参阅“选项组注意事项”。

使用第三方软件和 RDS for Oracle 数据库实例

您可以托管支持工具和第三方软件的 RDS for Oracle 数据库实例。

主题

将 Oracle GoldenGate 与 Amazon RDS for Oracle 结合使用

Oracle GoldenGate 可收集、复制和管理数据库之间的事务数据。它是一个基于日志的更改数据捕获（CDC）和复制软件包，与数据库结合使用，适用于联机事务处理（OLTP）系统。Oracle GoldenGate 从源数据库创建包含最近更改的数据的跟踪记录文件。然后，它将这些文件推送到服务器，在此处，一个进程将跟踪记录文件转换为标准 SQL 以应用于目标数据库。

Oracle GoldenGate 与 RDS for Oracle 结合使用可支持以下功能：

- 主动/主动数据库复制
- 灾难恢复
- 数据保护
- 区域内和跨区域复制
- 零停机时间迁移和升级
- 在 RDS for Oracle 数据库实例与非 Oracle 数据库之间复制数据

> **Note**
>
> 有关支持的数据库的列表，请参阅 Oracle 文档中的 Oracle Fusion Middleware 支持的系统配置。

您可以将 Oracle GoldenGate 与 RDS for Oracle 结合使用以升级到 Oracle Database 的主要版本。例如，可以使用 Oracle GoldenGate，从 Oracle Database 11g 本地数据库升级到 Amazon RDS 数据库实例上的 Oracle Database 19c。

主题

- Oracle GoldenGate 支持的版本和许可选项
- Oracle GoldenGate 的要求和限制
- Oracle GoldenGate 架构
- 设置 Oracle GoldenGate
- 使用 Oracle GoldenGate 的 EXTRACT 和 REPLICAT 实用程序
- 监控 Oracle GoldenGate
- Oracle GoldenGate 问题排查

Oracle GoldenGate 支持的版本和许可选项

可以将 RDS for Oracle 的标准版 2（SE2）或企业版（EE）与 Oracle GoldenGate 版本 12c 和更高版本结合使用。您可以使用以下 Oracle GoldenGate 功能：

- 支持 Oracle GoldenGate 远程捕获（提取）。
- 使用传统非 CDB 数据库架构的 RDS for Oracle 数据库实例支持捕获（提取）。运行 Oracle Database 21c 或 Oracle Database 19c 版本 19.0.0.0.ru-2024-04.rur-2024-04.r1 或更高版本的 CDB 支持 Oracle GoldenGate 远程 PDB 捕获。
- 使用非 CDB 或 CDB 架构的 RDS for Oracle 数据库实例支持 Oracle GoldenGate 远程交付（复制）。远程交付支持集成复制、并行复制、协调复制和经典复制。
- RDS for Oracle 支持 Oracle GoldenGate 的经典和微服务架构。
- 在使用集成捕获模式时，支持 Oracle GoldenGate DDL 和序列值复制。

您负责管理 Oracle GoldenGate 许可（自带许可）以用于所有 Amazon Web Services 区域中的 Amazon RDS。有关更多信息，请参阅 RDS for Oracle 许可选项。

Oracle GoldenGate 的要求和限制

在使用 Oracle GoldenGate 和 RDS for Oracle 时，请考虑以下要求和限制：

- 您负责设置和管理 Oracle GoldenGate 以用于 RDS for Oracle。
- 您负责设置经过源数据库和目标数据库认证的 Oracle GoldenGate 版本。有关更多信息，请参阅 Oracle 文档中的 Oracle Fusion Middleware 支持的系统配置。
- 您可以在许多不同的 Amazon 环境中将 Oracle GoldenGate 用于许多不同的使用案例。如果您遇到与 Oracle GoldenGate 相关的支持相关问题，请联系 Oracle 支持服务。
- 您可以在使用 Oracle 透明数据加密（TDE）的 RDS for Oracle 数据库实例上使用 Oracle GoldenGate。要保持复制数据的完整性，请使用 Amazon EBS 加密卷或跟踪记录文件加密在 Oracle GoldenGate 枢纽上配置加密。同时，为 Oracle GoldenGate 枢纽与源和目标数据库实例之间发送的数据配置加密。RDS for Oracle 数据库实例支持使用 Oracle 安全套接字层 或 Oracle 本机网络加密 加密。

Oracle GoldenGate 架构

与 Amazon RDS 结合使用的 Oracle GoldenGate 架构包含以下解耦模块：

源数据库

源数据库可以是本地 Oracle 数据库、Amazon EC2 实例上的 Oracle 数据库或 Amazon RDS 数据库实例上的 Oracle 数据库。

Oracle GoldenGate 枢纽

Oracle GoldenGate 枢纽将事务信息从源数据库移动到目标数据库。枢纽可以是以下任意一种：

- 安装了 Oracle Database 和 Oracle GoldenGate 的 Amazon EC2 实例
- 本地 Oracle 安装

可以有多个 Amazon EC2 枢纽。如果要将 Oracle GoldenGate 用于跨区域复制，建议您使用两个枢纽。

目标数据库

目标数据库可位于 Amazon RDS 数据库实例上、位于 Amazon EC2 实例上或位于一个本地位置。

以下各节介绍 Amazon RDS 上 Oracle GoldenGate 的常见场景。

主题

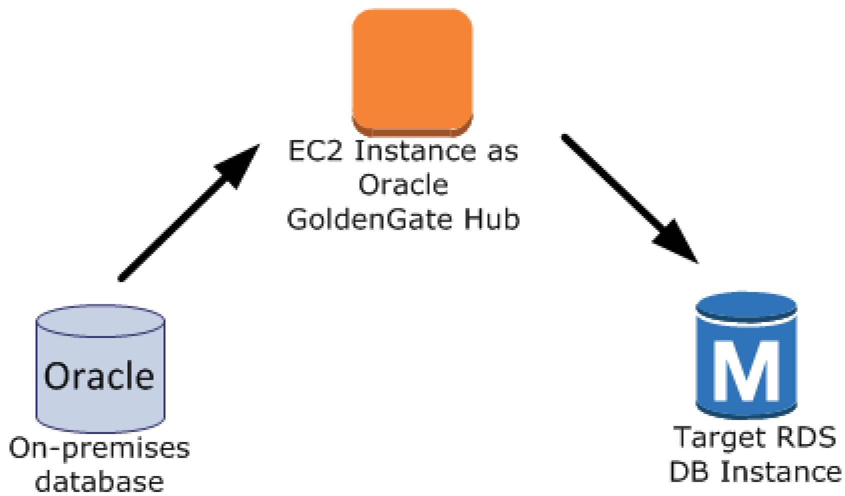
- [本地源数据库和 Oracle GoldenGate 枢纽]
- [本地源数据库和 Amazon EC2 枢纽]
- [Amazon RDS 源数据库和 Amazon EC2 枢纽]
- [Amazon EC2 源数据库和 Amazon EC2 枢纽]
- [不同 Amazon 区域中的 Amazon EC2 枢纽]

本地源数据库和 Oracle GoldenGate 枢纽

在此场景中，本地 Oracle 源数据库和本地 GoldenGate 枢纽向目标 Amazon RDS 数据库实例提供数据。

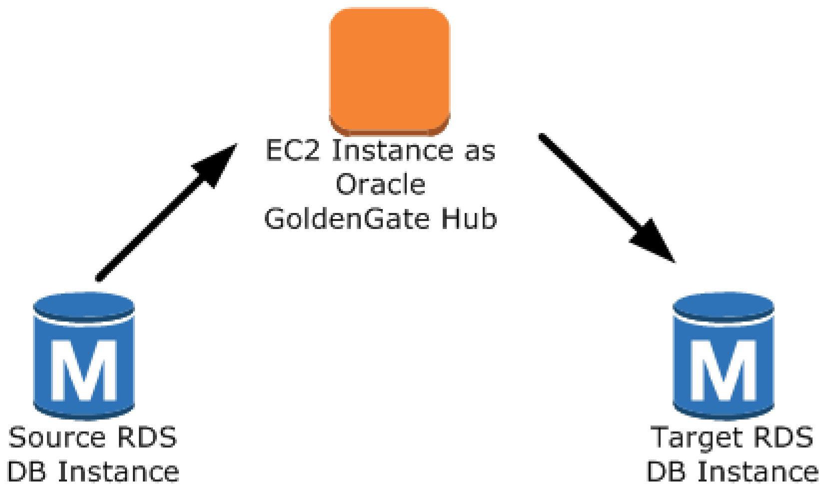
本地源数据库和 Amazon EC2 枢纽

在此场景中，本地 Oracle 数据库充当源数据库。其会连接到 Amazon EC2 实例枢纽。此枢纽向目标 RDS for Oracle 数据库实例提供数据。



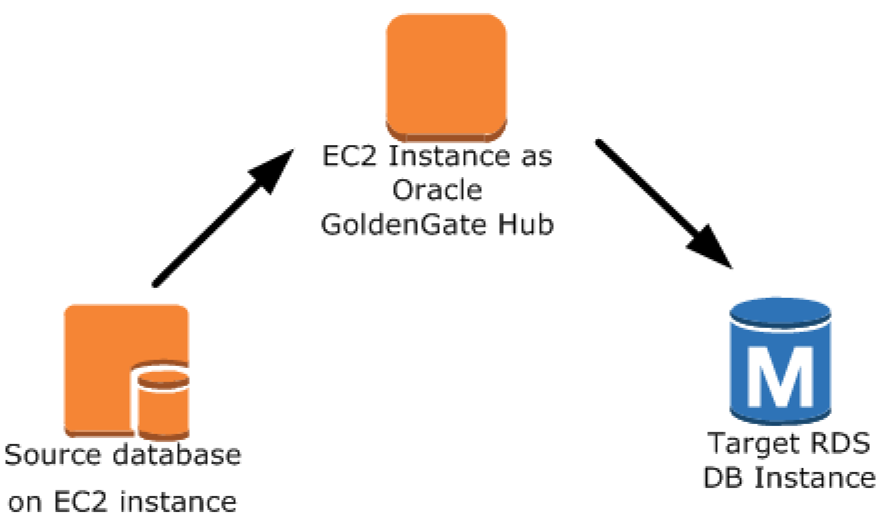
Amazon RDS 源数据库和 Amazon EC2 枢纽

在此场景中，RDS for Oracle 数据库实例充当源数据库。其会连接到 Amazon EC2 实例枢纽。此枢纽向目标 RDS for Oracle 数据库实例提供数据。



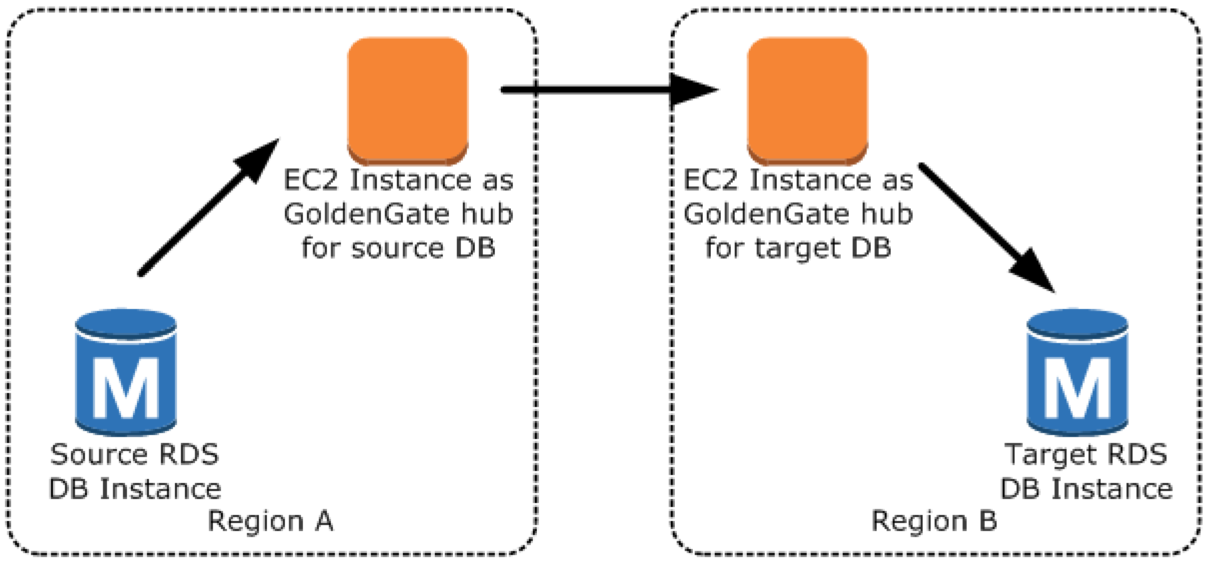
Amazon EC2 源数据库和 Amazon EC2 枢纽

在此场景中，Amazon EC2 实例上的 Oracle 数据库充当源数据库。其会连接到 Amazon EC2 实例枢纽。此枢纽向目标 RDS for Oracle 数据库实例提供数据。



不同 Amazon 区域中的 Amazon EC2 枢纽

在此场景中，Amazon RDS 数据库实例上的 Oracle 数据库连接到相同 Amazon 区域中的 Amazon EC2 实例枢纽。该枢纽连接到其他 Amazon 区域中的 Amazon EC2 实例枢纽。第二个枢纽向与第二个 Amazon EC2 实例枢纽位于相同 Amazon 区域中的目标 RDS for Oracle 数据库实例提供数据。



> **Note**
>
> 影响在本地环境中运行 Oracle GoldenGate 的任何问题也会影响在 Amazon 上运行 Oracle GoldenGate。强烈建议对 Oracle GoldenGate 枢纽进行监控，确保在发生故障转移的情况下恢复 `EXTRACT` 和 `REPLICAT`。由于 Oracle GoldenGate 枢纽在 Amazon EC2 实例上运行，因此 Amazon RDS 不会管理 Oracle GoldenGate 枢纽，也无法确保其是否在运行。

设置 Oracle GoldenGate

要使用 Amazon RDS 设置 Oracle GoldenGate，请在 Amazon EC2 实例上配置枢纽，然后配置源数据库和目标数据库。以下各节举例说明了如何设置 Oracle GoldenGate 以便与 Amazon RDS for Oracle 结合使用。

主题

- [在 Amazon EC2 上设置 Oracle GoldenGate 枢纽]
- [设置源数据库以便与 Amazon RDS 上的 Oracle GoldenGate 结合使用]
- [设置目标数据库以便与 Amazon RDS 上的 Oracle GoldenGate 结合使用]

在 Amazon EC2 上设置 Oracle GoldenGate 枢纽

要在 Amazon EC2 实例上创建 Oracle GoldenGate 枢纽，首先应使用 Oracle RDBMS 的完整客户端安装创建 Amazon EC2 实例。Amazon EC2 实例也必须已安装 Oracle GoldenGate 软件。Oracle

GoldenGate 软件版本取决于源和目标数据库版本。有关安装 Oracle GoldenGate 的更多信息，请参阅 [Oracle GoldenGate 文档](https://docs.oracle.com/goldengate/c1230/gg-winux/index.html)。

用作 Oracle GoldenGate 枢纽的 Amazon EC2 实例会处理来自源数据库的事务信息并将这些信息存储到跟踪记录文件中。要支持此过程，请确保满足以下要求：

- 您为跟踪记录文件分配了足够的存储空间。
- Amazon EC2 实例有足够的处理能力来管理数据量。
- EC2 实例在写入跟踪文件之前，有足够的内存来存储事务信息。

在 Amazon EC2 实例上建立 Oracle GoldenGate 经典架构枢纽

1. 在 Oracle GoldenGate 目录中创建子目录。

   在 Amazon EC2 命令行 shell 中，启动 `ggsci`（Oracle GoldenGate 命令解释器）。`CREATE SUBDIRS` 命令在 `/gg` 目录下为参数、报告和检查点文件创建子目录。

   ```
   prompt$ cd /gg
   prompt$ ./ggsci

   GGSCI> CREATE SUBDIRS
   ```

2. 配置 `mgr.prm` 文件。

   以下示例向 `$GGHOME/dirprm/mgr.prm` 文件中添加行。

   ```
   PORT 8199
   PurgeOldExtracts ./dirdat/*, UseCheckpoints, MINKEEPDAYS 5
   ```

3. 启动管理器。

   以下示例启动 `ggsci` 并运行 `start mgr` 命令。

   ```
   GGSCI> start mgr
   ```

Oracle GoldenGate 枢纽现已就绪，可供使用。

设置源数据库以便与 Amazon RDS 上的 Oracle GoldenGate 结合使用

完成以下任务，设置源数据库以便与 Oracle GoldenGate 结合使用：

设置步骤

- 步骤 1：在源数据库上开启补充日志记录
- 步骤 2：将 ENABLE_GOLDENGATE_REPLICATION 初始化参数设置为 true
- 步骤 3：在源数据库上设置日志保留期
- 步骤 4：在源数据库上创建 Oracle GoldenGate 用户账户。
- 步骤 5：向用户账户授予对源数据库的权限
- 步骤 6：为源数据库添加 TNS 别名

步骤 1：在源数据库上开启补充日志记录

要启用最低数据库级别的补充日志记录，请运行以下 PL/SQL 过程：

```
EXEC rdsadmin.rdsadmin_util.alter_supplemental_logging(p_action => 'ADD')
```

步骤 2：将 ENABLE_GOLDENGATE_REPLICATION 初始化参数设置为 true

将 `ENABLE_GOLDENGATE_REPLICATION` 初始化参数设置为 `true` 时，它允许数据库服务支持逻辑复制。如果源数据库位于 Amazon RDS 数据库实例上，请确保为数据库实例分配一个参数组，且将 `ENABLE_GOLDENGATE_REPLICATION` 初始化参数设置为 `true`。有关 `ENABLE_GOLDENGATE_REPLICATION` 初始化参数的更多信息，请参阅 Oracle Database 文档。

步骤 3：在源数据库上设置日志保留期

确保将源数据库配置为保留存档重做日志。请考虑以下准则：

- 以小时为单位指定日志保留的持续时间。最小值为一小时。
- 设置持续时间以超过源数据库实例的任何可能的停机时间、任何可能的通信时间以及源实例的任何可能的联网问题期间。指定这样的持续时间后，可允许 Oracle GoldenGate 根据需要从源实例恢复日志。
- 确保您的实例上有足够的文件存储空间。

例如，将存档重做日志的保留期设置为 24 小时。

```
EXEC rdsadmin.rdsadmin_util.set_configuration('archivelog retention hours',24)
```

如果未启用日志保留，或者保留值太小，则将收到类似以下内容的错误消息。

```
2022-03-06 06:17:27  ERROR   OGG-00446  error 2 (No such file or directory)
opening redo log /rdsdbdata/db/GGTEST3_A/onlinelog/o1_mf_2_9k4bp1n6_.log for sequence
 1306
Not able to establish initial position for begin time 2022-03-06 06:16:55.
```

由于您的数据库实例会保留存档的重做日志，因此请确保您有足够的空间存放这些文件。要查看过去 *num_hours* 小时内使用了多少空间，请使用以下查询，并将 *num_hours* 替换为小时数。

```
SELECT SUM(BLOCKS * BLOCK_SIZE) BYTES FROM V$ARCHIVED_LOG
   WHERE NEXT_TIME>=SYSDATE-num_hours/24 AND DEST_ID=1;
```

步骤 4：在源数据库上创建 Oracle GoldenGate 用户账户。

Oracle GoldenGate 作为数据库用户运行，并且需要相应数据库权限才能访问源数据库的重做和存档的重做日志。要提供这些权限，请在源数据库上创建用户账户。有关 Oracle GoldenGate 用户账户权限的更多信息，请参阅 Oracle 文档。

下列语句创建名为 oggadm1 的用户账户。

```
CREATE TABLESPACE administrator;
CREATE USER oggadm1  IDENTIFIED BY "password"
   DEFAULT TABLESPACE ADMINISTRATOR TEMPORARY TABLESPACE TEMP;
ALTER USER oggadm1 QUOTA UNLIMITED ON administrator;
```

> **Note**
> 作为安全最佳实践，请指定除此处所示提示以外的密码。

步骤 5：向用户账户授予对源数据库的权限

在此任务中，您将为源数据库上的数据库用户授予必要的账户权限。

向账户授予对源数据库的权限

1. 使用 SQL 命令 `grant` 和 `rdsadmin.rdsadmin_util` 过程 `grant_sys_object` 向 Oracle GoldenGate 用户账户授予必要权限。下列语句向名为 oggadm1 的用户授予权限。

   ```
   GRANT CREATE SESSION, ALTER SESSION TO oggadm1;
   ```

```
GRANT RESOURCE TO oggadm1;
GRANT SELECT ANY DICTIONARY TO oggadm1;
GRANT FLASHBACK ANY TABLE TO oggadm1;
GRANT SELECT ANY TABLE TO oggadm1;
GRANT SELECT_CATALOG_ROLE TO rds_master_user_name WITH ADMIN OPTION;
EXEC rdsadmin.rdsadmin_util.grant_sys_object ('DBA_CLUSTERS', 'OGGADM1');
GRANT EXECUTE ON DBMS_FLASHBACK TO oggadm1;
GRANT SELECT ON SYS.V_$DATABASE TO oggadm1;
GRANT ALTER ANY TABLE TO oggadm1;
```

2. 授予用户账户作为 Oracle GoldenGate 管理员所需的权限。运行以下 PL/SQL 程序。

```
EXEC rdsadmin.rdsadmin_dbms_goldengate_auth.grant_admin_privilege (
    grantee                 => 'OGGADM1',
    privilege_type          => 'capture',
    grant_select_privileges => true,
    do_grants               => TRUE);
```

要撤消权限，请使用同一个程序包中的 `revoke_admin_privilege` 过程。

步骤 6：为源数据库添加 TNS 别名

将以下条目添加到 Oracle 主目录中的 `$ORACLE_HOME/network/admin/tnsnames.ora`，以供 EXTRACT 进程使用。有关 `tnsnames.ora` 文件的更多信息，请参阅 Oracle 文档。

```
OGGSOURCE=
   (DESCRIPTION=
        (ENABLE=BROKEN)
        (ADDRESS_LIST=
            (ADDRESS=(PROTOCOL=TCP)(HOST=goldengate-source.abcdef12345.us-
west-2.rds.amazonaws.com)(PORT=8200)))
        (CONNECT_DATA=(SERVICE_NAME=ORCL))
    )
```

设置目标数据库以便与 Amazon RDS 上的 Oracle GoldenGate 结合使用

在本任务中，您将设置一个目标数据库实例以与 Oracle GoldenGate 结合使用。

设置步骤

- 步骤 1：将 ENABLE_GOLDENGATE_REPLICATION 初始化参数设置为 true
- 步骤 2：在目标数据库上创建 Oracle GoldenGate 用户账户

- [步骤 3：授予账户对目标数据库的权限]
- [步骤 4：为目标数据库添加 TNS 别名]

步骤 1：将 ENABLE_GOLDENGATE_REPLICATION 初始化参数设置为 true

将 `ENABLE_GOLDENGATE_REPLICATION` 初始化参数设置为 `true` 时，它允许数据库服务支持逻辑复制。如果源数据库位于 Amazon RDS 数据库实例上，请确保为数据库实例分配一个参数组，且将 `ENABLE_GOLDENGATE_REPLICATION` 初始化参数设置为 `true`。有关 `ENABLE_GOLDENGATE_REPLICATION` 初始化参数的更多信息，请参阅 Oracle Database 文档。

步骤 2：在目标数据库上创建 Oracle GoldenGate 用户账户

Oracle GoldenGate 以数据库用户身份运行，并且需要相应的数据库权限。为了确保它具备这些权限，请在目标数据库上创建用户帐户。

下列语句将创建名为 `oggadm1` 的用户。

```
CREATE TABLESPSACE administrator;
CREATE USER oggadm1  IDENTIFIED BY "password"
   DEFAULT TABLESPACE administrator
   TEMPORARY TABLESPACE temp;
ALTER USER oggadm1 QUOTA UNLIMITED ON administrator;
```

> **Note**
>
> 作为安全最佳实践，请指定除此处所示提示以外的密码。

步骤 3：授予账户对目标数据库的权限

在此任务中，您将为目标数据库上的数据库用户授予必要的账户权限。

授予账户对目标数据库的权限

1. 向目标数据库上的 Oracle GoldenGate 用户账户授予必要权限。在以下示例中，您向 `oggadm1` 授予权限。

```
GRANT CREATE SESSION        TO oggadm1;
GRANT ALTER SESSION         TO oggadm1;
```

```
GRANT CREATE CLUSTER        TO oggadm1;
GRANT CREATE INDEXTYPE      TO oggadm1;
GRANT CREATE OPERATOR       TO oggadm1;
GRANT CREATE PROCEDURE      TO oggadm1;
GRANT CREATE SEQUENCE       TO oggadm1;
GRANT CREATE TABLE          TO oggadm1;
GRANT CREATE TRIGGER        TO oggadm1;
GRANT CREATE TYPE           TO oggadm1;
GRANT SELECT ANY DICTIONARY TO oggadm1;
GRANT CREATE ANY TABLE      TO oggadm1;
GRANT ALTER ANY TABLE       TO oggadm1;
GRANT LOCK ANY TABLE        TO oggadm1;
GRANT SELECT ANY TABLE      TO oggadm1;
GRANT INSERT ANY TABLE      TO oggadm1;
GRANT UPDATE ANY TABLE      TO oggadm1;
GRANT DELETE ANY TABLE      TO oggadm1;
```

2. 授予用户账户作为 Oracle GoldenGate 管理员所需的权限。运行以下 PL/SQL 程序。

```
EXEC rdsadmin.rdsadmin_dbms_goldengate_auth.grant_admin_privilege (
    grantee                 => 'OGGADM1',
    privilege_type          => 'apply',
    grant_select_privileges => true,
    do_grants               => TRUE);
```

   要撤消权限，请使用同一个程序包中的 `revoke_admin_privilege` 过程。

步骤 4：为目标数据库添加 TNS 别名

将以下条目添加到 Oracle 主目录中的 `$ORACLE_HOME/network/admin/tnsnames.ora`，以供 `REPLICAT` 进程使用。对于 Oracle 多租户数据库，请确保 TNS 别名指向 PDB 的服务名称。有关 `tnsnames.ora` 文件的更多信息，请参阅 [Oracle 文档](#)。

```
OGGTARGET=
    (DESCRIPTION=
        (ENABLE=BROKEN)
        (ADDRESS_LIST=
            (ADDRESS=(PROTOCOL=TCP)(HOST=goldengate-target.abcdef12345.us-
west-2.rds.amazonaws.com)(PORT=8200)))
        (CONNECT_DATA=(SERVICE_NAME=ORCL))
    )
```

使用 Oracle GoldenGate 的 EXTRACT 和 REPLICAT 实用程序

Oracle GoldenGate 实用程序 EXTRACT 和 REPLICAT 结合使用，以使用跟踪记录文件通过增量事务复制将源数据库和目标数据库保持同步。源数据库上发生的所有更改会由 EXTRACT 自动检测，进行格式设置，然后传输至 Oracle GoldenGate 本地或 Amazon EC2 实例枢纽上的跟踪文件。初始加载完成后，由 REPLICAT 实用程序从这些文件读取数据并复制到目标数据库。

运行 Oracle GoldenGate EXTRACT 实用程序

EXTRACT 实用程序可对源数据库中的数据进行检索、转换并输出至跟踪文件。基本过程如下所示：

1. EXTRACT 将事务详细信息排队到内存或临时磁盘存储中。
2. 源数据库提交事务。
3. EXTRACT 将事务详细信息写入跟踪文件。
4. 跟踪记录文件将这些详细信息路由到 Oracle GoldenGate 本地或 Amazon EC2 实例枢纽，然后路由到目标数据库。

以下步骤启动 EXTRACT 实用程序，从源数据库 OGGSOURCE 的 EXAMPLE.TABLE 中捕获数据，然后创建跟踪记录文件。

运行 EXTRACT 实用程序

1. 在 Oracle GoldenGate 枢纽（本地或 Amazon EC2 实例）上配置 EXTRACT 参数文件。下面的列表显示了一个名为 $GGHOME/dirprm/eabc.prm 的示例 EXTRACT 参数文件。

```
EXTRACT EABC

USERID oggadm1@OGGSOURCE, PASSWORD "my-password"
EXTTRAIL /path/to/goldengate/dirdat/ab

IGNOREREPLICATES
GETAPPLOPS
TRANLOGOPTIONS EXCLUDEUSER OGGADM1

TABLE EXAMPLE.TABLE;
```

2. 在 Oracle GoldenGate 枢纽上，登录到源数据库，并启动 Oracle GoldenGate 命令行界面 ggsci。下面的示例说明了登录格式。

```
dblogin oggadm1@OGGSOURCE
```

3. 添加事务数据以便为数据库表开启补充日志记录。

```
add trandata EXAMPLE.TABLE
```

4. 使用 `ggsci` 命令行以及下列命令，启用 EXTRACT 实用程序。

```
add extract EABC tranlog, INTEGRATED tranlog, begin now
add exttrail /path/to/goldengate/dirdat/ab
   extract EABC,
   MEGABYTES 100
```

5. 向数据库注册 EXTRACT 实用程序，以便不删除存档日志。此任务可让您在必要时恢复未提交的旧事务。要向数据库注册 EXTRACT 实用程序，请使用下面的命令。

```
register EXTRACT EABC, DATABASE
```

6. 使用下面的命令启动 EXTRACT 实用程序。

```
start EABC
```

运行 Oracle GoldenGate REPLICAT 实用程序

REPLICAT 实用程序将跟踪记录文件中的事务信息“推送”至目标数据库。

以下步骤启用并启动 REPLICAT 实用程序，以便它可以将捕获的数据复制到目标数据库 OGGTARGET 的表 EXAMPLE.TABLE 中。

运行 REPLICATE 实用程序

1. 在 Oracle GoldenGate 枢纽（本地或 EC2 实例）上配置 REPLICAT 参数文件。下面的列表显示了一个名为 `$GGHOME/dirprm/rabc.prm` 的示例 REPLICAT 参数文件。

```
REPLICAT RABC

USERID oggadm1@OGGTARGET, password "my-password"

ASSUMETARGETDEFS
```

```
MAP EXAMPLE.TABLE, TARGET EXAMPLE.TABLE;
```

> **Note**
> 作为安全最佳实践，请指定除此处所示提示以外的密码。

2. 登录目标数据库并启动 Oracle GoldenGate 命令行界面（`ggsci`）。下面的示例说明了登录格式。

```
dblogin userid oggadm1@OGGTARGET
```

3. 使用 `ggsci` 命令行，添加检查点表。所指示的用户应是 Oracle GoldenGate 用户账户，而不是目标表架构拥有者。下面的示例创建一个名为 `gg_checkpoint` 的检查点表。

```
add checkpointtable oggadm1.oggchkpt
```

4. 要启用 `REPLICAT` 实用程序，请使用下面的命令。

```
add replicat RABC EXTTRAIL /path/to/goldengate/dirdat/ab CHECKPOINTTABLE
 oggadm1.oggchkpt
```

5. 使用下面的命令启动 `REPLICAT` 实用程序。

```
start RABC
```

监控 Oracle GoldenGate

使用 Oracle GoldenGate 进行复制时，请确保 Oracle GoldenGate 进程已启动并正在运行，且源数据库和目标数据库已同步。您可以使用以下监控工具：

- Amazon CloudWatch 是一种监控服务，在这种模式下用于监控 GoldenGate 错误日志。
- Amazon SNS 是一种消息通知服务，在此模式中用于发送电子邮件通知。

有关详细说明，请参阅使用 Amazon CloudWatch 监控 Oracle GoldenGate 日志。

Oracle GoldenGate 问题排查

本节说明 Oracle GoldenGate 与 Amazon RDS for Oracle 结合使用时的最常见问题。

主题

- [打开联机重做日志时出错]
- [Oracle GoldenGate 似乎配置正确，但复制不起作用]
- [由于 SYS 上的查询，集成 REPLICAT 速度缓慢。"_DBA_APPLY_CDR_INFO"]

打开联机重做日志时出错

确保将数据库配置为保留存档的重做日志。请考虑以下准则：

- 以小时为单位指定日志保留的持续时间。最小值为一小时。
- 设置持续时间以超过源数据库实例的任何可能的停机时间、任何可能的通信时间以及源数据库实例的任何可能的联网问题期间。指定这样的持续时间后，可允许 Oracle GoldenGate 根据需要从源数据库实例恢复日志。
- 确保您的实例上有足够的文件存储空间。

如果未启用日志保留，或者保留值太小，则将收到类似以下内容的错误消息。

```
2022-03-06 06:17:27  ERROR   OGG-00446  error 2 (No such file or directory)
opening redo log /rdsdbdata/db/GGTEST3_A/onlinelog/o1_mf_2_9k4bp1n6_.log for sequence
 1306
Not able to establish initial position for begin time 2022-03-06 06:16:55.
```

Oracle GoldenGate 似乎配置正确，但复制不起作用

对于预先存在的表，您必须指定 Oracle GoldenGate 从中运行的 SCN。

修复此问题

1. 登录源数据库并启动 Oracle GoldenGate 命令行界面（`ggsci`）。下面的示例说明了登录格式。

```
dblogin userid oggadm1@OGGSOURCE
```

2. 使用 `ggsci` 命令行为 `EXTRACT` 进程设置起始 SCN。以下示例为 `EXTRACT` 将 SCN 设置为 223274。

```
ALTER EXTRACT EABC SCN 223274
start EABC
```

3. 登录到目标数据库。下面的示例说明了登录格式。

```
dblogin userid oggadm1@OGGTARGET
```

4. 使用 ggsci 命令行为 REPLICAT 进程设置起始 SCN。以下示例为 REPLICAT 将 SCN 设置为 223274。

```
start RABC atcsn 223274
```

由于 SYS 上的查询，集成 REPLICAT 速度缓慢。"_DBA_APPLY_CDR_INFO"

Oracle GoldenGate 冲突检测和解决 (CDR) 提供了基本的冲突解决例程。例如，CDR 可以解决 INSERT 语句的唯一冲突。

在 CDR 解决冲突时，它可以临时将记录插入异常表 _DBA_APPLY_CDR_INFO 中。集成的 REPLICAT 稍后会删除这些记录。在极少数情况下，集成的 REPLICAT 可以处理大量冲突，但是新的集成 REPLICAT 并不能取代它。_DBA_APPLY_CDR_INFO 中的现有行不会被删除而是被孤立。任何新的集成 REPLICAT 进程都会减慢，因为它们正在查询 _DBA_APPLY_CDR_INFO 中的孤立行。

要从 _DBA_APPLY_CDR_INFO 中删除所有行，请使用 Amazon RDS 过程 rdsadmin.rdsadmin_util.truncate_apply$_cdr_info。此过程是作为 2020 年 10 月版本和补丁程序更新的一部分发布的。以下数据库版本中提供该过程：

- 版本 21.0.0.0.ru-2022-01.rur-2022-01.r1 及更高版本
- 版本 19.0.0.0.ru-2020-10.rur-2020-10.r1 及更高版本

以下示例展示了将表 _DBA_APPLY_CDR_INFO 截断。

```
SET SERVEROUTPUT ON SIZE 2000
EXEC rdsadmin.rdsadmin_util.truncate_apply$_cdr_info;
```

在 RDS for Oracle 上使用 Oracle Repository Creation Utility

您可以使用 Amazon RDS 托管 RDS for Oracle 数据库实例，此实例保留架构以支持 Oracle Fusion Middleware 组件。必须先在数据库中为 Fusion Middleware 组件创建和填充架构，然后才能使用这些组件。可使用 Oracle Repository Creation Utility (RCU) 来创建和填充架构。

RCU 支持的版本和许可选项

Amazon RDS 仅支持 Oracle Repository Creation Utility (RCU) 版本 12c。您可以在以下配置中使用 RCU：

- 带 Oracle Database 21c 的 RCU 12c
- 带 Oracle Database 19c 的 RCU 12c

在使用 RCU 之前，请确保执行以下操作：

- 获得 Oracle Fusion Middleware 的许可证。
- 遵循托管存储库的 Oracle 数据库的 Oracle 许可指南。有关更多信息，请参阅 Oracle 文档中的 Oracle Fusion Middleware 许可信息用户手册。

Fusion MiddleWare 支持 Oracle Database 企业版和标准版 2 上的存储库。Oracle 建议将 Enterprise Edition 用于需要分区的生产安装和需要在线重新建立索引的安装。

在创建 RDS for Oracle 数据库实例之前，确认支持要部署的组件所需的 Oracle 数据库版本。要查找要部署的 Fusion Middleware 组件和版本的要求，请使用 Certification Matrix。有关更多信息，请参阅 Oracle 文档中的 Oracle Fusion Middleware 支持的系统配置。

Amazon RDS 根据需要支持 Oracle 数据库版本升级。有关更多信息，请参阅 升级数据库实例引擎版本。

RCU 的要求和限制

您需要 Amazon VPC 才能使用 RCU。您的 Amazon RDS 数据库实例必须仅对您的 Fusion Middleware 组件可用，而不是公共互联网。因此，在私有子网中托管您的 Amazon RDS 数据库实例，可提供更好的安全性。您还需要 RDS for Oracle 数据库实例。有关更多信息，请参阅 创建并连接到 Oracle 数据库实例。

可以为 Amazon RDS 数据库实例中的任何 Fusion Middleware 组件存储架构。已经验证以下架构安装正确：

- 分析 (ACTIVITIES)
- 审核服务 (IAU)
- Audit Services Append (IAU_APPEND)
- Audit Services Viewer (IAU_VIEWER)
- 讨论 (DISCUSSIONS)
- 元数据服务 (MDS)
- Oracle Business Intelligence (BIPLATFORM)
- Oracle Platform Security Services (OPSS)
- 门户和服务 (WEBCENTER)
- Portlet 创建者 (PORTLET)
- 服务表 (STB)
- SOA 基础设施 (SOAINFRA)
- User Messaging Service (UCSUMS)
- WebLogic Services (WLS)

RCU 的使用准则

以下是针对在此方案中使用数据库实例的一些建议：

- 建议您对生产工作负载使用多可用区。有关使用多个可用区的更多信息，请参阅区域、可用区和 Local Zones。
- 为了提高安全性，Oracle 建议您使用透明数据加密 (TDE) 对静态数据进行加密。如果您拥有包含高级安全选项的 Enterprise Edition 许可证，则可使用 TDE 选项启用静态加密。有关更多信息，请参阅"Oracle 透明数据加密"。

  Amazon RDS 还提供适用于所有数据库版本的静态加密选项。有关更多信息，请参阅"加密 Amazon RDS 资源"。
- 将 VPC 安全组配置为允许应用程序服务器和 Amazon RDS 数据库实例之间的通信。承载 Fusion Middleware 组件的应用程序服务器可位于 Amazon EC2 上或位于本地。

运行 RCU

要创建和填充架构以支持 Fusion Middleware 组件，请使用 Oracle Repository Creation Utility (RCU)。可通过不同方式运行 RCU。

主题

使用命令行通过一个步骤运行 RCU

如果您在填充任何架构之前不需要对其进行编辑，则可通过一个步骤运行 RCU。否则，请参阅下一部分以了解通过多个步骤运行 RCU。

可使用命令行参数 `-silent` 在静默模式下运行 RCU。在静默模式下运行 RCU 时，可通过创建包含密码的文本文件来避免在命令行中输入密码。创建一个文本文件，其中 `dbUser` 的密码位于第一行，每个组件的密码位于后续行。您将密码文件的名称指定为 RCU 命令的最后一个参数。

Example

以下示例通过一个步骤为 SOA 基础设施组件 (及其依赖项) 创建和填充建构。

对于 Linux、macOS 或 Unix：

```
export ORACLE_HOME=/u01/app/oracle/product/12.2.1.0/fmw
export JAVA_HOME=/usr/java/jdk1.8.0_65
${ORACLE_HOME}/oracle_common/bin/rcu \
-silent \
-createRepository \
-connectString ${dbhost}:${dbport}:${dbname} \
-dbUser ${dbuser} \
-dbRole Normal \
-honorOMF \
-schemaPrefix ${SCHEMA_PREFIX} \
-component MDS \
-component STB \
-component OPSS \
-component IAU \
-component IAU_APPEND \
-component IAU_VIEWER \
-component UCSUMS \
-component WLS \
-component SOAINFRA \
-f < /tmp/passwordfile.txt
```

有关更多信息，请参阅 Oracle 文档中的从命令行运行 Repository Creation Utility。

使用命令行通过多个步骤运行 RCU

要手动编辑架构脚本，请通过多个步骤运行 RCU：

1. 使用 `-generateScript` 命令行参数在 Prepare Scripts for System Load 模式下运行 RCU 以便为架构创建脚本。
2. 手动编辑和运行生成的脚本 `script_systemLoad.sql`。
3. 再次使用 `-dataLoad` 命令行参数在 Perform Product Load 模式下运行 RCU 以填充架构。
4. 运行生成的清理脚本 `script_postDataLoad.sql`。

要在静默模式下运行 RCU，请指定命令行参数 `-silent`。在静默模式下运行 RCU 时，可通过创建包含密码的文本文件来避免在命令行中键入密码。创建一个文本文件，其中 `dbUser` 的密码位于第一行，每个组件的密码位于后续行。将密码文件的名称指定为 RCU 命令的最后一个参数。

Example

以下示例为 SOA 基础设施组件及其依赖项创建架构脚本。

对于 Linux、macOS 或 Unix：

```
export ORACLE_HOME=/u01/app/oracle/product/12.2.1.0/fmw
export JAVA_HOME=/usr/java/jdk1.8.0_65
${ORACLE_HOME}/oracle_common/bin/rcu \
-silent \
-generateScript \
-connectString ${dbhost}:${dbport}:${dbname} \
-dbUser ${dbuser} \
-dbRole Normal \
-honorOMF \
[-encryptTablespace true] \
-schemaPrefix ${SCHEMA_PREFIX} \
-component MDS \
-component STB \
-component OPSS \
-component IAU \
-component IAU_APPEND \
-component IAU_VIEWER \
-component UCSUMS \
-component WLS \
-component SOAINFRA \
```

```
-scriptLocation /tmp/rcuscripts \
-f < /tmp/passwordfile.txt
```

现在，您可以编辑生成的脚本，连接到您的 Oracle 数据库实例和运行脚本。生成的脚本名为 `script_systemLoad.sql`。有关连接到 Oracle 数据库实例的信息，请参阅步骤 3：将 SQL 客户端连接到 Oracle 数据库实例。

以下示例为 SOA 基础设施组件 (及其依赖项) 填充架构。

对于 Linux、macOS 或 Unix：

```
export JAVA_HOME=/usr/java/jdk1.8.0_65
${ORACLE_HOME}/oracle_common/bin/rcu \
-silent \
-dataLoad \
-connectString ${dbhost}:${dbport}:${dbname} \
-dbUser ${dbuser} \
-dbRole Normal \
-honorOMF \
-schemaPrefix ${SCHEMA_PREFIX} \
-component MDS \
-component STB \
-component OPSS \
-component IAU \
-component IAU_APPEND \
-component IAU_VIEWER \
-component UCSUMS \
-component WLS \
-component SOAINFRA \
-f < /tmp/passwordfile.txt
```

最后，您连接到 Oracle 数据库实例，并运行清除脚本。该脚本名为 `script_postDataLoad.sql`。

有关更多信息，请参阅 Oracle 文档中的从命令行运行 Repository Creation Utility。

以交互式模式运行 RCU

要使用 RCU 图形用户界面，请以交互模式运行 RCU。加入 `-interactive` 参数，然后省略 `-silent` 参数。有关更多信息，请参阅 Oracle 文档中的了解 Repository Creation Utility 屏幕。

Example

以下示例以交互模式启动 RCU 并预填写连接信息。

对于 Linux、macOS 或 Unix：

```
export ORACLE_HOME=/u01/app/oracle/product/12.2.1.0/fmw
export JAVA_HOME=/usr/java/jdk1.8.0_65
${ORACLE_HOME}/oracle_common/bin/rcu \
-interactive \
-createRepository \
-connectString ${dbhost}:${dbport}:${dbname} \
-dbUser ${dbuser} \
-dbRole Normal
```

RCU 问题排查

请注意以下问题。

主题

- Oracle Managed Files (OMF)
- 对象权限
- Enterprise Scheduler Service

Oracle Managed Files (OMF)

Amazon RDS 使用 OMF 数据文件来简化存储管理。您可以自定义表空间属性，例如大小和范围管理。但是，如果在运行 RCU 时指定数据文件名，则表空间代码将失败，错误为 `ORA-20900`。可通过以下方式将 RCU 与 OMF 结合使用：

- 在 RCU 12.2.1.0 和更高版本中，使用 `-honorOMF` 命令行参数。
- 在 RCU 12.1.0.3 和更高版本中，使用多个步骤并编辑生成的脚本。有关更多信息，请参阅 使用命令行通过多个步骤运行 RCU。

对象权限

由于 Amazon RDS 是一项托管式服务，因此，您不具有对 RDS for Oracle 数据库实例的完全 SYSDBA 访问权限。不过，RCU 12c 支持具有较低权限的用户。在大多数情况下，主用户权限足以创建存储库。

主账户可以直接授予已授予 `WITH GRANT OPTION` 的权限。在某些情况下，在尝试授予 `ORA-01031` 对象权限时，RCU 可能失败，错误为 `SYS`。您可以重试并运行 `rdsadmin_util.grant_sys_object` 存储过程，如以下示例所示：

```
BEGIN
  rdsadmin.rdsadmin_util.grant_sys_object('GV_$SESSION','MY_DBA','SELECT');
END;
/
```

如果您尝试授予对于对象 SCHEMA_VERSION_REGISTRY 的 SYS 权限，则操作可能会失败，错误为 `ORA-20199: Error in rdsadmin_util.grant_sys_object`。您可以使用架构拥有者名称（即 SYSTEM）限定表 SCHEMA_VERSION_REGISTRY$ 和视图 SCHEMA_VERSION_REGISTRY，然后重试该操作。或者，您可以创建同义词。以主用户身份登录并运行以下语句：

```
CREATE OR REPLACE VIEW SYSTEM.SCHEMA_VERSION_REGISTRY
  AS SELECT * FROM SYSTEM.SCHEMA_VERSION_REGISTRY$;
CREATE OR REPLACE PUBLIC SYNONYM SCHEMA_VERSION_REGISTRY FOR
 SYSTEM.SCHEMA_VERSION_REGISTRY;
CREATE OR REPLACE PUBLIC SYNONYM SCHEMA_VERSION_REGISTRY$ FOR SCHEMA_VERSION_REGISTRY;
```

Enterprise Scheduler Service

当您使用 RCU 删除 Enterprise Scheduler Service 存储库时，RCU 可能失败，错误为 `Error: Component drop check failed`。

在 Amazon EC2 实例上配置 Oracle Connection Manager

Oracle Connection Manager (CMAN) 是一种代理服务器，可将连接请求转发到数据库服务器或其他代理服务器。您可以使用 CMAN 配置以下内容：

访问控制

  您可以创建规则来筛选掉用户指定的客户端请求并接受其他请求。

会话多路复用

  您可以通过网络连接将多个客户端会话汇集到共享服务器目标。

通常，CMAN 驻留在独立于数据库服务器和客户端主机的主机上。有关更多信息，请参阅 Oracle Database 文档中的配置 Oracle Connection Manager。

主题

- CMAN 支持的版本和许可选项
- CMAN 的要求和限制
- 配置 CMAN

CMAN 支持的版本和许可选项

CMAN 支持 Amazon RDS 支持的所有版本的 Oracle Database Enterprise Edition。有关更多信息，请参阅 RDS for Oracle 版本。

您可以在安装 Oracle Database 的主机之外的主机上安装 Oracle Connection Manager。运行 CMAN 的主机不需要单独的许可证。

CMAN 的要求和限制

为了提供完全托管式体验，Amazon RDS 限制了对操作系统的访问。您无法修改需要操作系统访问权限的数据库参数。因此，Amazon RDS 不支持要求您登录操作系统的 CMAN 功能。

配置 CMAN

配置 CMAN 时，您会在 RDS for Oracle 数据库之外执行大部分工作。

主题

- 步骤 1：在与 RDS for Oracle 实例相同的 VPC 中的 Amazon EC2 实例上配置 CMAN
- 步骤 2：为 CMAN 配置数据库参数
- 步骤 3：将数据库实例与参数组相关联

步骤 1：在与 RDS for Oracle 实例相同的 VPC 中的 Amazon EC2 实例上配置 CMAN

要了解如何设置 CMAN，请按照博客文章《在 Amazon EC2 上为 Amazon RDS for Oracle 配置和使用 Oracle Connection Manager》中的详细说明进行操作。

步骤 2：为 CMAN 配置数据库参数

对于 CMAN 功能（如 Traffic Director 模式和会话多路复用），请将 `REMOTE_LISTENER` 参数设置为数据库参数组中 CMAN 实例的地址。考虑以下情况：

- CMAN 实例驻留在具有 IP 地址 `10.0.159.100` 的主机上，并使用端口 `1521`。
- 数据库 `orcla`、`orclb` 和 `orclc` 位于单独的 RDS for Oracle 数据库实例上。

下表显示了如何设置 `REMOTE_LISTENER` 值。Amazon RDS 自动设置 `LOCAL_LISTENER` 值。

数据库实例名称	数据库实例 IP	本地侦听器值（自动设置）	远程侦听器值（用户设置）
`orcla`	`10.0.159.200`	`( address= (protocol=tcp) (host=10.0.159.200) (port=1521) )`	`10.0.159.100:1521`
`orclb`	`10.0.159.300`	`( address= (protocol=tcp) (host=10.0.159.300) (port=1521) )`	`10.0.159.100:1521`
`orclc`	`10.0.159.400`	`( address= (protocol=tcp) (host=10.0.159.400)`	`10.0.159.100:1521`

数据库实例名称	数据库实例 IP	本地侦听器值（自动设置）	远程侦听器值（用户设置）
		<pre> (port=1521))</pre>	

步骤 3：将数据库实例与参数组相关联

创建或修改数据库实例，以便使用在 步骤 2：为 CMAN 配置数据库参数 中配置的参数组。有关更多信息，请参阅 在 Amazon RDS 中将数据库参数组与数据库实例关联。

在 Amazon RDS 上的 Oracle 中安装 Siebel 数据库

您可以使用 Amazon RDS 将 Siebel 数据库托管在 Oracle 数据库实例上。Siebel 数据库是 Siebel 客户关系管理 (CRM) 应用程序架构的一部分。有关说明，请参阅 Siebel 业务应用程序的通用架构。

使用以下主题可帮助在 Amazon RDS 上的 Oracle 数据库实例中设置 Siebel 数据库。您还可以了解如何使用 Amazon Web Services 支持 Siebel CRM 应用程序架构所需的其他组件。

> **Note**
>
> 要在 Amazon RDS 上的 Oracle 中安装 Siebel 数据库，您需要使用主用户账户。您无需 `SYSDBA` 权限；主用户权限就已足够。有关更多信息，请参阅"主用户账户权限"。

许可和版本

要在 Amazon RDS 上安装 Siebel 数据库，您必须使用自己的 Oracle Database 许可和 Siebel 许可。您必须拥有与数据库实例类和 Oracle Database 版本对应的 Oracle Database 许可 (及软件更新授权和支持)。有关更多信息，请参阅"RDS for Oracle 许可选项"。

Oracle Database Enterprise Edition 是 Siebel 针对此场景发布的唯一认证版本。Amazon RDS 支持 Siebel CRM 15.0 版或 16.0 版。

Amazon RDS 支持数据库版本升级。有关更多信息，请参阅"升级数据库实例引擎版本"。

开始前的准备工作

在开始之前，您需要 Amazon VPC。由于您的 Amazon RDS 数据库实例仅需要在 Siebel Enterprise Server (而非公共 Internet) 上可用，因此您的 Amazon RDS 数据库实例将托管在私有子网上，以便提高安全性。有关如何创建 Amazon VPC，从而与 Siebel CRM 配合使用的信息，请参阅 创建并连接到 Oracle 数据库实例。

在开始之前，您还需要 Oracle 数据库实例。有关如何创建用于 Siebel CRM 的 Oracle 数据库实例的信息，请参阅创建 Amazon RDS 数据库实例。

安装并配置 Siebel 数据库

创建 Oracle 数据库实例后，您可以安装 Siebel 数据库。您可以通过以下方法安装数据库：创建表所有者和管理员账户，安装存储的过程功能，然后运行 Siebel 数据库配置向导。有关更多信息，请参阅在 RDBMS 上安装 Siebel 数据库。

要运行 Siebel 数据库配置向导，您需要使用主用户账户。您无需 SYSDBA 权限；主用户权限就已足够。有关更多信息，请参阅“主用户账户权限”。

将其他 Amazon RDS 功能与 Siebel 数据库结合使用

创建 Oracle 数据库实例后，可使用其他 Amazon RDS 功能来帮助您自定义 Siebel 数据库。

使用 Oracle Statspack 选项收集统计信息

您可以通过使用数据库选项组中的选项向数据库实例添加各项功能。创建 Oracle 数据库实例时，您使用了默认的数据库选项组。如果要向数据库添加功能，您可以为数据库实例创建新的选项组。

如果要收集关于 Siebel 数据库的性能统计数据，您可以添加 Oracle Statspack 功能。有关更多信息，请参阅“Oracle Statspack”。

有些选项更改将立即应用，另一些选项更改将在数据库实例的下一维护时段内应用。有关更多信息，请参阅“使用选项组”。创建自定义选项组后，请修改数据库实例以附加选项组。有关更多信息，请参阅“修改 Amazon RDS 数据库实例”。

使用参数优化性能

您可以使用数据库参数组中的参数来管理数据库引擎配置。创建 Oracle 数据库实例时，您使用了默认的数据库参数组。如果要自定义数据库配置，您可以为数据库实例创建一个新的参数组。

更改参数时，更改可以立即生效，也可以在手动重新启动数据库实例后生效，具体取决于参数类型。有关更多信息，请参阅“Amazon RDS 的参数组”。创建自定义参数组后，请修改数据库实例以附加参数组。有关更多信息，请参阅“修改 Amazon RDS 数据库实例”。

要针对 Siebel CRM 优化 Oracle 数据库实例，您可以自定义某些参数。下表显示了一些推荐的参数设置。有关 Siebel CRM 性能优化的更多信息，请参阅 Siebel CRM 性能优化指南。

参数名称	默认值	最佳 Siebel CRM 性能指南
_always_semi_join	CHOOSE	OFF
_b_tree_bitmap_plans	TRUE	FALSE

参数名称	默认值	最佳 Siebel CRM 性能指南
_like_with_bind_as_equality	FALSE	TRUE
_no_or_expansion	FALSE	FALSE
_optimizer_join_sel_sanity_check	TRUE	TRUE
_optimizer_max_permutations	2000	100
_optimizer_sortmerge_join_enabled	TRUE	FALSE
_partition_view_enabled	TRUE	FALSE
open_cursors	300	至少 **2000**。

创建快照

创建 Siebel 数据库后，您可以使用 Amazon RDS 的快照功能来复制数据库。有关更多信息，请参阅"为 Amazon RDS 的单可用区数据库实例创建数据库快照"和"还原到数据库实例"。

对其他 Siebel CRM 组件的支持

除了 Siebel 数据库以外，您还可以使用 Amazon Web Services 来支持 Siebel CRM 应用程序架构的其他组件。可在下表中找到有关 Amazon Amazon 为其他 Siebel CRM 组件提供的支持的更多信息。

Siebel CRM 组件	Amazon Amazon Support
Siebel Enterprise (配备一个或多个 Siebel 服务器)	您可以在 Amazon Elastic Compute Cloud (Amazon EC2) 实例上托管 Siebel 服务器。您可以根据需要使用 Amazon EC2 启动任意数量的虚拟服务器。利用 Amazon EC2，您可以轻松进行向上扩展或向下扩展，以便应对需求变化。有关更多信息，请参阅什么是 Amazon EC2？ 您可以将服务器与数据库实例放置在同一 VPC 中，并使用 VPC 安全组来访问数据库。有关更多信息，请参阅“在 VPC 中使用数据库实例”。
Web 服务器 (配备 Siebel Web 服务器扩展)	您可以在多个 EC2 实例上安装多个 Web 服务器。然后，您可以使用 Elastic Load Balancing 在实例之间分发传入流量。有关更多信息，请参阅什么是 Elastic Load Balancing？。
Siebel 网关名称服务器	您可以在 EC2 实例上托管 Siebel 网关名称服务器。然后，您可以将服务器与数据库实例放置在同一 VPC 中，并使用 VPC 安全组访问数据库。有关更多信息，请参阅“在 VPC 中使用数据库实例”。

Oracle Database 引擎版本注释

对 Amazon RDS for Oracle 数据库实例的更新，确保为最新版本。如果您应用更新，则可以确信数据库实例正在运行已由 Oracle 和 Amazon 进行测试的数据库软件版本。我们不支持对单独的 RDS for Oracle 数据库实例应用一次性补丁。

当您创建新的数据库实例时，您可以指定目前支持的任何 Oracle Database 版本。您可以指定主要版本（例如 Oracle Database 19c），以及所指定主要版本的任何受支持的次要版本。如果没有指定版本，Amazon RDS 会默认使用支持的版本，通常是最新的版本。如果指定了主要版本但未指定次要版本，Amazon RDS 将默认使用您指定的主要版本的最新版本。要查看受支持版本的列表以及新建数据库实例的默认值，请使用 `describe-db-engine-versions` Amazon CLI 命令。

有关 Amazon RDS 支持的 Oracle Database 版本的详细信息，请参阅 Amazon RDS for Oracle 版本注释。

Amazon RDS for PostgreSQL

Amazon RDS 支持运行多种 PostgreSQL 版本的数据库实例。有关可用版本列表，请参阅可用的 PostgreSQL 数据库版本。

可创建数据库实例和数据库快照、时间点还原和备份。运行 PostgreSQL 的数据库实例支持多可用区部署、只读副本、预置 IOPS，可以在 Virtual Private Cloud (VPC) 内创建此类数据库实例。您还可以使用安全套接字层 (SSL) 连接到运行 PostgreSQL 的数据库实例。

创建数据库实例之前，确保完成 设置 Amazon RDS 环境 中的步骤。

您可以使用任何标准 SQL 客户端应用程序从客户端计算机对实例运行命令。此类应用程序包括 pgAdmin（用于 PostgreSQL 的流行开源管理和开发工具）或 psql（PostgreSQL 安装包含的命令行实用程序）。为了提供托管服务体验，Amazon RDS 未提供数据库实例的主机访问权限。并且它还限制了对需要高级权限的某些系统程序和表的访问。Amazon RDS 支持使用任何标准 SQL 客户端应用程序访问数据库实例上的数据库。Amazon RDS 不允许使用远程登录或安全外壳（SSH）直接从主机进行访问数据库实例。

Amazon RDS for PostgreSQL 符合很多行业标准。例如，您可以使用 Amazon RDS for PostgreSQL 数据库构建符合 HIPAA 标准的应用程序，并存储与医疗保健相关的信息。这包括根据与 Amazon 签订的完整业务合作协议 (BAA) 存储受保护的健康信息 (PHI)。Amazon RDS for PostgreSQL 还满足联邦风险与授权管理项目（FedRAMP）安全要求。在Amazon GovCloud (US)区域中，Amazon RDS for PostgreSQL 获得了 FedRAMP 联合授权委员会（JAB）的 FedRAMP 高基准临时授权操作（P-ATO）。有关支持的合规性标准的更多信息，请参阅Amazon云合规性。

要将 PostgreSQL 数据导入到数据库实例中，请按将数据导入 Amazon RDS 上的 PostgreSQL一节中的信息操作。

> **Important**
>
> 如果您的 RDS for PostgreSQL 数据库实例遇到问题，Amazon 支持座席可能需要有关数据库运行状况的更多信息。目标是确保 Amazon Support 尽快获得所需的信息。
> 您可以使用 PG Collector 帮助在整合的 HTML 文件中收集有价值的数据库信息。有关 PG Collector、如何运行它以及如何下载 HTML 报告的更多信息，请参阅 PG Collector。
> 成功完成后，除非另有说明，否则脚本将以可读的 HTML 格式返回输出。该脚本设计为从 HTML 中排除任何可能危及您业务的数据或安全细节。它也不会对您的数据库或其环境进行任何修改。但是，如果您在 HTML 中发现任何不愿意分享的信息，您可以在上传 HTML 之前随

意删除有问题的信息。当 HTML 的内容可以接受时，请在支持案例的案例详细信息中，使用附件部分进行上传。

主题

Amazon RDS for PostgreSQL 的常见管理任务

下面是对 Amazon RDS for PostgreSQL 数据库实例执行的常见管理任务，以及有关每个任务的相关文档的链接。

任务区域	相关文档
设置 Amazon RDS 供首次使用 创建数据库实例之前，确保完成一些先决条件。例如，默认情况下，数据库实例在创建后带有一个阻止对其进行访问的防火墙。因此，您需要使用正确的 IP 地址和网络配置创建安全组来访问数据库实例。	设置 Amazon RDS 环境
了解 Amazon RDS 数据库实例 如果您出于生产目的而创建数据库实例，则应该了解 Amazon RDS 中的实例类、存储类型和配置的 IOPS 的工作原理。	数据库实例类 Amazon RDS 存储类型 预置 IOPS SSD 存储
查找可用的 PostgreSQL 版本 Amazon RDS 支持 PostgreSQL 的多个版本。	可用的 PostgreSQL 数据库版本
设置高可用性和故障转移支持 生产数据库实例应使用多可用区部署。多可用区部署为数据库实例提供了更高的可用性、数据持久性和容错能力。	配置和管理 Amazon RDS 的多可用区部署
了解 Amazon Virtual Private Cloud (VPC) 网络 如果您的 Amazon 账户具有默认 VPC，则自动在默认 VPC 中创建数据库实例。在某些情况下，您的账户可能没有默认 VPC，并且您可能希望在 VPC 中使用数据库实例。在这些情况下，请在创建数据库实例之前创建 VPC 和子网组。	在 VPC 中使用数据库实例
将数据导入 Amazon RDS PostgreSQL	将数据导入 Amazon RDS 上的 PostgreSQL

任务区域	相关文档
您可以使用多种不同的工具将数据导入 Amazon RDS 中的 PostgreSQL 数据库实例。	
设置只读副本（主和备用） RDS for PostgreSQL 支持来自主实例的同一 Amazon 区域和不同 Amazon 区域的只读副本。	使用数据库实例只读副本 使用 Amazon RDS for PostgreSQL 只读副本 在不同的 Amazon Web Services 区域中创建只读副本
了解安全组 默认情况下，数据库实例在创建时带有一个防火墙，这个防火墙会阻止对实例进行访问。要通过该防火墙提供访问权限，您需要编辑与托管数据库实例的 VPC 关联的 VPC 安全组的入站规则。	使用安全组控制访问权限
设置参数组和特征 要更改数据库实例的默认参数，请创建自定义数据库参数组并将设置更改为该参数组。如果您在创建数据库实例之前执行此操作，则可以在创建实例时选择自定义数据库参数组。	Amazon RDS 的参数组
连接到您的 PostgreSQL 数据库实例 在创建安全组并将其与数据库实例关联后，您可以使用任何标准 SQL 客户端应用程序（如 `psql` 或 `pgAdmin`）连接到数据库实例。	连接到运行 PostgreSQL 数据库引擎的数据库实例 将 SSL 与 PostgreSQL 数据库实例结合使用
备份和还原数据库实例 您可以配置数据库实例以采用自动备份或采用手动快照，然后从备份或快照还原实例。	备份、还原和导出数据
监控数据库实例的活动和性能 您可以使用 CloudWatch Amazon RDS 指标、事件和增强监控来监控 PostgreSQL 数据库实例。	在 Amazon RDS 控制台中查看指标 查看 Amazon RDS 事件

任务区域	相关文档
升级 PostgreSQL 数据库版本 您可以对 PostgreSQL 数据库实例同时执行主要和次要版本升级。	升级 RDS for PostgreSQL 数据库引擎 为 RDS for PostgreSQL 升级选择主要版本
使用日志文件 您可以访问 PostgreSQL 数据库实例的日志文件。	RDS for PostgreSQL 数据库日志文件
了解 PostgreSQL 数据库实例的最佳实践 查找在 Amazon RDS 上使用 PostgreSQL 的一些最佳实践。	使用 PostgreSQL 的最佳实践

以下是本指南中可以帮助您了解和使用 RDS for PostgreSQL 的重要特征的其他章节列表：

- 了解 PostgreSQL 角色和权限
- 控制用户对 PostgreSQL 数据库的访问
- 在 RDS for PostgreSQL 数据库实例上使用参数
- 了解 RDS for PostgreSQL 支持的日志记录机制
- 在 Amazon RDS for PostgreSQL 上使用 PostgreSQL autovacuum
- 将自定义 DNS 服务器用于出站网络访问

使用数据库预览环境

PostgreSQL 社区会不断发布新的 PostgreSQL 版本和扩展，包括测试版。这让 PostgreSQL 用户有机会尽早试用新的 PostgreSQL 版本。要了解有关 PostgreSQL 社区测试版发布流程的更多信息，请参阅 PostgreSQL 文档中的测试版信息。同样，Amazon RDS 将某些 PostgreSQL 测试版作为预览版提供。这允许您使用预览版创建数据库实例，并在数据库预览环境中测试其特征。

数据库预览环境中的 RDS for PostgreSQL 数据库实例在功能上类似于其他 RDS for PostgreSQL 实例。但是，不能将预览版用于生产。

请注意以下重要限制：

- 所有数据库实例在创建之后的 60 天删除，包括其所有备份和快照。
- 您只能在 Virtual Private Cloud (VPC) 中创建基于 Amazon VPC 服务的数据库实例。
- 您只能使用通用 SSD 和预置 IOPS SSD 存储。
- 您无法从 Amazon Support 获得数据库实例的帮助。而是可以将您的问题发布到 Amazon 托管的问答社群，即 Amazon re:Post。
- 您无法将数据库实例的快照复制到生产环境。

预览版支持以下选项：

- 只能使用 M6i、R6i、M6g、M5、T3、R6g 和 R5 实例类型创建数据库实例。有关 RDS 实例类的更多信息，请参阅数据库实例类。
- 您可以同时使用单可用区和多可用区部署。
- 您可以使用标准 PostgreSQL 转储和加载函数从数据库预览环境中导出数据库或将数据库导入数据库预览环境。

主题

数据库预览环境中不支持的功能

以下功能在预览环境中不可用：

- 跨区域快照复制

- 跨区域只读副本

数据库预览环境中的 PostgreSQL 版本 17

> **Note**
>
> 这是 Amazon RDS PostgreSQL 版本 17 的预览文档。本文档随时可能更改。

PostgreSQL 版本 17.0 现可在 Amazon RDS 数据库预览环境中使用。PostgreSQL 版本 17.0 包含一些改进，如以下 PostgreSQL 文档中所述：PostgreSQL 17 Released!

有关数据库预览环境的信息，请参阅the section called "使用数据库预览环境"。要从控制台中访问预览环境，请选择 https://console.aws.amazon.com/rds-preview/。

可用的 PostgreSQL 数据库版本

Amazon RDS 支持运行多种 PostgreSQL 版本的数据库实例。在创建新数据库实例时，您可以指定目前可用的任何 PostgreSQL 版本。您可以指定主要版本（例如 PostgreSQL 14），以及所指定主要版本的任何可用的次要版本。如果没有指定任何版本，Amazon RDS 会默认使用可用的版本，通常是最新的版本。如果指定了主要版本但未指定次要版本，Amazon RDS 将默认使用您指定的主要版本的最新版本。

要查看受支持版本的列表以及新创建的数据库实例的默认值，请使用 `describe-db-engine-versions` Amazon CLI 命令。例如，要显示默认 PostgreSQL 引擎版本，请使用以下命令：

```
aws rds describe-db-engine-versions --default-only --engine postgres
```

有关 Amazon RDS 支持的 PostgreSQL 版本的详细信息，请参阅 Amazon RDS for PostgreSQL 发布说明。您还可以运行 describe-db-major-engine-versions Amazon CLI 命令或者使用 DescribeDBMajorEngineVersions RDS API 操作，来查看有关主要引擎版本的支持日期的信息。

如果您还没有准备好在 RDS 标准支持终止日期之前手动升级到新的主要引擎版本，Amazon RDS 将在 RDS 标准支持终止日期之后自动将您的数据库注册到 Amazon RDS Extended Support。然后，您可以继续运行 RDS for PostgreSQL 11 及更高版本。有关更多信息，请参阅Amazon RDS 的 Amazon RDS 扩展支持和 Amazon RDS 定价。

Amazon RDS for PostgreSQL 已弃用的版本

请注意以下已停用的版本：

- RDS for PostgreSQL 10 已于 2023 年 2 月停用。
- RDS for PostgreSQL 9.6 已于 2022 年 3 月停用。
- RDS for PostgreSQL 9.5 已于 2021 年 3 月停用。

若要了解有关 RDS for PostgreSQL 弃用策略的更多信息，请参阅 Amazon RDS 常见问题。有关 PostgreSQL 版本的更多信息，请参阅 PostgreSQL 文档中的版本控制策略。

了解 RDS for PostgreSQL 增量发行版流程

RDS for PostgreSQL 通过增量发行版提供安全修复、性能改进和新功能，同时保持次要版本兼容性。这些发行版标为 R1、R2、R3，以此类推。

发行版命名约定

- R1 是次要版本的初始发行版。它偶尔会包含新功能、扩展或对现有扩展的升级。
- 后续发行版（R2、R3 及更高版本）包括：
  - 安全更新
  - 性能改进
  - 错误修复
  - 扩展更新

RDS for PostgreSQL 增量发行版流程的优势

增量发行版流程提供以下优势：

- 快速采用新的 PostgreSQL 社区发行版，同时通过后续发行版单独管理 RDS 特定的增强功能。这简化了发行版流程，并确保更快地交付关键更新。
- 访问错误修复、新功能、安全更新和扩展更新，同时保持与 PostgreSQL 次要版本的兼容性。

管理发行版更新

Amazon RDS 通过 Amazon Web Services Management Console中的待处理维护操作来通知您有关新的增量发行版的信息。您可以使用以下方法之一更新数据库：

- 在计划的维护时段内启用自动更新。
- 通过待处理的维护操作手动应用更新。
- 将蓝绿部署与物理复制结合使用，以最大限度地减少停机时间。有关更多信息，请参阅 RDS 蓝绿部署支持 RDS for PostgreSQL 进行次要版本升级。

在更新数据库之前，请考虑以下关键点：

- 除非您将蓝绿部署与物理复制结合使用，否则更新需要重启数据库。
- 某些增量发行版是强制性的，尤其是那些包含安全修复的发行版。

有关更新 Amazon RDS 数据库实例的更多信息，请参阅 PostgreSQL 可信扩展和 apply-pending-maintenance-action。

支持的 PostgreSQL 扩展版本

RDS for PostgreSQL 支持许多 PostgreSQL 扩展。PostgreSQL 社区有时将这些扩展称为模块。扩展针对 PostgreSQL 引擎提供的功能进行了扩展。您可以在 PostgreSQL 版本的默认数据库参数组中找到 Amazon RDS 支持的扩展列表。您还可使用 `psql` 通过显示 `rds.extensions` 参数来查看当前扩展列表，如以下示例中所示。

```
SHOW rds.extensions;
```

> **Note**
>
> 在 `rds.extensions` 中使用 `psql` 参数时，可能不会准确显示在次要版本中添加的参数。

从 RDS for PostgreSQL 13 开始，某些扩展可以由数据库用户（而不是 `rds_superuser`）安装。它们称为可信扩展。要了解更多信息，请参阅 PostgreSQL 可信扩展。

某些版本的 RDS for PostgreSQL 支持 `rds.allowed_extensions` 参数。这个参数允许 `rds_superuser` 限制可以在 RDS for PostgreSQL 数据库实例中安装的扩展。有关更多信息，请参阅限制 PostgreSQL 扩展的安装。

对于每个可用 RDS For PostgreSQL 版本支持的 PostgreSQL 扩展和版本的列表，请参阅《Amazon RDS for PostgreSQL 版本注释》中的 Amazon RDS 上支持的 PostgreSQL 扩展。

限制 PostgreSQL 扩展的安装

您可以限制在 PostgreSQL 数据库实例上安装哪些扩展。默认情况下，不设置此参数，因此如果用户有添加扩展的权限，则可以添加任何受支持的扩展。为此，请将 `rds.allowed_extensions` 参数设置为一串以逗号分隔的扩展名称。通过向此参数添加扩展列表，您可以明确标识 RDS for PostgreSQL 数据库实例可使用的扩展。然后，只有这些扩展才能安装在 PostgreSQL 数据库实例中。

`rds.allowed_extensions` 参数的默认字符串为 '*'，这意味着可以安装引擎版本可用的任何扩展。更改 `rds.allowed_extensions` 参数不需要重新启动数据库，因为它是动态参数。

PostgreSQL 数据库实例引擎必须是以下版本之一才能使用 `rds.allowed_extensions` 参数：

- 所有 PostgreSQL 16 版本
- PostgreSQL 15 及所有更高版本

- PostgreSQL 14 及所有更高版本
- PostgreSQL 13.3 及更高的次要版本
- PostgreSQL 12.7 及更高的次要版本

要查看允许哪些扩展安装，请使用以下 psql 命令。

```
postgres=> SHOW rds.allowed_extensions;
 rds.allowed_extensions
------------------------
 *
```

如果扩展是在其被 `rds.allowed_extensions` 参数列表排除之前安装的，则该扩展仍可以正常使用，并且诸如 `ALTER EXTENSION` 和 `DROP EXTENSION` 之类的命令将继续运行。但是，在扩展被限制后，限制扩展的 `CREATE EXTENSION` 命令将失败。

使用 `CREATE EXTENSION CASCADE` 安装扩展依赖项也受到限制。必须在 `rds.allowed_extensions` 中指定扩展及其依赖项。如果扩展依赖项安装失败，整个 `CREATE EXTENSION CASCADE` 语句将失败。

如果 `rds.allowed_extensions` 参数中未包含扩展，则尝试安装时将看到如下错误。

```
ERROR: permission denied to create extension "extension-name"
HINT: This extension is not specified in "rds.allowed_extensions".
```

PostgreSQL 可信扩展

安装大部分 PostgreSQL 扩展都需要具备 `rds_superuser` 权限。PostgreSQL 13 推出了可信扩展，这些扩展减少了授予普通用户 `rds_superuser` 权限的需求。使用此特征，用户可以安装许多扩展（如果他们具有对当前数据库的 `CREATE` 权限，而不要求具备 `rds_superuser` 角色）。有关详细信息，请参阅 PostgreSQL 文档中的 SQL 创建扩展命令。

下面列出了具有对当前数据库的 `CREATE` 权限，但不需要具备 `rds_superuser` 角色的用户可以安装的扩展：

- bool_plperl
- btree_gin
- btree_gist
- citext

- cube
- dict_int
- fuzzystrmatch
- hstore
- intarray
- isn
- jsonb_plperl
- ltree
- pg_trgm
- pgcrypto
- plperl
- plpgsql
- pltcl
- tablefunc
- tsm_system_rows
- tsm_system_time
- unaccent
- uuid-ossp

对于每个可用 RDS For PostgreSQL 版本支持的 PostgreSQL 扩展和版本的列表，请参阅《Amazon RDS for PostgreSQL 版本注释》中的 Amazon RDS 上支持的 PostgreSQL 扩展。

使用 Amazon RDS for PostgreSQL 支持的 PostgreSQL 功能

Amazon RDS for PostgreSQL 支持很多最常见的 PostgreSQL 功能。例如，PostgreSQL 具有对数据库执行例行维护的 autovacuum 功能。原定设置情况下，autovacuum 功能处于活动状态。尽管您可以禁用此功能，但我们强烈建议您继续启用该功能。了解此功能以及您可以做些什么来确保该功能正常工作，如同任何 DBA 基本任务一样。有关 autovacuum 的更多信息，请参阅 在 Amazon RDS for PostgreSQL 上使用 PostgreSQL autovacuum。要了解有关其他常见 DBA 任务的更多信息，请参阅 Amazon RDS for PostgreSQL 的数据库管理员常见任务。

RDS for PostgreSQL 还支持向数据库实例添加重要功能的扩展。例如，您可以使用 PostGIS 扩展处理空间数据，或者使用 pg_cron 扩展从实例内计划维护。有关 PostgreSQL 扩展的更多信息，请参阅 将 PostgreSQL 扩展与 Amazon RDS for PostgreSQL 结合使用。

外部数据包装器是一种特定类型的扩展，旨在让您的 RDS for PostgreSQL 数据库实例与其他商用数据库或数据类型配合使用。有关 RDS for PostgreSQL 支持的外部数据包装器的更多信息，请参阅 使用 Amazon RDS for PostgreSQL 支持的外部数据包装器。

在下文中，您可以了解有关 RDS for PostgreSQL 支持的一些其他功能的信息。

主题

使用 RDS for PostgreSQL 自定义数据类型和枚举

PostgreSQL 支持创建自定义数据类型和使用枚举。有关创建和使用枚举以及其他数据类型的更多信息，请参阅 PostgreSQL 文档中的枚举类型。

以下是创建一种类型作为枚举然后在表中插入值的示例。

```
CREATE TYPE rainbow AS ENUM ('red', 'orange', 'yellow', 'green', 'blue', 'purple');
```

```
CREATE TYPE
CREATE TABLE t1 (colors rainbow);
CREATE TABLE
INSERT INTO t1 VALUES ('red'), ( 'orange');
INSERT 0 2
SELECT * from t1;
colors
--------
red
orange
(2 rows)
postgres=> ALTER TYPE rainbow RENAME VALUE 'red' TO 'crimson';
ALTER TYPE
postgres=> SELECT * from t1;
colors
---------
crimson
orange
(2 rows)
```

适用于 RDS for PostgreSQL 的事件触发条件

PostgreSQL 的所有当前版本都支持事件触发条件，RDS for PostgreSQL 的所有可用版本也支持事件触发条件。可以使用主用户账户（默认：`postgres`）创建、修改、重命名和删除事件触发条件。事件触发器位于数据库实例级别，因此，它们可应用于实例上的所有数据库。

例如，以下代码创建在每个数据定义语言 (DDL) 命令结束时打印当前用户的事件触发器。

```
CREATE OR REPLACE FUNCTION raise_notice_func()
    RETURNS event_trigger
    LANGUAGE plpgsql AS
$$
BEGIN
    RAISE NOTICE 'In trigger function: %', current_user;
END;
$$;

CREATE EVENT TRIGGER event_trigger_1
    ON ddl_command_end
EXECUTE PROCEDURE raise_notice_func();
```

有关 PostgreSQL 事件触发器的更多信息，请参阅 PostgreSQL 文档中的[事件触发器]。

在 Amazon RDS 上使用 PostgreSQL 事件触发器存在多个限制。这些功能包括：

- 您不能在只读副本上创建事件触发器。不过，您可以在只读副本源上创建事件触发器。然后，事件触发器复制到只读副本。从源推送更改时，不会触发只读副本上的事件触发器。不过，如果提升只读副本，则当数据库发生操作时将触发现有事件触发器。
- 要将主要版本升级到使用事件触发器的 PostgreSQL 数据库实例，请确保在升级该实例之前删除事件触发器。

适用于 RDS for PostgreSQL 的大页

大页是一项内存管理功能，可以减少数据库实例处理大量连续内存数据块（例如共享缓冲区使用的内存数据块）时的开销。RDS for PostgreSQL 的当前所有可用版本支持 PostgreSQL 的这个功能。您可以通过使用对 `mmap` 或 `SYSV` 共享内存的调用，为应用程序分配大页。RDS for PostgreSQL 支持 4 KB 和 2 MB 的页面大小。

您可以通过更改 `huge_pages` 参数的值来启用或禁用大型页面。原定设置情况下，微型、小型和中型数据库实例类以外的所有数据库实例类都会开启此功能。

RDS for PostgreSQL 根据可用的共享内存使用大页。如果由于共享内存约束，数据库实例无法使用大页，Amazon RDS 会阻止数据库实例启动。在这种情况下，Amazon RDS 会将数据库实例的状态设置为不兼容参数状态。在这种情况下，您可以将 `huge_pages` 参数设置为 `off` 以允许 Amazon RDS 启动数据库实例。

`shared_buffers` 参数是设置使用大页所需的共享内存池的关键。`shared_buffers` 参数的原定设置值使用数据库参数宏。此宏设置数据库实例内存可用的总共 8KB 分页的百分比。使用标准大页时，这些页面会位于大页中。如果共享内存参数设置为需要 90% 以上的数据库实例内存，则 Amazon RDS 将数据库实例置于不兼容参数状态。

若要了解有关 PostgreSQL 内存管理的更多信息，请参阅 PostgreSQL 文档中的资源消耗。

为 Amazon RDS for PostgreSQL 执行逻辑复制

从版本 10.4 开始，RDS for PostgreSQL 支持 PostgreSQL 10 中引入的发布和订阅 SQL 语法。要了解更多信息，请参阅 PostgreSQL 文档中的逻辑复制。

> **Note**
>
> 除了 PostgreSQL 10 中引入的原生 PostgreSQL 逻辑复制功能外，RDS for PostgreSQL 还支持 `pglogical` 扩展。有关更多信息，请参阅 使用 pglogical 跨实例同步数据。

在下文中，您可以了解有关为 RDS for PostgreSQL 数据库实例设置逻辑复制的信息。

主题

了解逻辑复制和逻辑解码

RDS for PostgreSQL 支持使用 PostgreSQL 的逻辑复制槽流式传输预写日志 (WAL) 更改。其还支持使用逻辑解码。您可以在实例上设置逻辑复制槽并通过这些槽将数据库更改流式传输到某个客户端（如 `pg_recvlogical`）。您可在数据库级别创建逻辑复制槽，它们支持与单个数据库的复制连接。

PostgreSQL 逻辑复制的最常见客户端是 Amazon Database Migration Service 或 Amazon EC2 实例上的自定义托管主机。逻辑复制槽没有关于流接收器的信息。此外，不要求目标是副本数据库。如果设置逻辑复制槽并且不从该槽进行读取，则数据可写入并快速填满数据库实例上的存储。

您可启用 Amazon RDS 的 PostgreSQL 逻辑复制和逻辑解码，带有参数、复制连接类型和安全角色。逻辑解码的客户端可以是能够与 PostgreSQL 数据库实例上的数据库建立复制连接的任何客户端。

为 RDS for PostgreSQL 数据库实例启用逻辑解码

1. 确保您使用的用户账户具有以下角色：
   - `rds_superuser` 角色，以使您可以启用逻辑复制
   - `rds_replication` 角色，以授予管理逻辑槽并使用逻辑槽流式处理数据的权限
2. 将 `rds.logical_replication` 静态参数设置为 1。在应用该参数时，还将设置参数 `wal_level`、`max_wal_senders`、`max_replication_slots` 和 `max_connections`。这些参数更改可能会增加 WAL 生成，因此，仅在使用逻辑槽时设置 `rds.logical_replication` 参数。
3. 重启数据库实例，静态 `rds.logical_replication` 参数才会生效。
4. 请按下一部分中的说明创建逻辑复制槽。该过程需要您指定解码插件。目前，RDS for PostgreSQL 支持 PostgreSQL 随附的 test_decoding 和 wal2json 输出插件。

有关 PostgreSQL 逻辑解码的更多信息，请参阅 PostgreSQL 文档。

使用逻辑复制槽

您可以通过 SQL 命令来使用逻辑槽。例如，以下命令使用默认的 PostgreSQL 输出插件 `test_slot` 创建一个名为 `test_decoding` 的逻辑槽。

```
SELECT * FROM pg_create_logical_replication_slot('test_slot', 'test_decoding');
slot_name     | xlog_position
-----------------+---------------
regression_slot | 0/16B1970
(1 row)
```

要列出逻辑槽，请使用以下命令。

```
SELECT * FROM pg_replication_slots;
```

要删除逻辑槽，请使用以下命令。

```
SELECT pg_drop_replication_slot('test_slot');
pg_drop_replication_slot
-----------------------
(1 row)
```

有关使用逻辑复制槽的更多示例，请参阅 PostgreSQL 文档中的逻辑解码示例。

创建逻辑复制槽后，就可以开始流式处理。以下示例显示了如何通过流式复制协议控制逻辑解码。此示例使用 PostgreSQL 发行版中包含的程序 pg_recvlogical。此操作要求设置客户端身份验证以允许复制连接。

```
pg_recvlogical -d postgres --slot test_slot -U postgres
    --host -instance-name.111122223333.aws-region.rds.amazonaws.com
    -f -  --start
```

要查看 `pg_replication_origin_status` 视图的内容，请查询 `pg_show_replication_origin_status` 函数。

```
SELECT * FROM pg_show_replication_origin_status();
local_id | external_id | remote_lsn | local_lsn
----------+-------------+------------+-----------
(0 rows)
```

使用逻辑复制来复制表级数据

在 RDS for PostgreSQL 中，可以使用逻辑复制将数据从源表复制到目标表。逻辑复制首先对源表中的现有数据执行初始加载，然后继续复制持续发生的更改。

1. 创建源表

   连接到 RDS for PostgreSQL 数据库实例中的源数据库。

   ```
   source=> CREATE TABLE testtab (slno int primary key);
   CREATE TABLE
   ```

2. 将数据插入到源表中

   ```
   source=> INSERT INTO testtab VALUES (generate_series(1,1000));
   INSERT 0 1000
   ```

3. 为源表创建发布

   - 为源表创建发布：

     ```
     source=> CREATE PUBLICATION testpub FOR TABLE testtab;
     CREATE PUBLICATION
     ```

   - 使用 SELECT 查询来验证已创建的发布的详细信息：

     ```
     source=> SELECT * FROM pg_publication;
       oid   | pubname | pubowner | puballtables | pubinsert | pubupdate | pubdelete |
      pubtruncate | pubviaroot
     --------+---------+----------+--------------+-----------+-----------+-----------
     +-------------+------------
      115069 | testpub |    16395 | f            | t         | t         | t         |
      t           | f
     (1 row)
     ```

- 验证源表已添加到发布中：

```
source=> SELECT * FROM pg_publication_tables;
pubname | schemaname | tablename
---------+------------+-----------
 testpub | public     | testtab
(1 rows)
```

- 要复制数据库中的所有表，请使用：

```
CREATE PUBLICATION testpub FOR ALL TABLES;
```

- 如果已经为单个表创建了发布，并且您需要添加新表，则可以运行下面的查询来将任何新表添加到现有发布中：

```
ALTER PUBLICATION <publication_name> add table <new_table_name>;
```

4. 连接到目标数据库并创建目标表

- 连接到目标数据库实例中的目标数据库。创建与源表同名的目标表：

```
target=> CREATE TABLE testtab (slno int primary key);
CREATE TABLE
```

- 通过对目标表运行 SELECT 查询，确保目标表中不存在数据：

```
target=> SELECT count(*) FROM testtab;
 count
-------
     0
(1 row)
```

5. 在目标数据库中创建并验证订阅

   • 在目标数据库中创建订阅：

```
target=> CREATE SUBSCRIPTION testsub
CONNECTION 'host=<source RDS/host endpoint> port=5432 dbname=<source_db_name>
 user=<user> password=<password>'
PUBLICATION testpub;
NOTICE:  Created replication slot "testsub" on publisher
CREATE SUBSCRIPTION
```

   • 使用 SELECT 查询来验证是否启用了订阅：

```
target=> SELECT oid, subname, subenabled, subslotname, subpublications FROM
 pg_subscription;
  oid  | subname | subenabled | subslotname | subpublications
-------+---------+------------+-------------+-----------------
 16434 | testsub | t          | testsub     | {testpub}
(1 row)
```

   • 创建订阅后，它会将源表中的所有数据加载到目标表。对目标表运行 SELECT 查询以验证初始数据是否已加载：

```
target=> SELECT count(*) FROM testtab;
 count
-------
  1000
(1 row)
```

6. 验证源数据库中的复制槽

   在目标数据库中创建订阅会在源数据库中创建复制槽。通过对源数据库运行以下 SELECT 查询来验证复制槽详细信息：

```
source=> SELECT * FROM pg_replication_slots;
```

```
slot_name |  plugin  | slot_type | datoid | database | temporary | active |
 active_pid | xmin | catalog_xmin | restart_lsn | confirmed_flush_lsn | wal_status
 | safe_wal_size
----------+----------+-----------+--------+----------+-----------+--------
+------------+------+--------------+-------------+---------------------
+------------+---------------
testsub   | pgoutput | logical   | 115048 | source   | f         | t      |
 846 |      |         6945 | 58/B4000568 | 58/B40005A0         | reserved   |
(1 row)
```

7. 测试复制

   - 通过在源表中插入行，测试源表中的数据更改是否复制到目标表：

```
source=> INSERT INTO testtab VALUES(generate_series(1001,2000));
INSERT 0 1000

source=> SELECT count(*) FROM testtab;
 count
-------
  2000
(1 row)
```

   - 验证目标表中的行数，以确认正在复制新的插入内容：

```
target=> SELECT count(*) FROM testtab;
 count
-------
  2000
(1 row)
```

8. 添加表后刷新订阅

   - 在向现有发布添加新表时，务必刷新订阅才能使更改生效：

```
ALTER SUBSCRIPTION <subscription_name> REFRESH PUBLICATION;
```

- 此命令从发布者那里提取缺失的表信息，并开始复制自创建或上次刷新订阅以来添加到已订阅发布中的表。

stats_temp_directory 的 RAM 磁盘

您可使用 RDS for PostgreSQL 参数 `rds.pg_stat_ramdisk_size` 指定分配给用于存储 PostgreSQL `stats_temp_directory` 的 RAM 磁盘系统内存。RAM 磁盘参数仅在 RDS for PostgreSQL 版本 14 及更低版本中可用。

在某些工作负载下，设置该参数可提高性能并降低 I/O 要求。有关 `stats_temp_directory` 的更多信息，请参阅 PostgreSQL 文档。

要为您的 `stats_temp_directory` 设置 RAM 磁盘，可在数据库实例使用的参数组中将 `rds.pg_stat_ramdisk_size` 参数设置为整数值。此参数表示 MB，因此必须使用整数值。表达式、公式和函数对 `rds.pg_stat_ramdisk_size` 参数无效。请务必重启数据库实例，以便更改能够生效。有关设置参数的信息，请参阅 Amazon RDS 的参数组。

例如，以下 Amazon CLI 命令将 RAM 磁盘参数设置为 256 MB。

```
aws rds modify-db-parameter-group \
    --db-parameter-group-name pg-95-ramdisk-testing \
    --parameters "ParameterName=rds.pg_stat_ramdisk_size, ParameterValue=256,
 ApplyMethod=pending-reboot"
```

在重启后，请运行以下命令查看 `stats_temp_directory` 的状态。

```
postgres=> SHOW stats_temp_directory;
```

该命令应返回以下内容。

```
stats_temp_directory
---------------------------
/rdsdbramdisk/pg_stat_tmp
(1 row)
```

适用于 RDS for PostgreSQL 的表空间

为了实现兼容性，RDS for PostgreSQL 支持表空间。由于所有存储都在单个逻辑卷上，因此不能使用表空间进行 I/O 拆分或隔离。我们的基准测试和经验表明，对于大多数使用案例，单个逻辑卷是最佳设置。

若要为 RDS for PostgreSQL 数据库实例创建和使用表空间，需要 `rds_superuser` 角色。RDS for PostgreSQL 数据库实例的主用户账户（默认名称：`postgres`）是此角色的成员。有关更多信息，请参阅 了解 PostgreSQL 角色和权限。

如果在创建表空间时指定文件名，则路径前缀为 `/rdsdbdata/db/base/tablespace`。以下示例将表空间文件放在 `/rdsdbdata/db/base/tablespace/data` 中。此示例假定 dbadmin 用户（角色）存在且已授予其使用表空间所需的 `rds_superuser` 角色。

```
postgres=> CREATE TABLESPACE act_data
  OWNER dbadmin
  LOCATION '/data';
CREATE TABLESPACE
```

若要了解有关 PostgreSQL 表空间的更多信息，请参阅 PostgreSQL 文档中的表空间。

用于 EBCDIC 和其他大型机迁移的 RDS for PostgreSQL 排序规则

RDS for PostgreSQL 版本 10 及更高版本包括 ICU 版本 60.2，该版本基于 Unicode 10.0，并包括来自 Unicode 通用区域设置数据存储库 CLDR 32 的排序规则。这些软件国际化库确保字符编码以一致的方式呈现，无论操作系统或平台如何。有关 Unicode CLDR-32 的更多信息，请参阅 Unicode CLDR 网站上的 CLDR 32 版本注释。您可以通过 ICU 技术委员会（ICU-TC）网站了解有关 Unicode (ICU) 国际化组件的更多信息。有关 ICU-60 的信息，请参阅下载 ICU 60。

从版本 14.3 开始，RDS for PostgreSQL 还包括排序规则，这些排序规则有助于从基于 EBCDIC 的系统进行数据集成和转换。扩展二进制编码的十进制交换码或 EBCDIC 编码通常由大型机操作系统使用。这些 Amazon RDS 提供的排序规则的定义非常狭窄，仅对那些直接映射到 EBCDIC 代码页的 Unicode 字符进行排序。字符按 EBCDIC 代码点顺序排序，以便在转换后进行数据验证。这些排序规则不包括非规范化表单，也不包含不直接映射到源 EBCDIC 代码页上字符的 Unicode 字符。

EBCDIC 代码页和 Unicode 代码点之间的字符映射基于 IBM 发布的表。可以从 IBM 下载整套的压缩文件。RDS for PostgreSQL 使用这些映射以及 ICU 提供的工具来创建本节的各表中列出的排序规则。排序规则名称包括 ICU 要求的语言和国家/地区。但是，EBCDIC 代码页没有指定语言，一些 EBCDIC 代码页涵盖多个国家/地区。这意味着表中排序规则名称的语言和国家/地区部分是任意的，它们不需要

与当前的区域设置相匹配。换句话说，代码页码是该表中排序规则名称的最重要部分。您可以在任何 RDS for PostgreSQL 数据库中使用以下各表列出的任何排序规则。

- Unicode to EBCDIC collations table – 一些大型机数据迁移工具在内部使用 LATIN1 或 LATIN9 来编码和处理数据。此类工具使用往返操作方案来保持数据完整性并支持反向转换。使用 LATIN1 编码处理数据（这种编码不需要特殊处理）的工具可以使用此表中的排序规则。
- Unicode to LATIN9 collations table – 您可以在任何 RDS for PostgreSQL 数据库中使用这些排序规则。

在下表中，您可以找到 RDS for PostgreSQL 中可用的排序规则，这些排序规则将 EBCDIC 代码页映射到 Unicode 代码点。我们建议您将此表中的排序规则用于需要根据 IBM 代码页的顺序进行排序的应用程序开发。

PostgreSQL 排序规则名称	代码页映射和排序顺序的描述
da-DK-cp277-x-icu	直接映射到 IBM EBCDIC 代码页 277（每个转换表）的 Unicode 字符按 IBM CP 277 代码点顺序排序
de-DE-cp273-x-icu	直接映射到 IBM EBCDIC 代码页 273（每个转换表）的 Unicode 字符按 IBM CP 273 代码点顺序排序
en-GB-cp285-x-icu	直接映射到 IBM EBCDIC 代码页 285（每个转换表）的 Unicode 字符按 IBM CP 285 代码点顺序排序
en-US-cp037-x-icu	直接映射到 IBM EBCDIC 代码页 037（每个转换表）的 Unicode 字符按 IBM CP 37 代码点顺序排序
es-ES-cp284-x-icu	直接映射到 IBM EBCDIC 代码页 284（每个转换表）的 Unicode 字符按 IBM CP 284 代码点顺序排序

PostgreSQL 排序规则名称	代码页映射和排序顺序的描述
fi-FI-cp278-x-icu	直接映射到 IBM EBCDIC 代码页 278（每个转换表）的 Unicode 字符按 IBM CP 278 代码点顺序排序
fr-FR-cp297-x-icu	直接映射到 IBM EBCDIC 代码页 297（每个转换表）的 Unicode 字符按 IBM CP 297 代码点顺序排序
it-IT-cp280-x-icu	直接映射到 IBM EBCDIC 代码页 280（每个转换表）的 Unicode 字符按 IBM CP 280 代码点顺序排序
nl-BE-cp500-x-icu	直接映射到 IBM EBCDIC 代码页 500（每个转换表）的 Unicode 字符按 IBM CP 500 代码点顺序排序

Amazon RDS 提供了一组额外的排序规则，这些排序规则使用 IBM 发布的表，按源数据的 EBCDIC 代码页的原始代码点顺序，对映射到 LATIN9 字符的 Unicode 代码点进行排序。

PostgreSQL 排序规则名称	代码页映射和排序顺序的描述
da-DK-cp1142m-x-icu	映射到最初从 IBM EBCDIC 代码页 1142（每个转换表）转换而来的 LATIN9 字符的 Unicode 字符按 IBM CP 1142 代码点顺序排序
de-DE-cp1141m-x-icu	映射到最初从 IBM EBCDIC 代码页 1141（每个转换表）转换而来的 LATIN9 字符的 Unicode 字符按 IBM CP 1141 代码点顺序排序
en-GB-cp1146m-x-icu	映射到最初从 IBM EBCDIC 代码页 1146（每个转换表）转换而来的 LATIN9 字符的 Unicode 字符按 IBM CP 1146 代码点顺序排序

PostgreSQL 排序规则名称	代码页映射和排序顺序的描述
en-US-cp1140m-x-icu	映射到最初从 IBM EBCDIC 代码页 1140（每个转换表）转换而来的 LATIN9 字符的 Unicode 字符按 IBM CP 1140 代码点顺序排序
es-ES-cp1145m-x-icu	映射到最初从 IBM EBCDIC 代码页 1145（每个转换表）转换而来的 LATIN9 字符的 Unicode 字符按 IBM CP 1145 代码点顺序排序
fi-FI-cp1143m-x-icu	映射到最初从 IBM EBCDIC 代码页 1143（每个转换表）转换而来的 LATIN9 字符的 Unicode 字符按 IBM CP 1143 代码点顺序排序
fr-FR-cp1147m-x-icu	映射到最初从 IBM EBCDIC 代码页 1147（每个转换表）转换而来的 LATIN9 字符的 Unicode 字符按 IBM CP 1147 代码点顺序排序
it-IT-cp1144m-x-icu	映射到最初从 IBM EBCDIC 代码页 1144（每个转换表）转换而来的 LATIN9 字符的 Unicode 字符按 IBM CP 1144 代码点顺序排序
nl-BE-cp1148m-x-icu	映射到最初从 IBM EBCDIC 代码页 1148（每个转换表）转换而来的 LATIN9 字符的 Unicode 字符按 IBM CP 1148 代码点顺序排序

在下文中，您可以找到使用 RDS for PostgreSQL 排序规则的示例。

```
db1=> SELECT pg_import_system_collations('pg_catalog');
 pg_import_system_collations
-----------------------------
                           36
db1=> SELECT '¤' < 'a' col1;
 col1
------
 t
db1=> SELECT '¤' < 'a' COLLATE "da-DK-cp277-x-icu" col1;
 col1
------
```

```
f
```

我们建议您将 Unicode to EBCDIC collations table和 Unicode to LATIN9 collations table中的排序规则用于需要根据 IBM 代码页的顺序进行排序的应用程序开发。以下排序规则（后缀为字母“b”）在 `pg_collation` 中也可见，但供 Amazon 中的大型机数据集成和迁移工具使用，这些工具将代码页映射到特定的代码点偏移，并要求在排序规则中进行特殊处理。换句话说，不建议使用以下排序规则。

- da-DK-277b-x-icu
- da-DK-1142b-x-icu
- de-DE-cp273b-x-icu
- de-DE-cp1141b-x-icu
- en-GB-cp1146b-x-icu
- en-GB-cp285b-x-icu
- en-US-cp037b-x-icu
- en-US-cp1140b-x-icu
- es-ES-cp1145b-x-icu
- es-ES-cp284b-x-icu
- fi-FI-cp1143b-x-icu
- fr-FR-cp1147b-x-icu
- fr-FR-cp297b-x-icu
- it-IT-cp1144b-x-icu
- it-IT-cp280b-x-icu
- nl-BE-cp1148b-x-icu
- nl-BE-cp500b-x-icu

要了解有关将应用程序从大型机环境迁移到 Amazon 的更多信息，请参阅什么是 Amazon Mainframe Modernization？

有关在 PostgreSQL 中管理排序规则的更多信息，请参阅 PostgreSQL 文档中的排序规则支持。

管理 RDS for PostgreSQL 的逻辑插槽同步

从社区 PostgreSQL 17 开始，已通过 `sync_replication_slots` 参数或相关的函数 `pg_sync_replication_slots()` 引入了一项新功能（即自动将逻辑复制插槽从主服务器同步到备用服务器），该功能可在执行时手动同步插槽。

从 RDS for PostgreSQL 17 开始提供这些功能。典型的设置将具有主实例及其只读副本，以及主实例的逻辑复制订阅用户。

确保在创建订阅时将失效转移选项设置为 true：

```
CREATE SUBSCRIPTION subname CONNECTION 'host=...' PUBLICATION pubname WITH (failover =
 true);
```

这将在发布者上创建一个启用失效转移的逻辑插槽。

```
postgres=> SELECT slot_name, slot_type, failover FROM pg_catalog.pg_replication_slots;
 slot_name | slot_type | failover
-----------+-----------+----------
 subname   | logical   | t
(1 row)
```

通过启用插槽同步，主实例上的所有失效转移逻辑复制插槽都将在物理备用实例上自动创建并定期同步。确保已通过参数组设置以下值：

- `rds.logical_replication` 必须为 1 才能启用逻辑复制
- 在备用实例上，`hot_standby_feedback` 必须为 1
- 备用实例上的 `rds.logical_slot_sync_dbname` 必须设置为有效的数据库名称

  此参数的默认值为 `postgres`。如果逻辑发布实例有 `postgres` 数据库，则无需更改默认参数。
- 主实例上的 `synchronized_standby_slots` 必须设置为要同步的备用实例的物理复制插槽
- `sync_replication_slots` 必须为 1 才能启用自动同步

使用启用失效转移的订阅插槽和上述参数值，当提升备用实例时，订阅用户可以更改其对这个新提升实例的订阅，并继续无缝地进行逻辑复制。

连接到运行 PostgreSQL 数据库引擎的数据库实例

在 Amazon RDS 预置数据库实例后，您可以使用任何标准 SQL 客户端应用程序连接到该实例。在可以连接之前，数据库实例必须是可用且可访问的。是否可以从 VPC 外部连接到实例取决于您创建 Amazon RDS 数据库实例的方式：

- 如果您将数据库实例创建为公有，VPC 外部的设备和 Amazon EC2 实例便可以连接到您的数据库。
- 如果您将数据库实例创建为私有，只有 Amazon VPC 内的 Amazon EC2 实例和设备可以连接到您的数据库。

若要检查数据库实例是公有还是私有实例，请使用 Amazon Web Services Management Console 查看实例的 Connectivity & security（连接和安全性）选项卡。您可以在 Security（安全性）项下找到"公开访问"值，其中"否"表示私有，"是"表示公有。

要详细了解不同的 Amazon RDS 和 Amazon VPC 配置及其如何影响可访问性，请参阅 在 VPC 中访问数据库实例的场景。

目录

安装 psql 客户端

要从 EC2 实例连接到您的数据库实例，您可以在 EC2 实例上安装 PostgreSQL 客户端。要在 Amazon Linux 2023 上安装 psql 客户端的最新版本，请运行以下命令：

```
sudo dnf install postgresql
```

要在 Amazon Linux 2 上安装 psql 客户端的最新版本，请运行以下命令：

```
sudo yum install -y postgresql
```

要在 Ubuntu 上安装 psql 客户端的最新版本，请运行以下命令：

```
sudo apt install -y postgresql-client
```

查找有关 RDS for PostgreSQL 数据库实例的连接信息

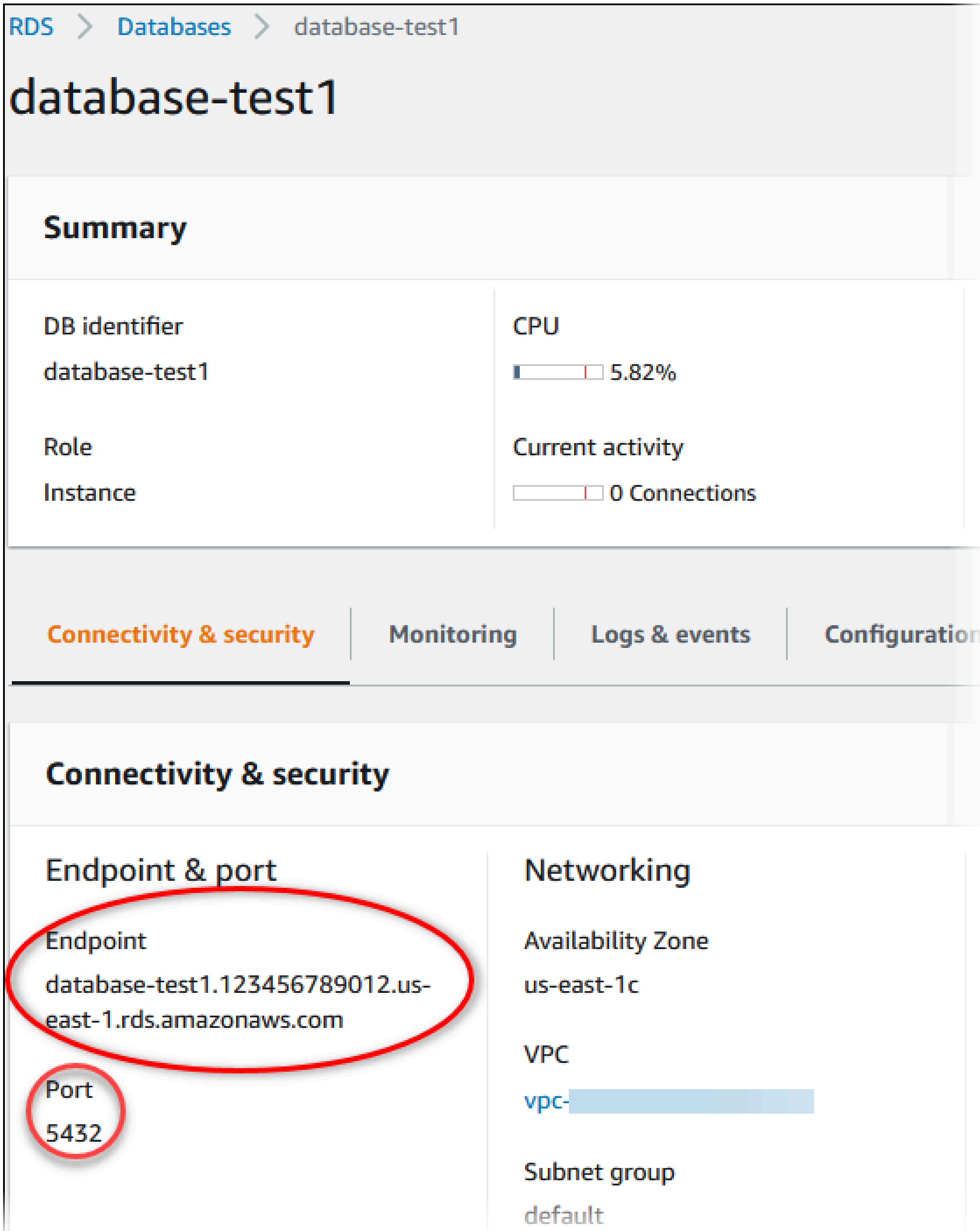
如果数据库实例可用且可访问，则可以通过向 SQL 客户端应用程序提供以下信息进行连接：

- 数据库实例端点，用作实例的主机名（DNS 名称）。
- 数据库实例侦听的端口。对于 PostgreSQL，原定设置端口为 5432。
- 数据库实例的用户名和密码。PostgreSQL 的原定设置“主用户名”是 `postgres`。
- 数据库的名称和密码（数据库名称）。

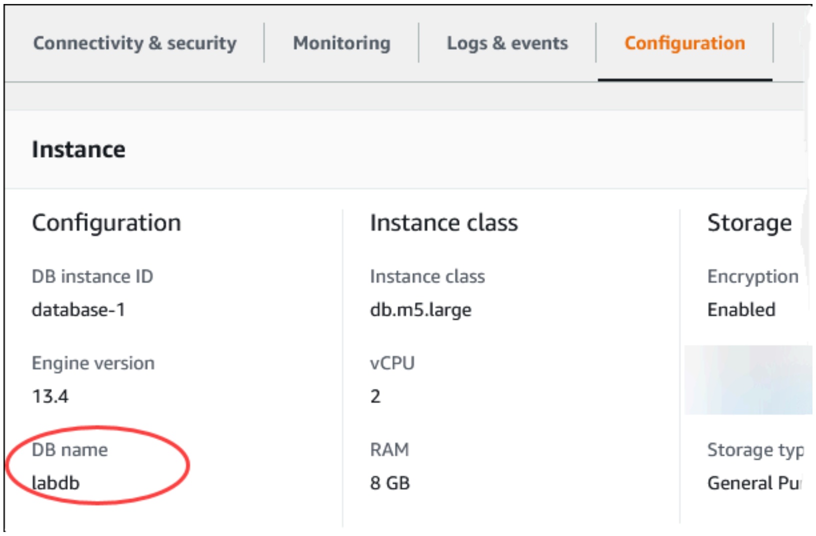
要获取这些详细信息，您可以使用 Amazon Web Services Management Console、Amazon CLI [describe-db-instances](#) 命令或 Amazon RDS API [DescribeDBInstances](#) 操作。

要使用 Amazon Web Services Management Console 查找端点、端口编号和数据库名称

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 打开 RDS 控制台，然后选择 Databases (数据库) 以显示数据库实例的列表。
3. 选择 PostgreSQL 数据库实例名称以显示其详细信息。
4. 在连接和安全性选项卡上，复制端点。另请注意端口号。您需要端点和端口号才能连接到数据库实例。



5. 在 Configuration（配置）选项卡中，记下数据库名称。如果您在创建 RDS for PostgreSQL 实例时创建了数据库，就会看到数据库名称下列出的名称。如果您没有创建数据库，则数据库名称将显示破折号 (-)。



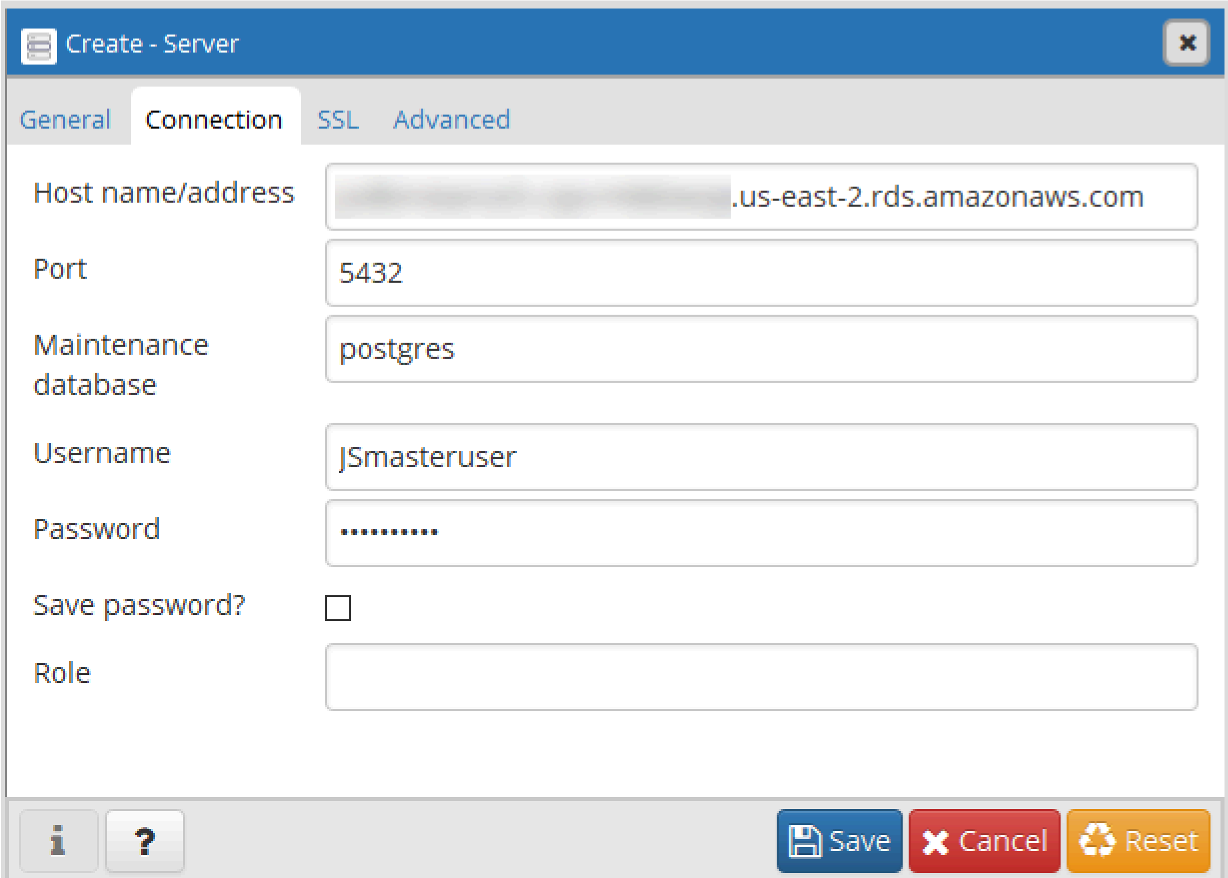
下面是两种连接到 PostgreSQL 数据库实例的方法。第一个示例使用 pgAdmin，这是一个时下受欢迎的用于 PostgreSQL 的开源管理和开发工具。第二个示例使用 psql，这是一个命令行实用工具，包含在 PostgreSQL 安装中。

使用 pgAdmin 连接到 RDS for PostgreSQL 数据库实例

您可以使用开源工具 pgAdmin 连接到 RDS for PostgreSQL 数据库实例。客户端计算机上不必具有 PostgreSQL 的本地实例，即可从 http://www.pgadmin.org/ 下载和使用 pgAdmin。

要使用 pgAdmin 连接到 RDS for PostgreSQL 数据库实例

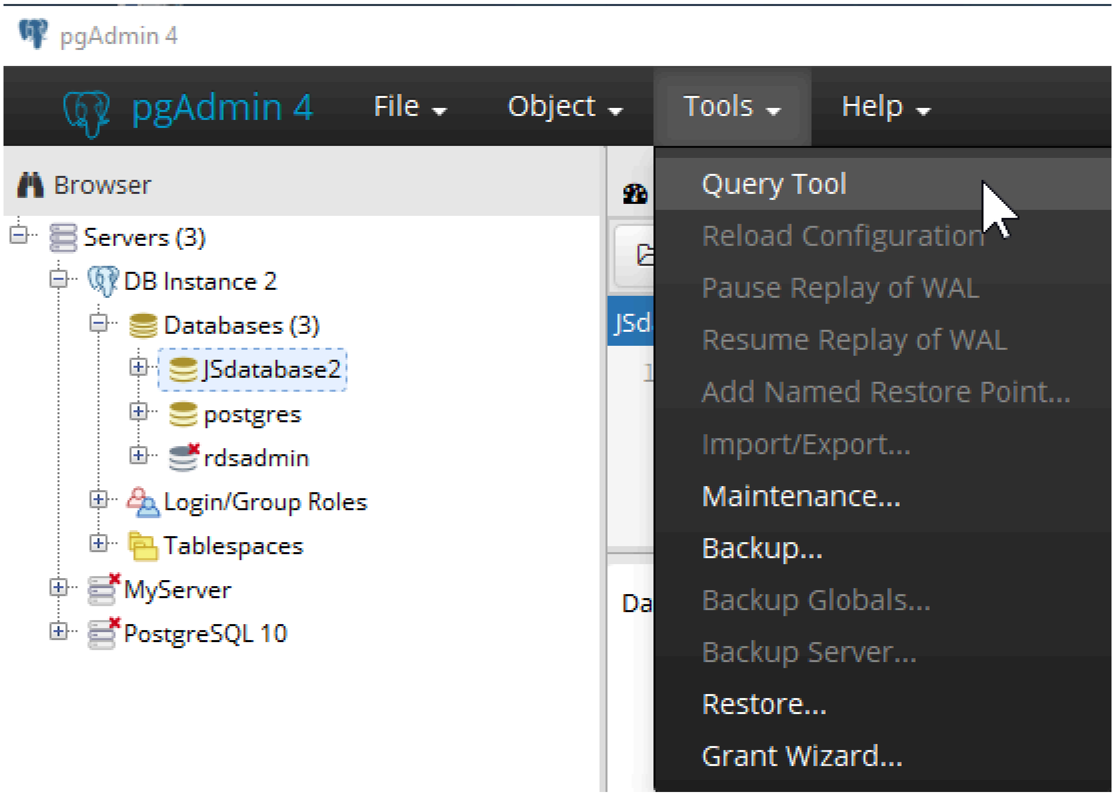
1. 在您的客户端计算机上启动 pgAdmin 应用程序。
2. 在 Dashboard (控制面板) 选项卡上，选择 Add New Server (添加新服务器)。
3. 在 Create - Server (创建 - 服务器) 对话框中，在 General (常规) 选项卡上键入名称以在 pgAdmin 中标识该服务器。
4. 在 Connection (连接) 选项卡上，键入您的数据库实例的以下信息：
   - 对于 Host (主机)，键入端点，例如 `mypostgresql.c6c8dntfzzhgv0.us-east-2.rds.amazonaws.com`。
   - 对于 Port (端口)，键入分配的端口。
   - 对于 Username（用户名），键入您在创建数据库实例时输入的用户名（如果您更改了“主用户名”的默认值，`postgres`）。
   - 对于 Password (密码)，键入您在创建数据库实例时输入的密码。



5. 选择保存。

   如果您有任何连接问题，请参阅排除与您的 RDS for PostgreSQL 实例的连接故障。

6. 要在 pgAdmin 浏览器中访问数据库，请依次展开 Servers (服务器)、数据库实例和 Databases (数据库)。选择数据库实例的数据库名称。



7. 要打开可在其中输入 SQL 命令的面板，请依次选择 Tools (工具) 和 Query Tool (查询工具)。

使用 psql 连接到 RDS for PostgreSQL 数据库实例

可使用 psql 命令行实用工具的本地实例连接到 RDS for PostgreSQL 数据库实例。需要在您的客户端计算机上安装 PostgreSQL 或 psql 客户端。

您可以从 PostgreSQL 网站下载 PostgreSQL 客户端。请按照特定于您的操作系统版本的说明安装 psql。

要使用 psql 连接到 RDS for PostgreSQL 数据库实例，您需要提供主机 (DNS) 信息、访问凭证以及数据库名称。

使用以下格式之一连接到 RDS for PostgreSQL 数据库实例。在连接时，系统将提示您输入密码。对于批处理作业或脚本，请使用 `--no-password` 选项。此选项面向整个会话设置。

> **Note**
>
> 如果服务器需要进行密码身份验证，而密码不能从其他来源获取，`--no-password` 连接尝试则会失败。有关更多信息，请参阅 psql 文档。

如果这是您第一次连接到此数据库实例，或者如果您尚未为此 RDS for PostgreSQL 实例创建数据库，则可以使用"主用户名"和密码连接到 postgres 数据库。

对于 Unix，使用以下格式。

```
psql \
   --host=<DB instance endpoint> \
   --port=<port> \
   --username=<master username> \
   --password \
   --dbname=<database name>
```

对于 Windows，使用以下格式。

```
psql ^
   --host=<DB instance endpoint> ^
   --port=<port> ^
   --username=<master username> ^
   --password ^
   --dbname=<database name>
```

例如，以下命令使用虚构的凭证连接到名为 mypgdb 的 PostgreSQL 数据库实例上名为 `mypostgresql` 的数据库。

```
psql --host=mypostgresql.c6c8mwvfdgv0.us-west-2.rds.amazonaws.com --port=5432 --
username=awsuser --password --dbname=mypgdb
```

使用 Amazon Web Services（Amazon）JDBC 驱动程序连接到 RDS for PostgreSQL

Amazon Web Services（Amazon）JDBC 驱动程序已重新设计为高级 JDBC 包装器。此包装器是对现有 JDBC 驱动程序各项功能的补充和扩展。该驱动程序与社区 pgJDBC 驱动程序兼容。

要安装 Amazon JDBC 驱动程序，请附加 Amazon JDBC 驱动程序.jar 文件（位于应用程序 `CLASSPATH` 中），并保留对相应社区驱动程序的引用。按如下方式更新相应的连接 URL 前缀：

- `jdbc:postgresql://` 到 `jdbc:aws-wrapper:postgresql://`

有关 Amazon JDBC 驱动程序的更多信息及其完整使用说明，请参阅 [Amazon Web Services (Amazon) JDBC Driver GitHub repository]。

使用 Amazon Web Services（Amazon）Python 驱动程序连接到 RDS for PostgreSQL

Amazon Web Services（Amazon）驱动程序设计为高级 Python 包装器。这款包装器是对开源 Psycopg 驱动程序的各项功能的补充和扩展。Amazon Python 驱动程序支持 Python 3.8 及更高版本。您可以使用 `pip` 命令和 `psycopg` 开源软件包安装 `aws-advanced-python-wrapper` 程序包。

有关 Amazon JDBC 驱动程序的更多信息及其完整使用说明，请参阅 [Amazon Web Services（Amazon）Python 驱动程序 GitHub 存储库]。

排除与您的 RDS for PostgreSQL 实例的连接故障

主题

- [错误 – FATAL：数据库名称不存在]
- [错误 – 无法连接到服务器：连接超时]
- [安全组访问规则错误]

错误 – FATAL：数据库*##*不存在

如果在尝试连接时收到 `FATAL: database *name* does not exist` 之类的错误，请尝试对 `--dbname` 选项使用默认数据库名称 postgres。

错误 – 无法连接到服务器：连接超时

如果无法连接到数据库实例，则最常见的错误是 `Could not connect to server: Connection timed out.`如果出现此错误，请检查以下内容：

- 检查所使用的主机名是否是数据库实例端点，以及所使用的端口号是否正确。
- 确保将数据库实例的公共可访问性设置为是，以允许外部连接。要修改公有访问权限共访问设置，请参阅 [修改 Amazon RDS 数据库实例]。
- 确保连接到数据库的用户对数据库具有 CONNECT 访问权限。您可以使用以下查询来提供对数据库的连接访问权限。

```
GRANT CONNECT ON DATABASE database name TO username;
```

- 检查分配给数据库实例的安全组是否具有允许您的连接通过相应防火墙进行访问的规则。例如，如果数据库实例是使用默认端口 5432 创建的，您公司的防火墙规则可能不允许外部公司设备连接到该端口。

要解决此问题，请修改数据库实例以使用不同的端口。另外，确保应用于数据库实例的安全组允许连接到新端口。要修改数据库端口设置，请参阅 修改 Amazon RDS 数据库实例。

- 检查您尝试使用的端口是否已被本地的 PostgreSQL 实例或计算机上运行的其他服务占用。例如，如果您有一个本地 PostgreSQL 数据库运行在同一个端口（默认为 5432），则这可能会让您无法成功连接到 RDS for PostgreSQL 数据库实例。请确保该端口空闲，或者如果可能的话，尝试使用不同的端口号进行连接。
- 另请参阅 安全组访问规则错误。

安全组访问规则错误

到目前为止，最常见的连接问题与分配给数据库实例的安全组的访问规则有关。如果在创建数据库实例时使用的是原定设置安全组，则该安全组可能没有允许您访问该实例的访问规则。

要使连接运行，您在创建数据库实例时分配给它的安全组必须允许访问该数据库实例。例如，如果数据库实例是在 VPC 中创建的，则它必须具有授权连接的 VPC 安全组。检查数据库实例是否是使用安全组创建的，而该安全组没有授权来自运行应用程序的设备或 Amazon EC2 实例的连接。

您可以在安全组中添加或编辑入站规则。对于 Source (源)，选择 My IP (我的 IP)，这允许从浏览器中检测到的 IP 地址访问数据库实例。有关更多信息，请参阅"通过创建安全组提供对 VPC 中的数据库实例的访问"。

或者，如果数据库实例是在 VPC 外部创建的，则它必须具有授权这些连接的数据库安全组。

有关 Amazon RDS 安全组的更多信息，请参阅 使用安全组控制访问权限。

使用 SSL/TLS 保护与 RDS for PostgreSQL 的连接

RDS for PostgreSQL 支持对 PostgreSQL 数据库实例进行安全套接字层 (SSL) 加密。使用 SSL 可加密应用程序与 PostgreSQL 数据库实例之间的 PostgreSQL 连接。您还可强制至 PostgreSQL 数据库实例的所有连接使用 SSL。RDS for PostgreSQL 还支持传输层安全性 (TLS)，这是 SSL 的下一代协议。

要了解有关 Amazon RDS 和数据保护的更多信息，包括使用 SSL/TLS 加密连接，请参阅 Amazon RDS 中的数据保护。

主题

- 将 SSL 与 PostgreSQL 数据库实例结合使用
- 更新应用程序以使用新的 SSL/TLS 证书连接到 PostgreSQL 数据库实例

将 SSL 与 PostgreSQL 数据库实例结合使用

Amazon RDS 支持对 PostgreSQL 数据库实例进行安全套接字层 (SSL) 加密。使用 SSL 可加密应用程序与 PostgreSQL 数据库实例之间的 PostgreSQL 连接。默认情况下，RDS for PostgreSQL 使用并期望所有客户端使用 SSL/TLS 进行连接，但您也可以要求它这样做。RDS for PostgreSQL 支持传输层安全性协议（TLS）版本 1.1、1.2 和 1.3。

有关 SSL 支持和 PostgreSQL 数据库的一般信息，请参阅 PostgreSQL 文档中的 SSL 支持。有关通过 JDBC 使用 SSL 连接的信息，请参阅 PostgreSQL 文档中的配置客户端。

所有Amazon区域均支持对 PostgreSQL 使用 SSL。在创建数据库实例时，Amazon RDS 会为 PostgreSQL 数据库实例创建一个 SSL 证书。如果启用 SSL 证书验证，SSL 证书会将数据库实例终端节点作为 SSL 证书的公用名 (CN) 包含在内以防止欺诈攻击。

主题

- 通过 SSL 连接到 PostgreSQL 数据库实例
- 需要至 PostgreSQL 数据库实例的 SSL 连接
- 确定 SSL 连接状态
- RDS for PostgreSQL 中的 SSL 密码套件

通过 SSL 连接到 PostgreSQL 数据库实例

通过 SSL 连接到 PostgreSQL 数据库实例

1. 下载证书。

   有关下载证书的信息，请参阅 使用 SSL/TLS 加密与数据库实例或集群的连接。
2. 通过 SSL 连接到您的 PostgreSQL 数据库实例。

   使用 SSL 连接时，客户端可以选择是否验证证书链。如果连接参数指定 `sslmode=verify-ca` 或 `sslmode=verify-full`，则客户端要求 RDS CA 证书位于其信任存储中或在连接 URL 中进行引用。此要求是为了验证签署您的数据库证书的证书链。

   当客户端（如 psql 或 JDBC）配置有 SSL 支持时，默认情况下，该客户端会首先尝试使用 SSL 连接到数据库。如果该客户端无法使用 SSL 进行连接，它将恢复为不使用 SSL 进行连接。基于 libpq 的客户端（例如 psql）和 JDBC 所使用的默认 `sslmode` 模式不同。基于 libpq 的客户端和 JDBC 客户端默认使用 `prefer`。

   使用 `sslrootcert` 参数引用证书，例如，`sslrootcert=rds-ssl-ca-cert.pem`。

以下是使用 `psql` 连接到 PostgreSQL 数据库实例的示例，该数据库实例使用 SSL 及证书验证。

```
$ psql "host=db-name.555555555555.ap-southeast-1.rds.amazonaws.com
    port=5432 dbname=testDB user=testuser sslrootcert=rds-ca-rsa2048-g1.pem
 sslmode=verify-full"
```

需要至 PostgreSQL 数据库实例的 SSL 连接

您可以使用 `rds.force_ssl` 参数要求至 PostgreSQL 数据库实例的连接使用 SSL。对于 RDS for PostgreSQL 版本 15 及更高版本，`rds.force_ssl` 参数默认值为 1（开启）。对于所有其它 RDS for PostgreSQL 主要版本 14 及更早版本，此参数的默认值均为 0（关闭）。您可将 `rds.force_ssl` 参数设置为 1 (on) 以要求使用 SSL/TLS 连接到数据库集群。您可将 `rds.force_ssl` 参数设置为 1 (on) 以要求至数据库实例的 SSL 连接。

要更改此参数的值，您需要创建自定义数据库参数组。然后将自定义数据库参数组中的 `rds.force_ssl` 的值更改为 1，以启用此功能。如果您在创建 RDS for PostgreSQL 数据库实例之前准备了自定义数据库参数组，则可以在创建过程中选择它（而不是默认参数组）。如果您在 RDS for PostgreSQL 数据库实例已运行之后执行此操作，则需要重启该实例，以便您的实例使用自定义参数组。有关更多信息，请参阅 Amazon RDS 的参数组。

当 rds.force_ssl 功能在数据库实例上处于活动状态时，不使用 SSL 的连接尝试将被拒绝，并显示以下消息：

```
$ psql -h db-name.555555555555.ap-southeast-1.rds.amazonaws.com port=5432 dbname=testDB
 user=testuser
psql: error: FATAL: no pg_hba.conf entry for host "w.x.y.z", user "testuser", database
 "testDB", SSL off
```

确定 SSL 连接状态

当您连接到数据库实例后，登录横幅中将显示连接的加密状态：

```
Password for user master:
psql (10.3)
SSL connection (cipher: DHE-RSA-AES256-SHA, bits: 256)
Type "help" for help.
postgres=>
```

也可加载 sslinfo 扩展，然后调用 ssl_is_used() 函数以判断是否在使用 SSL。如果连接使用的是 SSL，则此函数将返回 t；否则返回 f。

```
postgres=> CREATE EXTENSION sslinfo;
CREATE EXTENSION
postgres=> SELECT ssl_is_used();
ssl_is_used
---------
t
(1 row)
```

要获取更多详细信息，您可以使用以下查询从 pg_settings 中获取信息：

```
SELECT name as "Parameter name", setting as value, short_desc FROM pg_settings WHERE
 name LIKE '%ssl%';
             Parameter name            |                  value                  |
                  short_desc
----------------------------------------+-----------------------------------------
+-------------------------------------------------------
 ssl                                    | on                                      |
 Enables SSL connections.
 ssl_ca_file                            | /rdsdbdata/rds-metadata/ca-cert.pem     |
 Location of the SSL certificate authority file.
```

```
 ssl_cert_file                          | /rdsdbdata/rds-metadata/server-cert.pem |
 Location of the SSL server certificate file.
 ssl_ciphers                            | HIGH:!aNULL:!3DES                       |
 Sets the list of allowed SSL ciphers.
 ssl_crl_file                           |                                         |
 Location of the SSL certificate revocation list file.
 ssl_dh_params_file                     |                                         |
 Location of the SSL DH parameters file.
 ssl_ecdh_curve                         | prime256v1                              |
 Sets the curve to use for ECDH.
 ssl_key_file                           | /rdsdbdata/rds-metadata/server-key.pem  |
 Location of the SSL server private key file.
 ssl_library                            | OpenSSL                                 |
 Name of the SSL library.
 ssl_max_protocol_version               |                                         |
 Sets the maximum SSL/TLS protocol version to use.
 ssl_min_protocol_version               | TLSv1.2                                 |
 Sets the minimum SSL/TLS protocol version to use.
 ssl_passphrase_command                 |                                         |
 Command to obtain passphrases for SSL.
 ssl_passphrase_command_supports_reload | off                                     |
 Also use ssl_passphrase_command during server reload.
 ssl_prefer_server_ciphers              | on                                      |
 Give priority to server ciphersuite order.
(14 rows)
```

您还可以使用以下查询，按流程、客户端和应用程序收集有关 RDS for PostgreSQL 数据库实例 SSL 使用情况的所有信息：

```
SELECT datname as "Database name", usename as "User name", ssl, client_addr,
 application_name, backend_type
   FROM pg_stat_ssl
   JOIN pg_stat_activity
   ON pg_stat_ssl.pid = pg_stat_activity.pid
   ORDER BY ssl;
 Database name | User name | ssl |  client_addr   |    application_name    |
 backend_type
---------------+-----------+-----+----------------+------------------------
+------------------------------
               |           | f   |                |                        | autovacuum
 launcher
               | rdsadmin  | f   |                |                        | logical
 replication launcher
```

```
                |          | f   |                |                        | background
 writer
                |          | f   |                |                        |
 checkpointer
                |          | f   |                |                        | walwriter
 rdsadmin       | rdsadmin | t   | 127.0.0.1      |                        | client
 backend
 rdsadmin       | rdsadmin | t   | 127.0.0.1      | PostgreSQL JDBC Driver | client
 backend
 postgres       | postgres | t   | 204.246.162.36 | psql                   | client
 backend
(8 rows)
```

要识别用于 SSL 连接的密码，可按如下方式进行查询：

```
postgres=> SELECT ssl_cipher();
ssl_cipher
--------------------
DHE-RSA-AES256-SHA
(1 row)
```

要了解有关 `sslmode` 选项的更多信息，请参阅 PostgreSQL 文档中的数据库连接控制函数。

RDS for PostgreSQL 中的 SSL 密码套件

PostgreSQL 配置参数 ssl_ciphers 指定在使用 TLS 1.2 及更低版本时，与数据库的 SSL 连接所支持的密码套件类别。

在 RDS for PostgreSQL 16 及更高版本中，您可以修改 `ssl_ciphers` 参数，以使用列入许可列表的密码套件中的特定值。这是一个不要求重启数据库实例的动态参数。要查看列入许可列表的密码套件，请使用 Amazon RDS 控制台或以下 CLI Amazon 命令：

```
aws rds describe-db-parameters --db-parameter-group-name <your-parameter-group> --
region <region> --endpoint-url <endpoint-url> --output json | jq '.Parameters[] |
 select(.ParameterName == "ssl_ciphers")'
```

下表列出了支持自定义配置的版本的默认密码套件和许可的密码套件。

PostgreSQL 引擎版本	默认 ssl_cipher 套件值	列入许可列表的自定义 ssl_cipher 套件值
17	HIGH:!aNULL:!3DES	TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
16	HIGH:!aNULL:!3DES	TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
15	HIGH:!aNULL:!3DES	不支持自定义 ssl_cipher

PostgreSQL 引擎版本	默认 ssl_cipher 套件值	列入许可列表的自定义 ssl_cipher 套件值
14	`HIGH:!aNULL:!3DES`	不支持自定义 ssl_cipher
13	`HIGH:!aNULL:!3DES`	不支持自定义 ssl_cipher
12	`HIGH:!aNULL:!3DES`	不支持自定义 ssl_cipher
11.4 及更高的次要版本	`HIGH:MEDIUM:+3DES:!aNULL:!RC4`	不支持自定义 ssl_cipher
11.1、11.2	`HIGH:MEDIUM:+3DES:!aNULL`	不支持自定义 ssl_cipher
10.9 及更高的次要版本	`HIGH:MEDIUM:+3DES:!aNULL:!RC4`	不支持自定义 ssl_cipher
10.7 及更低的次要版本	`HIGH:MEDIUM:+3DES:!aNULL`	不支持自定义 ssl_cipher

要将所有实例连接配置为使用 `TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384` 密码套件，请修改参数组，如以下示例所示：

```
aws rds modify-db-parameter-group --db-parameter-group-name <your-parameter-group> --parameters
 "ParameterName='ssl_ciphers',ParameterValue='TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384',ApplyMet
```

此示例使用 ECDSA 密码，这要求您的实例使用具有椭圆曲线加密（ECC）的证书颁发机构来建立连接。有关 Amazon RDS 提供的证书颁发机构的信息，请参阅证书颁发机构。

您可以通过确定 SSL 连接状态中所述的方法来验证所使用的密码。

根据上下文，密码可能有不同的名称：

- 您可以在参数组中配置的列入许可列表的密码以其 IANA 名称来引用。
- `sslinfo` 和 `psql` 登录横幅使用密码的 OpenSSL 名称来引用这些密码。

默认情况下，RDS for PostgreSQL 16 及更高版本中 `ssl_max_protocol_version` 的值为 TLS v1.3。您必须将此参数的值设置为 TLS v1.2，因为 TLS v1.3 不使用在 `ssl_ciphers` 参数中指定的密码配置。当您将该值设置为 TLS v1.2 时，连接将仅使用您在 `ssl_ciphers` 中定义的密码。

```
aws rds modify-db-parameter-group --db-parameter-group-name <your-parameter-group> --
parameters
 "ParameterName='ssl_max_protocol_version',ParameterValue='TLSv1.2',ApplyMethod=immediate"
```

为确保数据库连接使用 SSL，请在参数组中将 `rds.force_ssl parameter` 设置为 1。有关参数和参数组的更多信息，请参阅 Amazon RDS 的参数组。

更新应用程序以使用新的 SSL/TLS 证书连接到 PostgreSQL 数据库实例

用于安全套接字层或传输层安全 (SSL/TLS) 的证书通常具有设定的生命周期。服务提供商更新其证书颁发机构 (CA) 证书时，客户端必须更新其应用程序才能使用新证书。在下文中，您可以了解有关如何确定客户端应用程序是否使用 SSL/TLS 连接到 Amazon RDS for PostgreSQL 数据库实例的信息。您还可以找到有关如何检查这些应用程序在连接时是否验证服务器证书的信息。

> **Note**
>
> 配置为在 SSL/TLS 连接之前验证服务器证书的客户端应用程序必须在客户端的信任存储中具有有效的 CA 证书。必要时更新客户端信任存储以获取新证书。

更新客户端应用程序信任存储中的 CA 证书后，可以在数据库实例上轮换这些证书。强烈建议在生产环境中实现这些过程之前，先在非生产环境中测试它们。

有关证书轮换的更多信息，请参阅轮换 SSL/TLS 证书。有关下载证书的更多信息，请参阅使用 SSL/TLS 加密与数据库实例或集群的连接。有关对 PostgreSQL 数据库实例使用 SSL/TLS 的信息，请参阅将 SSL 与 PostgreSQL 数据库实例结合使用。

主题

- 确定是否有应用程序使用 SSL 连接到 PostgreSQL 数据库实例
- 确定客户端是否需要证书验证才能连接
- 更新应用程序信任存储
- 对不同类型的应用程序使用 SSL/TLS 连接

确定是否有应用程序使用 SSL 连接到 PostgreSQL 数据库实例

检查数据库实例配置中 rds.force_ssl 参数的值。默认情况下，对于使用 PostgreSQL 版本 15 之前版本的数据库实例，rds.force_ssl 参数设置为 0（关闭）。默认情况下，对于使用 PostgreSQL 版本 15 及更高主要版本的数据库实例，rds.force_ssl 设置为 1（打开）。如果 rds.force_ssl 参数设置为 1（开），则客户端需要使用 SSL/TLS 进行连接。有关参数组的更多信息，请参阅 Amazon RDS 的参数组。

如果您使用的是 RDS PostgreSQL 版本 9.5 或更高的主要版本，并且 rds.force_ssl 未设置为 1（开），请查询 pg_stat_ssl 视图来检查使用 SSL 的连接。例如，以下查询仅返回 SSL 连接和有关使用 SSL 的客户端的信息。

```
SELECT datname, usename, ssl, client_addr
  FROM pg_stat_ssl INNER JOIN pg_stat_activity ON pg_stat_ssl.pid =
 pg_stat_activity.pid
  WHERE ssl is true and usename<>'rdsadmin';
```

只有使用 SSL/TLS 连接的行才显示有关连接的信息。下面是示例输出。

```
 datname  | usename | ssl | client_addr
----------+---------+-----+-------------
 benchdb  | pgadmin | t   | 53.95.6.13
 postgres | pgadmin | t   | 53.95.6.13
(2 rows)
```

此查询仅显示进行查询时有效的连接。没有结果并不表示没有应用程序在使用 SSL 连接。其他 SSL 连接可能在不同的时间建立。

确定客户端是否需要证书验证才能连接

当客户端（如 psql 或 JDBC）配置有 SSL 支持时，默认情况下，该客户端会首先尝试使用 SSL 连接到数据库。如果该客户端无法使用 SSL 进行连接，它将恢复为不使用 SSL 进行连接。用于基于 libpq 的客户端（例如 psql）和 JDBC 的默认 sslmode 模式设置为 prefer。仅当提供 sslrootcert 且 sslmode 设置为 verify-ca 或 verify-full 时，才会验证服务器上的证书。如果证书无效，则会引发错误。

使用 PGSSLROOTCERT 验证具有 PGSSLMODE 环境变量的证书，且 PGSSLMODE 设置为 verify-ca 或 verify-full。

```
PGSSLMODE=verify-full PGSSLROOTCERT=/fullpath/ssl-cert.pem psql -h
 pgdbidentifier.cxxxxxxxx.us-east-2.rds.amazonaws.com -U masteruser -d postgres
```

使用 sslrootcert 参数验证 sslmode 为连接字符串格式的证书，将 sslmode 设置为 verify-ca 或 verify-full 来验证证书。

```
psql "host=pgdbidentifier.cxxxxxxxx.us-east-2.rds.amazonaws.com sslmode=verify-full
 sslrootcert=/full/path/ssl-cert.pem user=masteruser dbname=postgres"
```

例如，在上述情况下，如果您使用无效的根证书，则会在客户端上看到类似于以下内容的错误。

```
psql: SSL error: certificate verify failed
```

更新应用程序信任存储

有关更新 PostgreSQL 应用程序的信任存储的信息，请参阅 PostgreSQL 文档中的使用 SSL 保护 TCP/IP 连接。

有关下载根证书的信息，请参阅 使用 SSL/TLS 加密与数据库实例或集群的连接。

有关导入证书的示例脚本，请参阅 将证书导入信任存储的示例脚本。

> **Note**
>
> 更新信任存储时，除了添加新证书外，还可以保留较旧证书。

对不同类型的应用程序使用 SSL/TLS 连接

下面提供了有关对不同类型应用程序使用 SSL/TLS 连接的信息：

- psql

  通过将选项指定为连接字符串或环境变量，可以从命令行调用客户端。对于 SSL/TLS 连接，相关选项为 sslmode（环境变量 PGSSLMODE）和 sslrootcert（环境变量 PGSSLROOTCERT）。

  有关完整的选项列表，请参阅 PostgreSQL 文档中的参数关键字。有关完整的环境变量列表，请参阅 PostgreSQL 文档中的环境变量。
- pgAdmin

这个基于浏览器的客户端是一个更加用户友好的界面，用于连接到 PostgreSQL 数据库。

有关配置连接的信息，请参阅 pgAdmin 文档。

- JDBC

  JDBC 支持与 Java 应用程序的数据库连接。

  有关使用 JDBC 连接到 PostgreSQL 数据库的一般信息，请参阅 PostgreSQL JDBC 驱动程序文档中的连接到数据库。有关使用 SSL/TLS 进行连接的信息，请参阅 PostgreSQL JDBC 驱动程序文档中的配置客户端。

- Python

  一个常用的连接到 PostgreSQL 数据库的 Python 库是 `psycopg2`。

  有关如何使用 `psycopg2` 的信息，请参阅 psycopg2 文档。有关如何连接到 PostgreSQL 数据库的简短教程，请参阅 Psycopg2 教程。您可以在 psycopg2 模块内容中找到有关连接命令接受的选项的信息。

> **Important**
>
> 在确定了数据库连接使用 SSL/TLS 并更新了应用程序信任存储之后，可以更新数据库以使用 rds-ca-rsa2048-g1 证书。有关说明，请参阅通过修改数据库实例或集群来更新 CA 证书中的步骤 3。

在 Amazon RDS for PostgreSQL 中使用 Kerberos 身份验证

在用户连接到运行 PostgreSQL 的数据库实例时，您可以使用 Kerberos 对用户进行身份验证。为此，请将数据库实例配置为使用 Amazon Directory Service for Microsoft Active Directory 进行 Kerberos 身份验证。Amazon Directory Service for Microsoft Active Directory 也称为 Amazon Managed Microsoft AD。这是 Amazon Directory Service 提供的一项功能。要了解更多信息，请参阅《Amazon Directory Service 管理指南》中的什么是 Amazon Directory Service？

要开始操作，请创建一个 Amazon Managed Microsoft AD 目录来存储用户凭证。然后，将 Active Directory 的域和其他信息提供给 PostgreSQL 数据库实例。当用户使用 PostgreSQL 数据库实例进行身份验证时，身份验证请求将转发到 Amazon Managed Microsoft AD 目录。

将所有凭证保存在同一目录中可以节省您的时间和精力。您具有一个集中位置用于存储和管理多个数据库实例的凭证。使用目录还可以改善您的整体安全概要。

此外，还可以从自己的本地 Microsoft Active Directory 访问凭证。为此，请创建一个信任域关系，以便 Amazon Managed Microsoft AD 目录信任您的本地 Microsoft Active Directory。通过这种方式，您的用户可以使用 Windows 单点登录 (SSO) 访问 PostgreSQL 实例，获得与访问本地网络中的工作负载相同的体验。

数据库可以使用密码身份验证，也可以使用带有 Kerberos 或 Amazon Identity and Access Management（IAM）身份验证的密码身份验证。有关 IAM 身份验证的更多信息，请参阅 适用于 MariaDB、MySQL 和 PostgreSQL 的 IAM 数据库身份验证。

主题

区域和版本可用性

功能可用性和支持因每个数据库引擎的特定版本以及 Amazon Web Services 区域而异。有关使用 Kerberos 身份验证的 RDS for PostgreSQL 的版本和区域可用性的更多信息，请参阅支持 Amazon RDS 中 Kerberos 身份验证功能的区域和数据库引擎。

PostgreSQL 数据库实例的 Kerberos 身份验证概述

要为 PostgreSQL 数据库实例设置 Kerberos 身份验证，请执行以下步骤，稍后将详细介绍这些步骤：

1. 使用 Amazon Managed Microsoft AD 创建 Amazon Managed Microsoft AD 目录。您可以使用 Amazon Web Services Management Console、Amazon CLI 或 Amazon Directory Service API 创建目录。确保打开目录安全组上的相关出站端口，以便目录可以与实例进行通信。
2. 创建一个角色，以提供对您的 Amazon Managed Microsoft AD 目录进行调用的 Amazon RDS 访问权限。为此，请创建使用托管 IAM 策略 `AmazonRDSDirectoryServiceAccess` 的 Amazon Identity and Access Management (IAM) 角色。

   要使 IAM 角色允许访问，必须在您的 Amazon 账户的正确 Amazon 区域中激活 Amazon Security Token Service（Amazon STS）端点。原定设置情况下，Amazon STS 端点在所有 Amazon Web Services 区域中处于活动状态，您可以直接使用这些端点，而无需执行任何其他操作。有关更多信息，请参阅 IAM 用户指南中的在 Amazon STS 区域中激活和停用 Amazon。
3. 使用 Microsoft Active Directory 工具在 Amazon Managed Microsoft AD 目录中创建和配置用户。有关在 Active Directory 中创建用户的更多信息，请参阅 Amazon 管理指南中的在 Amazon Directory Service 托管 Microsoft AD 中管理用户和组。
4. 如果您计划在不同的 Amazon 账户或 Virtual Private Cloud (VPC) 中查找目录和数据库实例，请配置 VPC 对等连接。有关更多信息，请参阅 Amazon VPC Peering Guide 中的什么是 VPC 对等连接？。
5. 使用以下方法之一，从控制台、CLI 或 RDS API 中创建或修改 PostgreSQL 数据库实例：
   - 创建 Amazon RDS 数据库实例
   - 修改 Amazon RDS 数据库实例
   - 还原到数据库实例
   - 将 Amazon RDS 的数据库实例还原到指定时间

   您可以在与目录相同的 Amazon Virtual Private Cloud (VPC) 中或在不同的 Amazon 账户或 VPC 中定位实例。创建或修改 PostgreSQL 数据库实例时，请执行以下操作：
   - 请提供创建目录时生成的域标识符（`d-*` 标识符）。
   - 还要提供您创建的 IAM 角色的名称。
   - 确保数据库实例安全组可以从目录安全组接收入站流量。
6. 使用 RDS 主用户凭证连接到 PostgreSQL 数据库实例。在 PostgreSQL 中创建用户以在外部进行标识。外部标识的用户可以使用 Kerberos 身份验证登录到 PostgreSQL 数据库实例。

为 PostgreSQL 数据库实例设置 Kerberos 身份验证

您可以使用 Amazon Directory Service for Microsoft Active Directory (Amazon Managed Microsoft AD) 为 PostgreSQL 数据库实例设置 Kerberos 身份验证。要设置 Kerberos 身份验证，请执行以下步骤。

主题

步骤 1：使用 Amazon Managed Microsoft AD 创建目录

Amazon Directory Service 将在 Amazon 云中创建一个完全托管的 Active Directory。在创建 Amazon Managed Microsoft AD 目录时，Amazon Directory Service 将创建两个域控制器和 DNS 服务器。目录服务器在 VPC 中的不同子网中创建。这种冗余有助于确保始终可以访问目录，即使发生了故障。

创建 Amazon Managed Microsoft AD 目录时，Amazon Directory Service 将代表您执行下列任务：

- 在您的 VPC 中设置 Active Directory。
- 创建具有用户名 `Admin` 和指定密码的目录管理员账户。您可以使用此账户管理您的目录。

> **Important**
>
> 请务必保存此密码。Amazon Directory Service 不会存储此密码并且它无法取回或重置。

- 为目录控制器创建安全组。安全组必须允许与 PostgreSQL 数据库实例进行通信。

在启动 Amazon Directory Service for Microsoft Active Directory 时，Amazon 创建一个组织单位 (OU)，其中包含目录的所有对象。此 OU（具有您在创建目录时输入的 NetBIOS 名称）位于域根目录中。此域根目录由 Amazon 拥有和管理。

使用 Admin 目录创建的 Amazon Managed Microsoft AD 账户有权为您的 OU 执行最常见的管理活动：

- 创建、更新或删除用户
- 将资源添加到域（如文件或打印服务器），然后为 OU 中的用户分配这些资源的权限
- 创建额外的 OU 和容器
- 委托授权
- 从 Active Directory 回收站还原删除的对象
- 在 Active Directory Web 服务上为 Windows PowerShell 运行 Active Directory 和域名服务 (DNS) 模块

Admin 账户还有权执行以下域范围的活动：

- 管理 DNS 配置（添加、删除或更新记录、区域和转发器）
- 查看 DNS 事件日志
- 查看安全事件日志

使用 Amazon Managed Microsoft AD 创建目录

1. 在 Amazon Directory Service 控制台导航窗格中，选择 Directories (目录)，然后选择 Set up directory (设置目录)。
2. 选择 Amazon Managed Microsoft AD。Amazon Managed Microsoft AD 是当前唯一支持用于 Amazon RDS 的选项。
3. 选择下一步。
4. 在输入目录信息页面上，提供以下信息：

   版本

      选择符合您要求的版本。

   目录 DNS 名称

      目录的完全限定名称，例如 **corp.example.com**。

   目录 NetBIOS 名称

      可选的目录短名称，如 CORP。

目录描述

目录的可选描述。

管理员密码

目录管理员的密码。目录创建过程将创建一个具有 `Admin` 用户名和此密码的管理员账户。

目录管理员密码不能包含单词“admin”。此密码区分大小写，且长度必须介于 8 – 64 个字符之间。至少，它还必须包含下列四种类别中三种类别的一个字符：

- 小写字母 (a–z)
- 大写字母 (A–Z)
- 数字 (0–9)
- 非字母数字字符 (~!@#$%^&*_-+=`|\(){}[]:;"'<>,.?/)

确认密码

重新键入管理员密码。

> **Important**
> 确保您保存该密码。Amazon Directory Service 不会存储该密码，也无法检索或重置该密码。

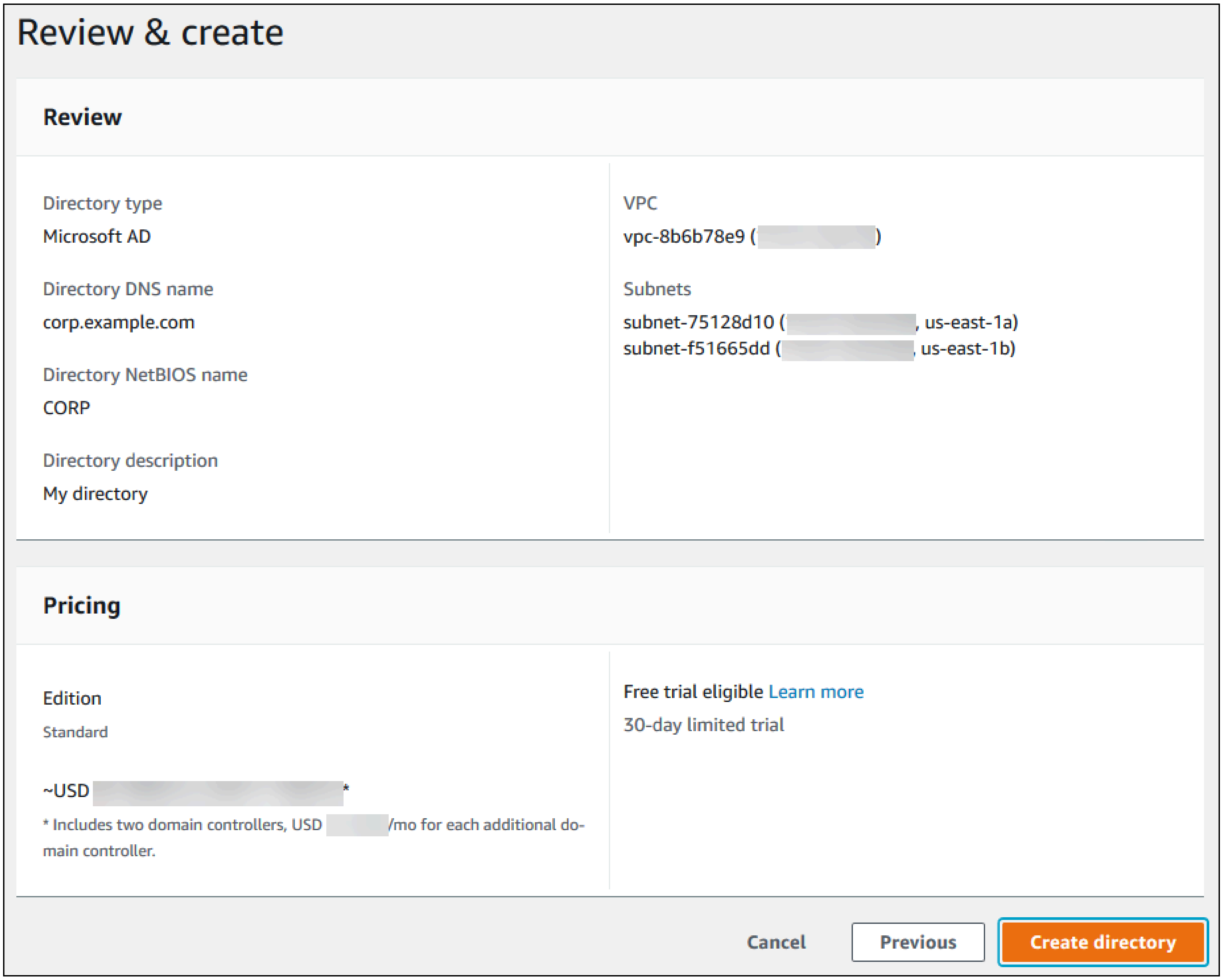
5. 选择下一步。
6. 在选择 VPC 和子网页面上，提供以下信息：

   VPC

   为目录选择 VPC。您可以在该相同 VPC 或不同的 VPC 中创建 PostgreSQL 数据库实例。

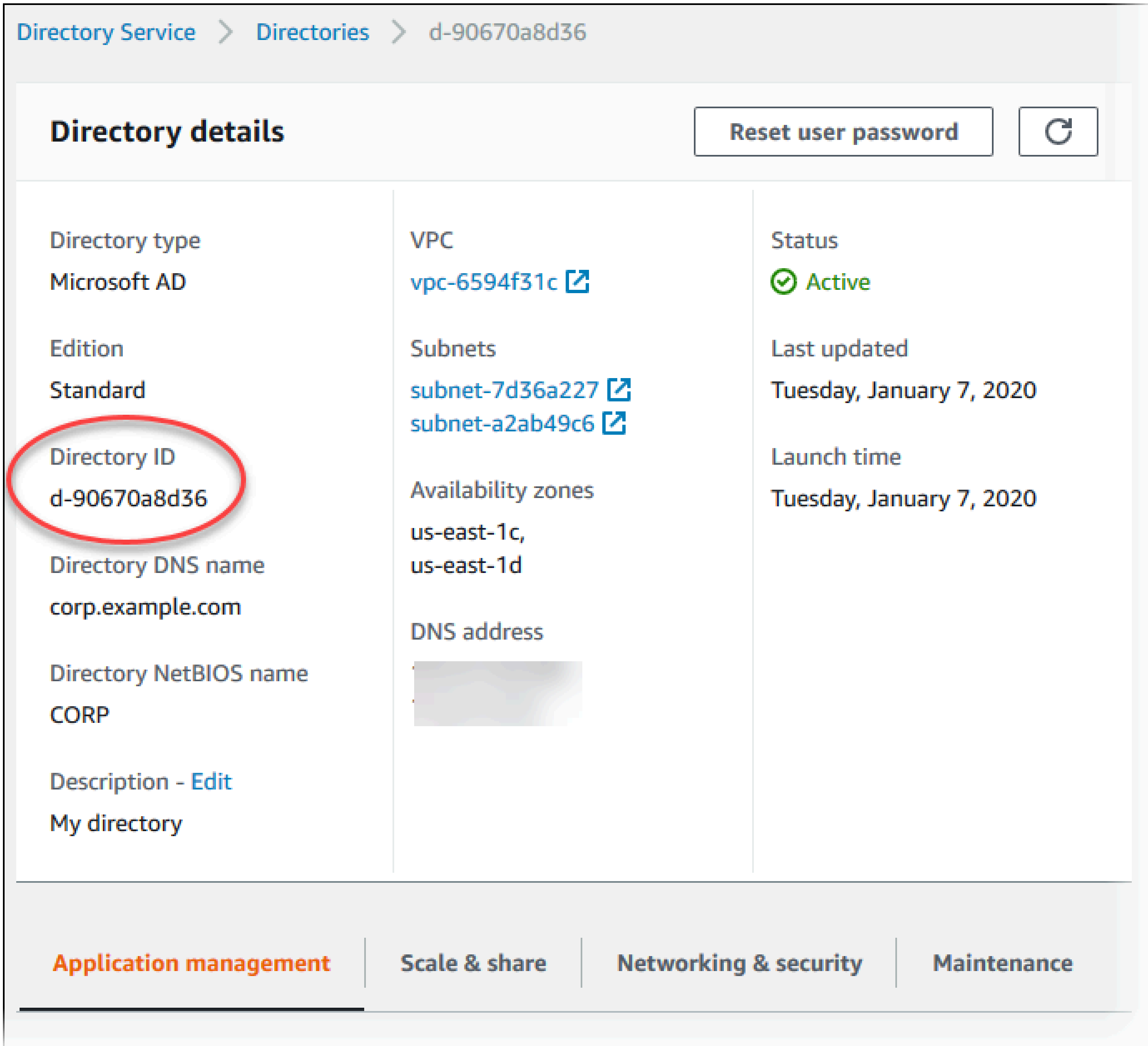
   子网

   为目录服务器选择子网。两个子网必须位于不同的可用区。
7. 选择下一步。
8. 检查目录信息。如果需要进行更改，请选择 Previous (上一步) 并作出更改。如果信息正确，请选择 Create directory (创建目录)。



目录创建需要几分钟时间。创建成功后，Status (状态) 值将更改为 Active (活动)。

要查看有关您的目录的信息，请在目录列表中选择目录。记下目录 ID 值。在创建或修改 PostgreSQL 数据库实例时，您需要使用该值。



步骤 2：（可选）在本地 Active Directory 和 Amazon Directory Service 之间创建信任关系

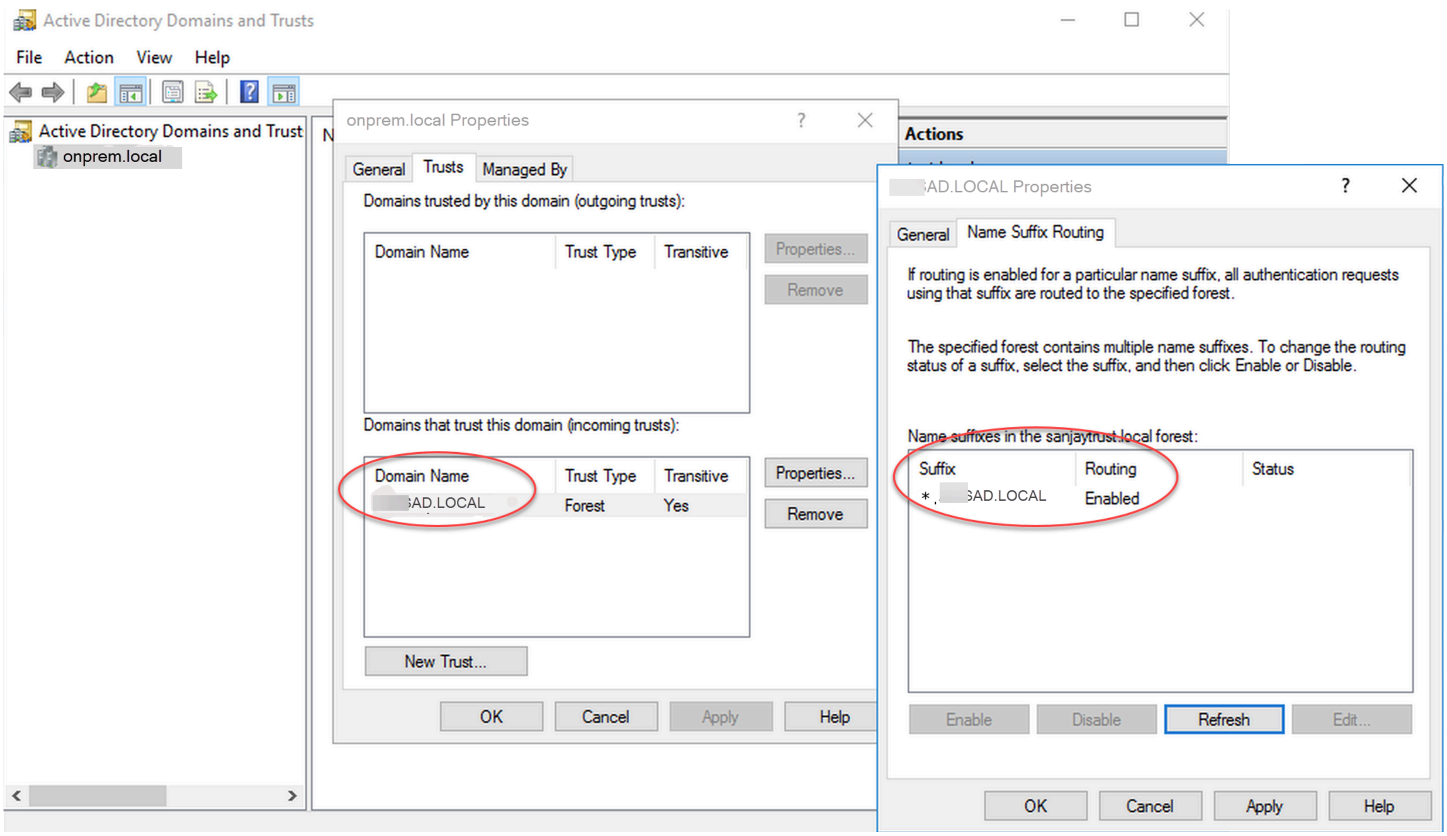
如果您不打算使用自己的本地 Microsoft Active Directory，请跳转至 步骤 3：为 Amazon RDS 创建 IAM 角色以访问 Amazon Directory Service。

要使用本地 Active Directory 获取 Kerberos 身份验证，您需要使用林信任在本地 Microsoft Active Directory 和 Amazon Managed Microsoft AD 目录（在 步骤 1：使用 Amazon Managed Microsoft AD 创建目录 中创建）之间创建信任域关系。信任可以是单向的，此时 Amazon Managed Microsoft AD 目录信任本地 Microsoft Active Directory。信任也可以是双向的，此时两个 Active Directory 相互信任。

有关使用 Amazon Directory Service 设置信任的更多信息，请参阅 Amazon Directory Service 管理指南中的[何时创建信任关系](#)。

> **Note**
>
> 如果您使用本地 Microsoft Active Directory，则 Windows 客户端使用端点中 Amazon Directory Service 的域名而不是 rds.amazonaws.com 进行连接。要了解更多信息，请参阅[使用 Kerberos 身份验证连接到 PostgreSQL](#)。

请确保您的本地 Microsoft Active Directory 域名包含与新创建的信任关系对应的 DNS 后缀路由。以下屏幕截图显示一个示例。



步骤 3：为 Amazon RDS 创建 IAM 角色以访问 Amazon Directory Service

要使 Amazon RDS 为您调用 Amazon Directory Service，您的 Amazon 账户需要一个使用托管式 IAM 策略 `AmazonRDSDirectoryServiceAccess` 的 IAM 角色。该角色允许 Amazon RDS 对 Amazon Directory Service 进行调用。

当您使用 Amazon Web Services Management Console 创建数据库实例并且控制台用户账户具有 `iam:CreateRole` 权限时，控制台将自动创建所需的 IAM 角色。在这种情况下，

角色名为 rds-directoryservice-kerberos-access-role。否则，您必须手动创建 IAM 角色。在创建该 IAM 角色时，请选择 Directory Service，然后将 Amazon 托管策略 AmazonRDSDirectoryServiceAccess 附加到该角色。

有关为服务创建 IAM 角色的更多信息，请参阅 IAM 用户指南中的创建向Amazon服务委托权限的角色。

> **Note**
>
> 用于 RDS for Microsoft SQL Server 的 Windows 身份验证的 IAM 角色不能用于 Amazon RDS for PostgreSQL。

作为使用 AmazonRDSDirectoryServiceAccess 托管式策略的替代方法，您可以创建具有所需权限的策略。在这种情况下，IAM 角色必须具有以下 IAM 信任策略。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "directoryservice.rds.amazonaws.com",
          "rds.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

角色还必须具有以下 IAM 角色策略。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "ds:DescribeDirectories",
```

```
        "ds:AuthorizeApplication",
        "ds:UnauthorizeApplication",
        "ds:GetAuthorizedApplicationDetails"
      ],
    "Effect": "Allow",
    "Resource": "*"
    }
  ]
}
```

对于选择加入的 Amazon Web Services 区域，请在 IAM 角色信任策略中使用区域特定的服务主体。在为这些区域中的服务创建信任策略时，请在服务主体中指定区域代码。

以下示例显示了包括区域特定的服务主体的信任策略：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "directoryservice.rds.REGION-CODE.amazonaws.com",
          "rds.REGION-CODE.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

将 REGION-CODE 替换为特定区域的代码。例如，对亚太地区（墨尔本）区域使用以下服务主体：

```
"Service": [
  "directoryservice.rds.ap-southeast-4.amazonaws.com",
  "rds.ap-southeast-4.amazonaws.com"
]
```

步骤 4：创建和配置用户

您可以使用“Active Directory 用户和计算机”工具创建用户。该工具是 Active Directory Domain Services 和 Active Directory Lightweight Directory Services 工具之一。有关更多信息，请参阅 Microsoft 文档中的将用户和计算机添加到 Active Directory 域。在这种情况下，用户是个人或其他实体，例如属于域的计算机，其身份在目录中维护。

要在 Amazon Directory Service 目录中创建用户，您必须连接到属于 Amazon Directory Service 目录成员的基于 Windows 的 Amazon EC2 实例。同时，您必须以具有创建用户权限的用户身份登录。有关更多信息，请参阅 Amazon Directory Service 管理指南中的创建用户。

步骤 5：在目录和数据库实例之间启用跨 VPC 流量

如果您打算将目录和数据库实例放在同一 VPC 中，请跳过该步骤，然后转到 步骤 6：创建或修改 PostgreSQL 数据库实例。

如果您计划在不同 VPC 中查找目录和数据库实例，请使用 VPC 对等连接或 Amazon Transit Gateway 配置跨 VPC 流量。

以下过程使用 VPC 对等连接启用 VPC 之间的流量。请按照 Amazon Virtual Private Cloud 对等连接指南中的什么是 VPC 对等连接？操作。

使用 VPC 对等连接启用跨 VPC 流量

1. 设置适合的 VPC 路由规则，以便确保网络流量可以双向流动。
2. 确保数据库实例安全组可以从目录安全组接收入站流量。
3. 确保没有任何网络访问控制列表 (ACL) 规则会阻止流量。

如果该目录由不同的 Amazon 账户拥有，则您必须共享该目录。

在 Amazon 账户之间共享目录

1. 按照 Amazon 管理指南中的教程：共享 Amazon 托管 Microsoft AD 目录以实现无缝 EC2 域加入中的说明，开始与将要在其中创建数据库实例的 Amazon Directory Service 账户共享目录。
2. 使用数据库实例的账户登录到 Amazon Directory Service 控制台，并确保在处理之前域具有 `SHARED` 状态。
3. 使用数据库实例的账户登录 Amazon Directory Service 控制台时，请记录目录 ID 值。您可以使用此目录 ID 将数据库实例加入域。

步骤 6：创建或修改 PostgreSQL 数据库实例

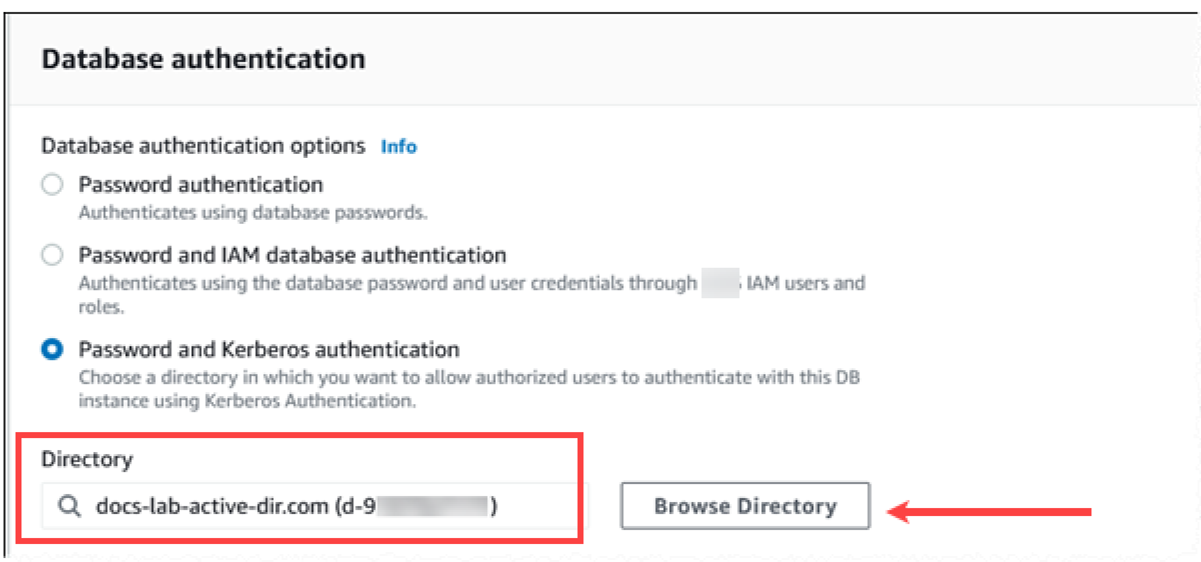
创建或修改 PostgreSQL 数据库实例以用于您的目录。您可以使用控制台、CLI 或 RDS API 将数据库实例与目录关联。您可以通过下列方式之一来执行该操作：

- 使用控制台、create-db-instance CLI 命令或 CreateDBInstance RDS API 操作创建新的 PostgreSQL 数据库实例。有关说明，请参阅 创建 Amazon RDS 数据库实例。
- 使用控制台、modify-db-instance CLI 命令或 ModifyDBInstance RDS API 操作修改现有的 PostgreSQL 数据库实例。有关说明，请参阅 修改 Amazon RDS 数据库实例。
- 使用控制台、restore-db-instance-from-db-snapshot CLI 命令或 RestoreDBInstanceFromDBSnapshot RDS API 操作，从数据库快照还原 PostgreSQL 数据库实例。有关说明，请参阅 还原到数据库实例。
- 使用控制台、restore-db-instance-to-point-in-time CLI 命令或 RestoreDBInstanceToPointInTime RDS API 操作，将 PostgreSQL 数据库实例还原到某个时间点。有关说明，请参阅 将 Amazon RDS 的数据库实例还原到指定时间。

仅 VPC 中的 PostgreSQL 数据库实例支持 Kerberos 身份验证。数据库实例可以与目录在同一 VPC 中或在不同 VPC 中。数据库实例必须使用允许在目录的 VPC 中传入和传出的安全组，以便数据库实例与目录通信。

控制台

在使用控制台创建、修改或还原数据库实例时，请选择数据库身份验证部分中的密码和 Kerberos 身份验证。然后选择浏览目录。选择目录或选择创建一个新目录以使用 Directory Service。



Amazon CLI

使用 Amazon CLI 时，数据库实例需要以下参数才能使用您创建的目录：

- 对于 `--domain` 参数，请使用创建目录时生成的域标识符（“d-*”标识符）。
- 对于 `--domain-iam-role-name` 参数，请使用您使用托管 IAM 策略 `AmazonRDSDirectoryServiceAccess` 创建的角色。

例如，以下 CLI 命令会修改数据库实例以使用目录。

```
aws rds modify-db-instance --db-instance-identifier mydbinstance --domain d-Directory-ID --domain-iam-role-name role-name
```

> **Important**
>
> 如果修改数据库实例以启用 Kerberos 身份验证，请在做出更改之后重新启动数据库实例。

步骤 7：为您的 Kerberos 主体创建 PostgreSQL 用户

此时，您的 RDS for PostgreSQL 数据库实例已加入 Amazon Managed Microsoft AD 域。您需要将您在 步骤 4：创建和配置用户 的目录中创建的用户设置为 PostgreSQL 数据库用户，并授予登录数据库的权限。为此，请以具有 `rds_superuser` 权限的数据库用户身份登录。例如，如果在创建 RDS for PostgreSQL 数据库实例时接受了缺省设置值，请使用 `postgres`，如以下步骤所示。

为 Kerberos 主体创建 PostgreSQL 数据库用户

1. 使用 `psql` 通过 `psql` 连接到 RDS for PostgreSQL 数据库实例端点。以下示例对于 `rds_superuser` 角色使用原定设置 `postgres` 账户。

   ```
   psql --host=cluster-instance-1.111122223333.aws-region.rds.amazonaws.com --port=5432 --username=postgres --password
   ```

2. 为要访问数据库的每个 Kerberos 主体（Active Directory 用户名）创建一个数据库用户名。使用 Active Directory 实例中定义的规范用户名（身份），即该用户名的小写 `alias`（Active Directory 中的用户名）和大写 Active Directory 域名。Active Directory 用户名是经过外部身份验证的用户，因此请在名称前后使用引号，如下所示。

   ```
   postgres=> CREATE USER "username@CORP.EXAMPLE.COM" WITH LOGIN;
   ```

```
CREATE ROLE
```

3. 将 `rds_ad` 角色授予数据库用户。

```
postgres=> GRANT rds_ad TO "username@CORP.EXAMPLE.COM";
GRANT ROLE
```

为 Active Directory 用户身份创建完所有 PostgreSQL 用户后，用户可以使用他们的 Kerberos 凭证访问 RDS for PostgreSQL 数据库实例。

使用 Kerberos 进行身份验证的数据库用户需要从属于 Active Directory 域成员的客户端计算机进行身份验证。

被授予 `rds_ad` 角色的数据库用户不能同时拥有 `rds_iam` 角色。这也适用于嵌套成员资格。有关更多信息，请参阅 适用于 MariaDB、MySQL 和 PostgreSQL 的 IAM 数据库身份验证。

步骤 8：配置 PostgreSQL 客户端

要配置 PostgreSQL 客户端，请采取以下步骤：

- 创建一个 krb5.conf 文件（或等效的文件）以指向域。
- 验证流量是否可以在客户端主机和 Amazon Directory Service 之间流动。使用网络实用程序（如 Netcat）执行以下操作：
  - 验证端口 53 上通过 DNS 的流量。
  - 验证端口 53 上通过 TCP/UDP 的流量以及 Kerberos 的流量，包括用于 Amazon Directory Service 的端口 88 和 464。
- 验证流量是否可以通过数据库端口在客户端主机和数据库实例之间流动。例如，使用 psql 连接和访问数据库。

以下是 Amazon Managed Microsoft AD 的示例 Krb5.conf 内容。

```
[libdefaults]
 default_realm = EXAMPLE.COM
[realms]
 EXAMPLE.COM = {
  kdc = example.com
  admin_server = example.com
 }
[domain_realm]
```

```
 .example.com = EXAMPLE.COM
 example.com = EXAMPLE.COM
```

以下是本地 Microsoft Active Directory 的示例 krb5.conf 内容。

```
[libdefaults]
 default_realm = EXAMPLE.COM
[realms]
 EXAMPLE.COM = {
  kdc = example.com
  admin_server = example.com
 }
 ONPREM.COM = {
  kdc = onprem.com
  admin_server = onprem.com
 }
[domain_realm]
 .example.com = EXAMPLE.COM
 example.com = EXAMPLE.COM
 .onprem.com = ONPREM.COM
 onprem.com = ONPREM.COM
 .rds.amazonaws.com = EXAMPLE.COM
 .amazonaws.com.cn = EXAMPLE.COM
 .amazon.com = EXAMPLE.COM
```

在 Active Directory 域中管理 RDS for PostgreSQL 数据库实例

可以使用控制台、CLI 或 RDS API 来管理数据库实例及其与 Microsoft Active Directory 的关系。例如，您可以关联 Active Directory 以启用 Kerberos 身份验证。您也可以删除 Active Directory 关联以禁用 Kerberos 身份验证。您也可以将由一个 Microsoft Active Directory 在外部进行身份验证的数据库实例移动到另一个 Active Directory。

例如，使用 CLI，您可以执行下列操作：

- 要再次尝试为失败的成员启用 Kerberos 身份验证，请使用 modify-db-instance CLI 命令。为 `--domain` 选项指定当前成员的目录 ID。
- 要在数据库实例上禁用 Kerberos 身份验证，请使用 modify-db-instance CLI 命令。为 `none` 选项指定 `--domain`。
- 要将数据库实例从一个域移动到另一个域，请使用 modify-db-instance CLI 命令。为 `--domain` 选项指定新域的域标识符。

了解域成员资格

在创建或修改数据库实例后，其将成为域的成员。您可以在控制台中查看域成员身份状态，也可以通过运行 describe-db-instances CLI 命令来查看。数据库实例的状态可以是以下状态之一：

- `kerberos-enabled` – 数据库实例已启用 Kerberos 身份验证。
- `enabling-kerberos` – Amazon是在此数据库实例上启用 Kerberos 身份验证的过程。
- `pending-enable-kerberos` – 启用 Kerberos 身份验证正在此数据库实例上等待处理。
- `pending-maintenance-enable-kerberos` – Amazon将尝试在下一个计划的维护时段在数据库实例上启用 Kerberos 身份验证。
- `pending-disable-kerberos` – 禁用 Kerberos 身份验证正在此数据库实例上等待处理。
- `pending-maintenance-disable-kerberos` – Amazon 将尝试在下一个计划的维护时段在数据库实例上禁用 Kerberos 身份验证。
- `enable-kerberos-failed` – 出现一个配置问题，导致 Amazon 无法在数据库实例上启用 Kerberos 身份验证。在重新发出命令以修改数据库实例之前纠正配置问题。
- `disabling-kerberos` – Amazon是在此数据库实例上启用 Kerberos 身份验证的过程。

启用 Kerberos 身份验证的请求可能因网络连接问题或不正确的 IAM 角色而失败。在某些情况下，在创建或修改数据库实例时，尝试启用 Kerberos 身份验证可能会失败。如果是这样，请确保使用正确的 IAM 角色，然后修改数据库实例以加入域。

> **Note**
>
> 仅针对 RDS for PostgreSQL 的 Kerberos 身份验证将流量发送到域的 DNS 服务器。在运行 PostgreSQL 的数据库实例上，所有其他 DNS 请求将被视为出站网络访问。有关使用 RDS for PostgreSQL 进行出站网络访问的更多信息，请参阅 将自定义 DNS 服务器用于出站网络访问。

使用 Kerberos 身份验证连接到 PostgreSQL

您可以使用 pgAdmin 界面或命令行界面（如 psql）通过 Kerberos 身份验证连接到 PostgreSQL。有关连接的更多信息，请参阅 连接到运行 PostgreSQL 数据库引擎的数据库实例 。有关获取连接所需的端点、端口号和其他详细信息的信息，请参阅连接到 PostgreSQL 数据库实例。

> **Note**
>
> PostgreSQL 中的 GSSAPI 身份验证和加密由 Kerberos 库 `libkrb5.so` 实施。`postgres_fdw` 和 `dblink` 等功能也依赖于同一个库，以通过 Kerberos 身份验证或加密进行出站连接。

pgAdmin

要使用 pgAdmin 通过 Kerberos 身份验证连接到 PostgreSQL，请执行以下步骤：

1. 在您的客户端计算机上启动 pgAdmin 应用程序。
2. 在 Dashboard (控制面板) 选项卡上，选择 Add New Server (添加新服务器)。
3. 在 Create - Server (创建 - 服务器) 对话框中，在 General (常规) 选项卡上键入名称以在 pgAdmin 中标识该服务器。
4. 在 Connection (连接) 选项卡上，键入您的 RDS for PostgreSQL 数据库的以下信息。
   - 对于 Host（主机），输入 RDS for PostgreSQL 数据库实例的端点。端点看起来类似于以下所示：

     ```
     RDS-DB-instance.111122223333.aws-region.rds.amazonaws.com
     ```

     要从 Windows 客户端连接到本地 Microsoft Active Directory，请使用 Amazon Managed Active Directory 的域名，而不是主机端点中的 `rds.amazonaws.com`。例如，假设 Amazon Managed Active Directory 的域名为 `corp.example.com`。然后，对于 Host（主机），按如下方式指定端点：

     ```
     RDS-DB-instance.111122223333.aws-region.corp.example.com
     ```

   - 对于 Port (端口)，输入分配的端口。
   - 对于 Maintenance database (维护数据库)，输入客户端将连接到的初始数据库的名称。
   - 对于用户名，键入您在 步骤 7：为您的 Kerberos 主体创建 PostgreSQL 用户 中为 Kerberos 身份验证输入的用户名。
5. 选择 Save。

psql

要使用 psql 通过 Kerberos 身份验证连接到 PostgreSQL，请执行以下步骤：

1. 在命令提示符处，运行以下命令。

```
kinit username
```

将 *username* 替换为用户名。在提示符下，输入在 Microsoft Active Directory 中为用户存储的密码。

2. 如果 PostgreSQL 数据库实例使用可公开访问的 VPC，请将数据库实例端点的 IP 地址放在 EC2 客户端上的 `/etc/hosts` 文件中。例如，以下命令获取 IP 地址，然后将其放在 `/etc/hosts` 文件中。

```
% dig +short PostgreSQL-endpoint.Amazon-Region.rds.amazonaws.com
;; Truncated, retrying in TCP mode.
ec2-34-210-197-118.Amazon-Region.compute.amazonaws.com.
34.210.197.118

% echo " 34.210.197.118  PostgreSQL-endpoint.Amazon-Region.rds.amazonaws.com" >> /
etc/hosts
```

如果您从 Windows 客户端使用本地 Microsoft Active Directory，则需要使用特殊终端节点进行连接。在主机终端节点中使用 `rds.amazonaws.com` Managed Active Directory 的域名，而不是 Amazon 域 Amazon。

例如，假设 Amazon Managed Active Directory 的域名为 `corp.example.com`。为终端节点使用格式 *`PostgreSQL-endpoint.Amazon-Region.corp.example.com`*，将其放入 `/etc/hosts` 文件。

```
% echo " 34.210.197.118  PostgreSQL-endpoint.Amazon-Region.corp.example.com" >> /etc/
hosts
```

3. 使用以下 psql 命令登录到与 Active Directory 集成的 PostgreSQL 数据库实例。

```
psql -U username@CORP.EXAMPLE.COM -p 5432 -h PostgreSQL-endpoint.Amazon-
Region.rds.amazonaws.com postgres
```

要从 Windows 客户端使用本地 Active Directory 登录 PostgreSQL 数据库集群，请使用以下 psql 命令以及上一步中的域名 (`corp.example.com`)：

```
psql -U username@CORP.EXAMPLE.COM -p 5432 -h PostgreSQL-endpoint.Amazon-
Region.corp.example.com postgres
```

将自定义 DNS 服务器用于出站网络访问

RDS for PostgreSQL 支持您数据库实例上的出站网络访问，并允许从客户拥有的自定义 DNS 服务器进行域名服务 (DNS) 解析。通过自定义 DNS 服务器，您可以仅解析 RDS for PostgreSQL 数据库实例中的完全限定域名。

主题

- 启用自定义 DNS 解析
- 禁用自定义 DNS 解析
- 设置自定义 DNS 服务器

启用自定义 DNS 解析

要在您的客户 VPC 中启用 DNS 解析，请首先将自定义数据库参数组与 RDS for PostgreSQL 实例相关联。然后，通过将 `rds.custom_dns_resolution` 参数设置为 1 启用该参数，并重新启动数据库实例以使更改生效。

禁用自定义 DNS 解析

要在您的客户 VPC 中关闭 DNS 解析，请首先将自定义数据库参数组的 `rds.custom_dns_resolution` 参数设置为 0 禁用该参数。然后，重新启动数据库实例以使更改生效。

设置自定义 DNS 服务器

自定义 DNS 名称服务器设置完成后，它需要 30 分钟将更改传播到数据库实例中。将更改传播到数据库实例后，需要进行 DNS 查找的所有出站网络流量将通过端口 53 查询您的 DNS 服务器。

> **Note**
>
> 如果未设置自定义 DNS 服务器，并且 `rds.custom_dns_resolution` 设置为 1，则会使用 Amazon Route 53 私有区域解析主机。有关更多信息，请参阅使用私有托管区域。

为您的 RDS for PostgreSQL 数据库实例设置自定义 DNS 服务器

1. 在附加到 VPC 的动态主机配置协议 (DHCP) 选项集中，将 `domain-name-servers` 选项设置为您的 DNS 名称服务器的 IP 地址。有关更多信息，请参阅 DHCP 选项集。

> **Note**
>
> `domain-name-servers` 选项最多可接受四个值，但您的 Amazon RDS 数据库实例只使用第一个值。

2. 请确保您的 DNS 服务器可以解析所有查找查询，包括公有 DNS 名称、Amazon EC2 私有 DNS 名称和特定客户的 DNS 名称。如果出站网络流量中包含您的 DNS 服务器不能处理的 DNS 查找，您的 DNS 服务器必须已配置适当的上游 DNS 提供程序。
3. 请配置您的 DNS 服务器，以生成等于或小于 512 字节的用户数据报协议 (UDP) 响应。
4. 请配置您的 DNS 服务器，以生成等于或小于 1024 字节的传输控制协议 (TCP) 响应。
5. 配置您的 DNS 服务器，允许流量通过 Amazon RDS 数据库实例的端口 53 入站。如果您的 DNS 服务器在 Amazon VPC 中，此 VPC 的安全组必须包含以下入站规则：允许端口 53 上的 UDP 和 TCP 流量。如果 DNS 服务器不在 Amazon VPC 中，它必须具有适当的防火墙设置，以允许端口 53 上的 UDP 和 TCP 入站流量。

   有关更多信息，请参阅您的 VPC 的安全组和添加和删除规则。
6. 配置您的 Amazon RDS 数据库实例的 VPC，允许流量通过端口 53 出站。您的 VPC 安全组必须包含以下出站规则：允许端口 53 上的 UDP 和 TCP 流量。

   有关更多信息，请参阅 Amazon VPC 用户指南中的您的 VPC 的安全组和添加和删除规则。
7. 确保正确配置 Amazon RDS 数据库实例和 DNS 服务器之间的路由路径以允许 DNS 流量。

   此外，如果 Amazon RDS 数据库实例和 DNS 服务器不在同一 VPC 中，确保在它们之间设置对等连接。有关更多信息，请参阅 Amazon VPC Peering Guide 中的什么是 VPC 对等连接？。

升级 RDS for PostgreSQL 数据库引擎

您可以为 PostgreSQL 数据库管理两种类型的升级：

- 操作系统更新 – 有时，为了应用安全修复或操作系统更改，Amazon RDS 可能需要更新数据库的基础操作系统。您可通过使用 Amazon Command Line Interface（Amazon CLI）或 RDS API 来决定 Amazon RDS 何时应用操作系统更新。有关操作系统更新的更多信息，请参阅 对数据库实例应用更新。
- 数据库引擎升级 – 如果 Amazon RDS 支持数据库引擎的新版本，您可以将您的数据库升级到新版本。

此上下文中的数据库是 RDS for PostgreSQL 数据库实例或多可用区数据库集群。

PostgreSQL 数据库有两种引擎升级方式：主要版本升级和次要版本升级。

主要版本升级。

主要版本升级会包含不与现有应用程序向后兼容的数据库更改。因此，您必须手动执行数据库的主要版本升级。您可以通过修改数据库实例或多可用区数据库集群来启动主要版本升级。在执行主要版本升级之前，建议您按照 为 RDS for PostgreSQL 升级选择主要版本中所述的步骤操作。

Amazon RDS 通过以下方式处理多可用区主版本升级：

- 多可用区数据库实例部署 – Amazon RDS 会同时升级主实例和任何备用实例。在升级完成期间，数据库可能会在几分钟内不可用。
- 多可用区数据库集群部署 – Amazon RDS 同时升级写入器和读取器实例。在升级完成期间，数据库可能会在几分钟内不可用。

如果您升级具有区域内只读副本的数据库实例，Amazon RDS 会升级副本以及主数据库实例。

Amazon RDS 不会升级多可用区数据库集群只读副本。如果您对多可用区数据库集群执行主要版本升级，则其只读副本的复制状态会更改为终止。升级完成后，您必须手动删除并重新创建只读副本。

> **Tip**
>
> 您可以使用蓝绿部署，最大限度地减少主要版本升级所需的停机时间。有关更多信息，请参阅 使用蓝绿部署进行数据库更新。

次要版本升级

相比之下，次要版本升级仅包括与现有应用程序向后兼容的更改。您可以通过修改数据库来手动启动次要版本升级。或者，您可以在创建或修改数据库时启用自动次要版本升级选项。这样做意味着 Amazon RDS 会在测试并批准新版本后自动升级您的数据库。

Amazon RDS 通过以下方式处理多可用区次要版本升级：

- 多可用区数据库实例部署 – Amazon RDS 会同时升级主实例和任何备用实例。在升级完成期间，数据库可能会在几分钟内不可用。
- 多可用区数据库集群部署 – Amazon RDS 一次升级一个读取器数据库实例。然后，其中一个读取器数据库实例将切换为新的写入器数据库实例。然后，Amazon RDS 升级旧的写入器实例（现在是读取器实例）。多可用区数据库集群通常可将次要版本升级的停机时间缩短至大约 35 秒。与 RDS 代理一起使用时，这种方法可以将停机时间进一步缩短到一秒或更短。有关更多信息，请参阅 Amazon RDS 代理。或者，您可以使用开源数据库代理，例如 ProxySQL、PgBouncer 或 Amazon Advanced JDBC Wrapper Driver。

如果数据库具有只读副本，则您必须在升级源实例或集群之前首先升级所有只读副本。

有关更多信息，请参阅 RDS for PostgreSQL 的自动次要版本升级。有关手动执行次要版本升级的信息，请参阅 手动升级引擎版本。

有关数据库引擎版本和弃用数据库引擎版本的策略的更多信息，请参阅 Amazon RDS 常见问题解答中的数据库引擎版本。

主题

PostgreSQL 升级的注意事项

为了安全地升级数据库，Amazon RDS 使用 PostgreSQL documentation 中所述的 pg_upgrade 实用程序

如果您的备份保留期大于 0，Amazon RDS 会在升级期间获取两份数据库快照。第一个数据库快照是数据库在进行任何升级更改前的快照。如果数据库升级失败，您可以还原此快照，以创建一个运行旧版本的数据库。第二个数据库快照是在升级完成后制作的。一旦备份保留期到期，就会自动删除这些数据库快照。

> **Note**
>
> 仅当您将数据库的备份保留期设置为大于 0 的数字时，Amazon RDS 才会在升级过程中制作数据库快照。要更改数据库实例的备份保留期，请参阅the section called “修改数据库实例”。您无法为多可用区数据库集群配置自定义备份保留期。

执行数据库实例的主要版本升级时，任何区域内只读副本也会自动升级。升级工作流程开始后，只读副本实例会等待主数据库实例上的 pg_upgrade 成功完成。然后，主数据库实例升级会等待只读副本实例升级完成。在升级完成之前，您会遇到中断。当您对多可用区数据库集群执行主要版本升级时，只读副本的复制状态会更改为终止。

升级完成后，您无法还原到以前版本的数据库引擎。如果要返回之前的版本，则需要还原在升级之前创建的数据库快照，以创建新的数据库。

查找有效的升级目标

使用 Amazon Web Services Management Console 升级数据库时，它会显示数据库的有效升级目标。您还可以使用以下 Amazon CLI 命令来标识数据库的有效升级目标：

对于 Linux、macOS 或 Unix：

```
aws rds describe-db-engine-versions \
  --engine postgres \
  --engine-version version-number \
  --query "DBEngineVersions[*].ValidUpgradeTarget[*].{EngineVersion:EngineVersion}" --
output text
```

对于 Windows：

```
aws rds describe-db-engine-versions ^
  --engine postgres ^
  --engine-version version-number ^
  --query "DBEngineVersions[*].ValidUpgradeTarget[*].{EngineVersion:EngineVersion}" --
output text
```

例如，要标识 PostgreSQL 版本 16.1 数据库的有效升级目标，请运行以下 Amazon CLI 命令：

对于 Linux、macOS 或 Unix：

```
aws rds describe-db-engine-versions \
  --engine postgres \
  --engine-version 16.1 \
  --query "DBEngineVersions[*].ValidUpgradeTarget[*].{EngineVersion:EngineVersion}" --
output text
```

对于 Windows：

```
aws rds describe-db-engine-versions ^
  --engine postgres ^
  --engine-version 16.1 ^
  --query "DBEngineVersions[*].ValidUpgradeTarget[*].{EngineVersion:EngineVersion}" --
output text
```

PostgreSQL 版本号

PostgreSQL 数据库引擎的版本编号序列如下所示：

- 对于 PostgreSQL 版本 10 及更高版本，引擎版本号格式为 major.minor。主要版本号是版本号的整数部分。次要版本号是版本号的小数部分。

  主要版本升级会增加版本号的整数部分，例如从 10.次要版本 升级为 11.次要版本。
- 对于版本低于 10 的 PostgreSQL 版本，引擎版本号的格式为 major.major.minor。主引擎版本号是版本号的整数和第一个小数点后的部分。例如，9.6 是一个主要版本。次要版本号是版本号的第三部分。例如，版本 9.6.12 中 12 是次要版本号。

  主要版本升级会增加版本号的主要版本部分。例如，从 9.6.12 升级到 11.14 是主要版本升级，其中 9.6 和 11 为主要版本号。

有关 RDS Extended Support 版本号的信息，请参阅Amazon RDS Extended Support 版本命名。

RDS for PostgreSQL 中的 RDS 版本号

RDS 版本号使用 *major.minor.patch* 命名方案。RDS 补丁版本包括在发布后添加到次要版本的重要错误修复。有关 RDS 扩展支持版本号的信息，请参阅Amazon RDS Extended Support 版本命名。

要识别数据库的 Amazon RDS 版本号，必须先使用以下命令创建 `rds_tools` 扩展：

```
CREATE EXTENSION rds_tools;
```

从 PostgreSQL 版本 15.2-R2 的发行版开始，您可以使用以下 SQL 查询找出 RDS for PostgreSQL 数据库的 RDS 版本号：

```
postgres=> SELECT rds_tools.rds_version();
```

例如，查询 RDS for PostgreSQL 15.2 数据库将返回以下内容：

```
rds_version
----------------
 15.2.R2
(1 row)
```

为 RDS for PostgreSQL 升级选择主要版本

主要版本升级可包含不与数据库的以前版本向后兼容的更改。新功能会导致现有应用程序无法正常工作。出于此原因，Amazon RDS 不会自动应用主要版本升级。要执行主要版本升级，请手动修改数据库。在将升级应用到生产数据库之前，确保彻底地测试任何升级，验证您的应用程序是否正常工作。执行 PostgreSQL 主要版本升级时，建议您按照 如何执行 RDS for PostgreSQL 的主要版本升级 中所述的步骤操作。

将 PostgreSQL 单可用区数据库实例或多可用区数据库实例部署升级到其下一个主要版本时，与数据库关联的任何只读副本也会升级到该下一个主要版本。在某些情况下，您可以在升级时跳至更高的主要版本。如果跳过升级主要版本，则只读副本也会升级到该目标主要版本。跳过其他主要版本升级到版本 11 有一定的限制。您可以在 如何执行 RDS for PostgreSQL 的主要版本升级 中所述的步骤中找到详细信息。

PostgreSQL 引擎升级期间大多数 PostgreSQL 扩展都不会升级。这些扩展必须单独升级。有关更多信息，请参阅 在 RDS for PostgreSQL 数据库中升级 PostgreSQL 扩展。

通过运行以下 Amazon CLI 查询，您可以找出哪些主要版本适用于您的 RDS for PostgreSQL 数据库：

```
aws rds describe-db-engine-versions --engine postgres  --engine-version your-version
 --query "DBEngineVersions[*].ValidUpgradeTarget[*].{EngineVersion:EngineVersion}" --
output text
```

下表总结了此查询结果的所有可用版本。版本号上的星号（*）表示不再支持该版本。如果当前版本不受支持，建议您升级到最新的次要版本升级目标或该版本的其他可用升级目标之一。

当前源版本	升级目标
17.5	无
17.4	17.5
17.3	17.5、17.4
17.2	17.5、17.4、17.3
17.1	17.5、17.4、17.3、17.2
16.9	17.5
16.8	17.5、17.4
16.7	17.5、17.4、17.3 16.8
16.6	17.5、17.4、17.3、17.2 16.8、16.7
16.5	17.5、17.4、17.3、17.2、17.1 16.8、16.7、16.6
16.4	17.5、17.4、17.3、17.2、17.1 16.8、16.7、16.6、16.5
16.3	17.5、17.4、17.3、17.2、17.1

当前源版本	升级目标
	16.8、16.7、16.6、16.5、16.4
16.2	17.5、17.4、17.3、17.2、17.1 16.8、16.7、16.6、16.5、16.4、16.3
16.1	17.5、17.4、17.3、17.2、17.1 16.8、16.7、16.6、16.5、16.4、16.3、16.2
15.13	17.5 16.9
15.12	17.4 16.9、16.8 15.13
15.11	17.3 16.9、16.8、16.7 15.13、15.12
15.10	17.2 16.9、16.8、16.7、16.6 15.13、15.12、15.11
15.9	17.1 16.9、16.8、16.7、16.6、16.5 15.13、15.12、15.11、15.10
15.8	16.9、16.8、16.7、16.6、16.5、16.4 15.13、15.12、15.11、15.10、15.9

当前源版本	升级目标
15.7	16.9、16.8、16.7、16.6、16.5、16.4、16.3 15.13、15.12、15.11、15.10、15.9、15.8
15.6	16.9、16.8、16.7、16.6、16.5、16.4、16.3、16.2 15.13、15.12、15.11、15.10、15.9、15.8、15.7
15.5	16.9、16.8、16.7、16.6、16.5、16.4、16.3、16.2、16. 15.13、15.12、15.11、15.10、15.9、15.8、15.7、15.6
15.4	16.9、16.8、16.7、16.6、16.5、16.4、16.3、16.2、16. 15.13、15.12、15.11、15.10、15.9、15.8、15.7、15.6
15.3*	16.9、16.8、16.7、16.6、16.5、16.4、16.3、16.2、16. 15.13、15.12、15.11、15.10、15.9、15.8、15.7、15.6
15.2*	16.9、16.8、16.7、16.6、16.5、16.4、16.3、16.2、16. 15.13、15.12、15.11、15.10、15.9、15.8、15.7、15.6
14.18	17.5 16.9 15.13
14.17	17.4 16.8 15.13、15.12 14.18

当前源版本	升级目标
14.16	17.3 16.7 15.13、15.12、15.11 14.18、14.17
14.15	17.2 16.6 15.13、15.12、15.11、15.10 14.18、14.17、14.16
14.14	17.1 16.5 15.13、15.12、15.11、15.10、15.9 14.18、14.17、14.16、14.15
14.13	16.4 15.13、15.12、15.11、15.10、15.9、15.8 14.18、14.17、14.16、14.15、14.14
14.12	16.3 15.13、15.12、15.11、15.10、15.9、15.8、15.7 14.18、14.17、14.16、14.15、14.14、14.13
14.11	16.2 15.13、15.12、15.11、15.10、15.9、15.8、15.7、15.6 14.18、14.17、14.16、14.15、14.14、14.13、14.12

当前源版本	升级目标
14.10	16.1 15.13、15.12、15.11、15.10、15.9、15.8、15.7、15.6 14.18、14.17、14.16、14.15、14.14、14.13、14.12、1
14.9	15.13、15.12、15.11、15.10、15.9、15.8、15.7、15.6 14.18、14.17、14.16、14.15、14.14、14.13、14.12、1
14.8*	15.13、15.12、15.11、15.10、15.9、15.8、15.7、15.6 14.18、14.17、14.16、14.15、14.14、14.13、14.12、1
14.7.*	15.13、15.12、15.11、15.10、15.9、15.8、15.7、15.6 14.18、14.17、14.16、14.15、14.14、14.13、14.12、1
14.6*	15.13、15.12、15.11、15.10、15.9、15.8、15.7、15.6 14.18、14.17、14.16、14.15、14.14、14.13、14.12、1
14.5*	15.13、15.12、15.11、15.10、15.9、15.8、15.7、15.6 14.18、14.17、14.16、14.15、14.14、14.13、14.12、1
14.4*	15.13、15.12、15.11、15.10、15.9、15.8、15.7、15.6 14.18、14.17、14.16、14.15、14.14、14.13、14.12、1
14.3*	15.13、15.12、15.11、15.10、15.9、15.8、15.7、15.6 14.18、14.17、14.16、14.15、14.14、14.13、14.12、1
14.2*	15.13、15.12、15.11、15.10、15.9、15.8、15.7、15.6 14.18、14.17、14.16、14.15、14.14、14.13、14.12、1

当前源版本	升级目标
14.1*	15.13、15.12、15.11、15.10、15.9、15.8、15.7、15.6 14.18、14.17、14.16、14.15、14.14、14.13、14.12、1
13.21	17.5 16.9 15.13 14.17
13.20	17.4 16.8 15.12 14.18、14.17 13.21
13.19	17.3 16.7 15.11 14.18、14.17、14.16 13.21、13.20
13.18	16.6 15.10 14.18、14.17、14.15 13.21、13.20、13.19

当前源版本	升级目标
13.17	16.5 15.9 14.18、14.17、14.15、14.14 13.21、13.20、13.19、13.18
13.16	16.4 15.8 14.18、14.17、14.15、14.14、14.13 13.21、13.20、13.19、13.18、13.17
13.15	16.3 15.8、15.7 14.18、14.17、14.15、14.14、14.13、14.12 13.21、13.20、13.19、13.18、13.17、13.16
13.14	16.2 15.6 14.18、14.17、14.15、14.14、14.13、14.12、14.11 13.21、13.20、13.19、13.18、13.17、13.16、13.15
13.13	16.1 15.5 14.18、14.17、14.15、14.14、14.13、14.12、14.11、1 13.21、13.20、13.19、13.18、13.17、13.16、13.15、1

当前源版本	升级目标
13.12	15.4 14.18、14.17、14.15、14.14、14.13、14.12、14.11、1 13.21、13.20、13.19、13.18、13.17、13.16、13.15、1
13.11	15.3 14.18、14.17、14.15、14.14、14.13、14.12、14.11、1 13.21、13.20、13.19、13.18、13.17、13.16、13.15、1
13.10*	15.2 14.18、14.17、14.15、14.14、14.13、14.12、14.11、1 13.21、13.20、13.19、13.18、13.17、13.16、13.14、1
13.9*	14.18、14.17、14.15、14.14、14.13、14.12、14.11、1 13.21、13.20、13.19、13.18、13.17、13.16、13.14、1
13.8*	14.18、14.17、14.15、14.14、14.13、14.12、14.11、1 13.21、13.20、13.19、13.18、13.17、13.16、13.14、1
13.7*	14.18、14.17、14.15、14.14、14.13、14.12、14.11、1 13.21、13.20、13.19、13.18、13.17、13.16、13.14、1
13.6*	14.18、14.17、14.15、14.14、14.13、14.12、14.11、1 13.21、13.20、13.19、13.18、13.17、13.16、13.14、1
13.5*	14.18、14.17、14.15、14.14、14.13、14.12、14.11、1 13.21、13.20、13.19、13.18、13.17、13.16、13.14、1
13.4*	14.18、14.17、14.15、14.14、14.13、14.12、14.11、1 13.21、13.20、13.19、13.18、13.17、13.16、13.14、1

当前源版本	升级目标
13.3*	14.18、14.17、14.15、14.14、14.13、14.12、14.11、1 13.21、13.20、13.19、13.18、13.17、13.16、13.14、1
13.2*、13.1*	14.18、14.17、14.15、14.14、14.13、14.12、14.11、1 13.21、13.20、13.19、13.18、13.17、13.16、13.14、1
12.22	17.2 16.6 15.10 14.15 13.18
12.21	17.1 16.5 15.9 14.14 13.18、13.17 12.22
12.20	16.4 15.8 14.13 13.18、13.17、13.16 12.22、12.21

当前源版本	升级目标
12.19	16.3 15.7 14.12 13.18、13.17、13.16、13.15 12.22、12.21、12.20
12.18	16.2 15.6 14.11 13.18、13.17、13.16、13.15、13.14 12.22、12.21、12.20、12.19
12.17	16.1 15.5 14.10 13.18、13.17、13.16、13.15、13.14、13.13 12.22、12.21、12.20、12.19、12.18
12.16	15.4 14.9 13.18、13.17、13.16、13.15、13.14、13.13、13.12 12.22、12.21、12.20、12.19、12.18、12.17

当前源版本	升级目标
12.15	15.3 14.8 13.18、13.17、13.16、13.15、13.14、13.13、13.12、1 12.22、12.21、12.2012.19、12.18、12.17、12.16
12.14*	15.2 14.7 13.18、13.17、13.16、13.15、13.14、13.13、13.12、1 12.22、12.21、12.20、12.19、12.18、12.17、12.16、1
12.13*	14.6 13.18、13.17、13.16、13.15、13.14、13.13、13.12、1 12.22、12.21、12.20、12.19、12.18、12.17、12.16、1
12.12*	14.5 13.18、13.17、13.16、13.15、13.14、13.13、13.12、1 12.22、12.21、12.20、12.19、12.18、12.17、12.16、1
12.11*	14.4 13.18、13.17、13.16、13.15、13.14、13.13、13.12、1 12.22、12.21、12.20、12.19、12.18、12.17、12.16、1
12.10*	14.2 13.18、13.17、13.16、13.15、13.14、13.13、13.12、1 12.22、12.21、12.20、12.19、12.18、12.17、12.16、1

当前源版本	升级目标
12.9*	14.1 13.18、13.17、13.16、13.15、13.14、13.13、13.12、1 12.22、12.21、12.20、12.19、12.18、12.17、12.16、1
12.8*	13.18、13.17、13.16、13.15、13.14、13.13、13.12、1 12.22、12.21、12.20、12.19、12.18、12.17、12.16、1
12.7*	13.18、13.17、13.16、13.15、13.14、13.13、13.12、1 12.22、12.21、12.20、12.19、12.18、12.17、12.16、1
12.6*、12.5*、12.4*、12.3*、12.2*	13.18、13.17、13.16、13.15、13.14、13.13、13.12、1 12.22、12.21、12.20、12.19、12.18、12.17、12.16、1
11.22	16.1 15.5 14.10 13.13 12.17 11.22-RDS.20240418

* 此版本不再受支持。

如何执行 RDS for PostgreSQL 的主要版本升级

在 Amazon RDS for PostgreSQL 数据库上执行主要版本升级时，我们建议采用以下过程：

1. 准备一个版本兼容的参数组 – 如果您使用的是自定义参数组，则您有两个选择。您可为新数据库引擎版本指定默认参数组。或者您可为新数据库引擎版本创建您自己的自定义参数组。有关更多信息，请参阅the section called “参数组” 和the section called “数据库集群参数组”。

2. 检查不受支持的数据库类 – 检查数据库的实例类是否与您要升级到的 PostgreSQL 版本兼容。有关更多信息，请参阅 数据库实例类支持的数据库引擎。
3. 检查是否有不支持的使用方式：
   - 准备的事务 – 在尝试升级前，提交或回滚所有打开的已准备事务。

     您可以使用以下查询来验证您的数据库上是否没有未结的已准备事务。

     ```
     SELECT count(*) FROM pg_catalog.pg_prepared_xacts;
     ```
   - Reg* 数据类型 – 在尝试升级前取消使用所有 reg* 数据类型。除了 `regtype` 和 `regclass` 以外，您不能升级 reg* 数据类型。`pg_upgrade` 实用程序无法保留此数据类型，Amazon RDS 使用它来执行升级。

     要验证是否没有使用不支持的 reg* 数据类型，请对每个数据库使用以下查询。

     ```
     SELECT count(*) FROM pg_catalog.pg_class c, pg_catalog.pg_namespace n,
      pg_catalog.pg_attribute a
       WHERE c.oid = a.attrelid
           AND NOT a.attisdropped
           AND a.atttypid IN ('pg_catalog.regproc'::pg_catalog.regtype,
                              'pg_catalog.regprocedure'::pg_catalog.regtype,
                              'pg_catalog.regoper'::pg_catalog.regtype,
                              'pg_catalog.regoperator'::pg_catalog.regtype,
                              'pg_catalog.regconfig'::pg_catalog.regtype,
                              'pg_catalog.regdictionary'::pg_catalog.regtype)
           AND c.relnamespace = n.oid
           AND n.nspname NOT IN ('pg_catalog', 'information_schema');
     ```
4. 检查是否存在无效的数据库：
   - 确保没有无效的数据库。`pg_database` 目录中的 `datconnlimit` 列包含一个值 `-2`，用于将 `DROP DATABASE` 操作期间中断的数据库标记为无效。

     使用以下查询来检查是否存在无效的数据库：

     ```
     SELECT datname FROM pg_database WHERE datconnlimit = - 2;
     ```
   - 上一个查询返回无效数据库的名称。可以使用 `DROP DATABASE invalid_db_name;` 删除无效的数据库。也可以使用以下命令来删除无效的数据库：

```
SELECT 'DROP DATABASE ' || quote_ident(datname) || ';' FROM pg_database WHERE
 datconnlimit = -2 \gexec
```

有关无效数据库的更多信息，请参阅了解对无效数据库使用 autovacuum 的行为。

5. 处理逻辑复制槽：如果数据库有逻辑复制槽，则无法进行升级。逻辑复制插槽通常用于 Amazon DMS 迁移，以及将表从数据库复制到数据湖、BI 工具和其他目标。升级之前，请确保您知道正在使用的任何逻辑复制槽的用途，并确认可以将其删除。如果逻辑复制槽仍在使用中，则不应将其删除，但也无法继续升级。

   如果不需要逻辑复制槽，则可以使用以下 SQL 将其删除：

   ```
   SELECT * FROM pg_replication_slots WHERE slot_type NOT LIKE 'physical';
   SELECT pg_drop_replication_slot(slot_name);
   ```

   使用 `pglogical` 扩展的逻辑复制设置还需要删除插槽才能成功升级主要版本。有关如何识别和删除使用 `pglogical` 扩展创建的插槽的信息，请参阅 管理 RDS for PostgreSQL 的逻辑复制查槽。

6. 处理只读副本 – 单可用区数据库实例或多可用区数据库实例部署的升级也会将区域内只读副本与主数据库实例一起升级。Amazon RDS 不会升级多可用区数据库集群只读副本。

   您无法单独升级只读副本。如果您可以单独升级，则可能会导致主数据库和副本数据库具有不同 PostgreSQL 主要版本的情况。但是，只读副本升级可能会增加主数据库实例上的停机时间。为防止只读副本升级，请在开始升级过程之前将副本提升为独立实例或将其删除。

   升级过程将根据只读副本的当前参数组重新创建副本的参数组。只有在升级完成后，才能通过修改只读副本将自定义参数组应用于只读副本。有关只读副本的更多信息，请参阅 使用 Amazon RDS for PostgreSQL 只读副本。

7. 执行备份 – 建议您在执行主要版本升级前执行备份，以便为您的数据库获得一个已知的还原点。如果您的备份保留期大于 0，则在升级之前和之后，升级过程会创建数据库的数据库快照。要更改您的备份保留期，请参阅修改 Amazon RDS 数据库实例和the section called “修改多可用区数据库集群”。

   要手动执行备份，请参阅the section called “为单可用区数据库实例创建数据库快照”和the section called “创建多可用区数据库集群的快照”。

8. 在主要版本升级之前更新某些扩展 – 如果打算在升级时跳过主要版本，您需要在执行主要版本升级之前更新某些扩展。例如，从版本 9.5.x 或 9.6.x 升级到版本 11.x 将跳过主要版本。要更新的扩展包括 PostGIS 和用于处理空间数据的相关扩展。

- address_standardizer
- address_standardizer_data_us
- postgis_raster
- postgis_tiger_geocoder
- postgis_topology

由于 rdkit 不兼容，如果您使用的是 rdkit 版本 4.6.0 及更低版本，以及 PostgreSQL 版本 16 及更低版本，则无法直接升级到 PostgreSQL 版本 17。以下是升级选项：

- 如果您使用的是 PostgreSQL 版本 13 及更低版本，则需要先执行主要版本升级，升级到版本 14.14 及更高的 14 版本、15.9 及更高的 15 版本或者 16.5 及更高的 16 版本，然后再执行版本升级，升级到 PostgreSQL 17。
- 如果您使用的是 PostgreSQL 版本 14、15 或 16，则需要执行次要版本升级到 14.14 及更高版本的 14 版本、15.9 及更高版本的 15 版本或 16.5 及更高版本的 16 版本，然后再升级到 PostgreSQL 版本 17。

为您要使用的每个扩展运行以下命令：

```
ALTER EXTENSION PostgreSQL-extension UPDATE TO 'new-version';
```

有关更多信息，请参阅 在 RDS for PostgreSQL 数据库中升级 PostgreSQL 扩展。要了解有关升级 PostGIS 的更多信息，请参阅步骤 6：升级 PostGIS 扩展。

9. 在主要版本升级之前删除某些扩展 – 跳过主要版本以升级到版本 11.x 的升级不支持更新 pgRouting 扩展。从版本 9.4.x、9.5.x 或 9.6.x 升级到版本 11.x 时，将会跳过主要版本。可以安全地删除 pgRouting 扩展，然后在升级后将其重新安装到兼容的版本中。有关您可以更新到的扩展版本，请参阅支持的 PostgreSQL 扩展版本。

   PostgreSQL 版本 11 或更高版本不再支持 tsearch2 和 chkpass 扩展。如果要升级到版本 11.x，请在升级之前删除 tsearch2 和 chkpass 扩展。

10. 删除未知数据类型 – 根据目标版本删除 unknown 数据类型。

   PostgreSQL 版本 10 停止支持 unknown 数据类型。如果版本 9.6 数据库使用 unknown 数据类型，升级到版本 10 将显示错误消息，如下所示：

```
Database instance is in a state that cannot be upgraded: PreUpgrade checks failed: 
The instance could not be upgraded because the 'unknown' data type is used in user
 tables.
```

```
Please remove all usages of the 'unknown' data type and try again."
```

要在数据库中查找 unknown 数据类型以便删除有问题的列或将其更改为支持的数据类型，请使用以下 SQL：

```
SELECT DISTINCT data_type FROM information_schema.columns WHERE data_type ILIKE
 'unknown';
```

11. 执行升级试运行 – 我们强烈建议您在生产数据库的副本上测试主要版本升级，然后再尝试在生产数据库上进行升级。您可以监控重复测试数据库上的执行计划，以了解任何可能的执行计划回归，并评估其性能。要创建副本测试实例，您可以从最近的快照还原数据库，也可以通过时间点还原将数据库还原到最近的可还原时间。

    有关更多信息，请参阅 the section called “从快照还原” 或 the section called “时间点故障恢复”。对于多可用区数据库集群，请参阅the section called “从快照还原到多可用区数据库集群”或the section called “将多可用区数据库集群还原到指定时间”。

    有关执行升级的详细信息，请参阅 the section called “手动升级引擎版本”。

    请注意，在将版本 9.6 数据库升级到版本 10 时，PostgreSQL 10 在默认情况下启用并行查询。您可以将测试数据库上的 max_parallel_workers_per_gather 参数更改为 2，从而在升级之前测试并行度的影响。

    > **Note**
    >
    > 在 default.postgresql10 数据库参数组中，max_parallel_workers_per_gather 参数的默认值为 2。

    有关更多信息，请参阅 PostgreSQL 文档中的并行查询。要在版本 10 上禁用并行，请将 max_parallel_workers_per_gather 参数设置为 0。

    在主要版本升级期间，将暂时重命名 public 和 template1 数据库以及每个数据库中的 public 架构。这些对象显示在日志中，同时附加了它们的原始名称和随机字符串。附加该字符串的目的是，在主要版本升级期间保留自定义设置，例如，locale 和 owner。升级完成后，对象再次重命名为它们的原始名称。

> **Note**
>
> 在主要版本升级过程中，您无法为数据库实例或多可用区数据库集群执行时间点还原。Amazon RDS 执行升级之后，它会自动备份数据库。您可以通过执行时间点还原，还原到数据库自动备份完成之后、升级操作开始之前的时间。

12. 如果升级失败并出现预检查过程错误，请解决这些问题 – 在主要版本升级过程中，Amazon RDS for PostgreSQL 先运行预检查过程，以确定可能导致升级失败的任何问题。预检查过程检查实例上的所有数据库中的所有潜在的不兼容情况。

    如果预检查遇到问题，它将创建一个日志事件以指示升级预检查失败。对于所有数据库，预检查过程详细信息位于相应数据库的升级日志（名称为 pg_upgrade_precheck.log）中。Amazon RDS 在文件名中附加时间戳。有关查看日志的详细信息，请参阅 监控 Amazon RDS 日志文件。

    如果只读副本升级在预检查时失败，则失败的只读副本上的复制将中断，并将该副本置于终止状态。删除该只读副本，并根据升级后的主数据库实例重新创建只读副本。

    解决预检查日志中确定的所有问题，然后重试主要版本升级。以下是一个预检查日志示例。

```
------------------------------------------------------------------------
Upgrade could not be run on Wed Apr 4 18:30:52 2018
-------------------------------------------------------------------------
The instance could not be upgraded from 9.6.11 to 10.6 for the following reasons.
Please take appropriate action on databases that have usage incompatible with the
 requested major engine version upgrade and try the upgrade again.

* There are uncommitted prepared transactions. Please commit or rollback all prepared
 transactions.* One or more role names start with 'pg_'. Rename all role names that
 start with 'pg_'.

* The following issues in the database 'my"million$"db' need to be corrected before
 upgrading:** The ["line","reg*"] data types are used in user tables. Remove all
 usage of these data types.
** The database name contains characters that are not supported by RDS for
 PostgreSQL. Rename the database.
** The database has extensions installed that are not supported on the target
 database version. Drop the following extensions from your database: ["tsearch2"].
```

```
* The following issues in the database 'mydb' need to be corrected before
 upgrading:** The database has views or materialized views that depend on
 'pg_stat_activity'. Drop the views.
```

13. 如果在升级数据库时只读副本升级失败，解决问题 – 将失败的只读副本置于 `incompatible-restore` 状态并终止数据库上的复制。删除该只读副本，并根据升级后的主数据库实例重新创建只读副本。

   Note

   Amazon RDS 不会升级多可用区数据库集群的只读副本。如果您对多可用区数据库集群执行主要版本升级，则其只读副本的复制状态会更改为终止。

   只读副本升级可能会因以下原因而失败：

   - 即使等待了一段时间，也无法跟上主数据库实例。
   - 处于终端或不兼容的生命周期状态，如 storage-full、incompatible-restore 等。
   - 主数据库实例升级开始后，在只读副本上运行了单独的次要版本升级。
   - 只读副本实例使用了不兼容的参数。
   - 只读副本实例无法与主实例通信以同步数据文件夹。

14. 升级生产数据库 – 在成功完成试运行主要版本升级后，您就可以放心地升级生产数据库了。有关更多信息，请参阅 [手动升级引擎版本](#)。

15. 运行 `ANALYZE` 操作以刷新 `pg_statistic` 表。您应该为所有 PostgreSQL 数据库上的每个数据库执行此操作。在主要版本升级期间不会传输优化程序统计数据，因此您需要重新生成所有统计数据，避免出现性能问题。运行不带任何参数的命令，为当前数据库中的所有常规表生成统计数据，如下所示：

```
ANALYZE VERBOSE;
```

   `VERBOSE` 标记为可选项，可用于显示进度。有关更多信息，请参阅 PostgreSQL 文档中的 [ANALYZE](#)。

   Note

   升级后在系统上运行 ANALYZE，避免出现性能问题。

在完成主要版本升级后，我们建议执行以下操作：

- PostgreSQL 升级不会升级任何 PostgreSQL 扩展。要升级扩展，请参阅在 RDS for PostgreSQL 数据库中升级 PostgreSQL 扩展。
- （可选）使用 Amazon RDS 查看 `pg_upgrade` 实用程序生成的两个日志。它们是 `pg_upgrade_internal.log` 和 `pg_upgrade_server.log`。Amazon RDS 会在这些日志的文件名中附加时间戳。您可以像查看其他任何日志一样查看这些日志。有关更多信息，请参阅"监控 Amazon RDS 日志文件"。

  您也可以将升级日志上传到 Amazon CloudWatch Logs 中。有关更多信息，请参阅"将 PostgreSQL 日志发布到 Amazon CloudWatch Logs"。
- 要验证是否一切正常，请在升级的数据库上使用类似的工作负载测试应用程序。验证升级之后，您可以删除此测试实例。

RDS for PostgreSQL 的自动次要版本升级

如果在创建或修改数据库实例或多可用区数据库集群时启用自动次要版本升级选项，您可以自动升级数据库。

对于每个 RDS for PostgreSQL 主要版本，RDS 将一个次要版本指定为自动升级版本。在 Amazon RDS 测试并批准次要版本后，将在维护时段内自动进行次要版本升级。RDS 不会自动将发布的较新次要版本设置为自动升级版本。在 RDS 指定较新的自动升级版本之前，将会考虑一些条件，例如：

- 已知的安全问题
- PostgreSQL 社区版本中的错误
- 发布次要版本后的整体队列稳定性

您可以使用以下 Amazon CLI 命令确定特定 Amazon Web Services 区域中指定 PostgreSQL 次要版本的当前自动次要升级目标版本。

对于 Linux、macOS 或 Unix：

```
aws rds describe-db-engine-versions \
--engine postgres \
--engine-version minor-version \
--region region \
--query "DBEngineVersions[*].ValidUpgradeTarget[*].
{AutoUpgrade:AutoUpgrade,EngineVersion:EngineVersion}" \
```

```
--output text
```

对于 Windows：

```
aws rds describe-db-engine-versions ^
--engine postgres ^
--engine-version minor-version ^
--region region ^
--query "DBEngineVersions[*].ValidUpgradeTarget[*].
{AutoUpgrade:AutoUpgrade,EngineVersion:EngineVersion}" ^
--output text
```

例如，以下 Amazon CLI 命令可确定美国东部（俄亥俄州）Amazon Web Services 区域（us-east-2）中 PostgreSQL 次要版本 16.1 的自动次要升级目标。

对于 Linux、macOS 或 Unix：

```
aws rds describe-db-engine-versions \
--engine postgres \
--engine-version 16.1 \
--region us-east-2 \
--query "DBEngineVersions[*].ValidUpgradeTarget[*].
{AutoUpgrade:AutoUpgrade,EngineVersion:EngineVersion}" \
--output table
```

对于 Windows：

```
aws rds describe-db-engine-versions ^
--engine postgres ^
--engine-version 16.1 ^
--region us-east-2 ^
--query "DBEngineVersions[*].ValidUpgradeTarget[*].
{AutoUpgrade:AutoUpgrade,EngineVersion:EngineVersion}" ^
--output table
```

输出类似于以下内容。

```
----------------------------------
|    DescribeDBEngineVersions    |
+--------------+-----------------+
|  AutoUpgrade |  EngineVersion  |
+--------------+-----------------+
```

```
|   False        |  16.2            |
|   True         |  16.3            |
|   False        |  16.4            |
|   False        |  16.5            |
|   False        |  16.6            |
|   False        |  17.1            |
|   False        |  17.2            |
+----------------+------------------+
```

在此示例中，AutoUpgrade 值对于 PostgreSQL 版本 16.3 为 True。因此，自动次要升级目标是 PostgreSQL 版本 16.3，它在输出中突出显示。

如果满足以下条件，则会在维护时段内自动升级 PostgreSQL 数据库：

- 数据库启用了自动次要版本升级选项。
- 数据库运行的次要数据库引擎版本低于当前的自动升级次要版本。

有关更多信息，请参阅 自动升级次要引擎版本。

> **Note**
>
> PostgreSQL 升级不会升级 PostgreSQL 扩展。要升级扩展，请参阅在 RDS for PostgreSQL 数据库中升级 PostgreSQL 扩展。

在 RDS for PostgreSQL 数据库中升级 PostgreSQL 扩展

PostgreSQL 引擎升级不会升级大多数的 PostgreSQL 扩展。要在版本升级后更新扩展，请使用 ALTER EXTENSION UPDATE 命令。

> **Note**
>
> 有关更新 PostGIS 扩展的信息，请参阅使用 PostGIS 扩展管理空间数据（步骤 6：升级 PostGIS 扩展）。
> 要升级 pg_repack 扩展，先删除该扩展，然后在升级后的数据库中创建新版本。有关更多信息，请参阅 pg_repack 文档中的安装 pg_repack。

要升级扩展，请使用以下命令。

```
ALTER EXTENSION extension_name UPDATE TO 'new_version';
```

有关受支持的 PostgreSQL 扩展版本列表，请参阅 支持的 PostgreSQL 扩展版本。

要列出当前安装的扩展，请在以下命令中使用 PostgreSQL pg_extension 目录。

```
SELECT * FROM pg_extension;
```

要查看可用于安装的特定扩展版本的列表，请在以下命令中使用 PostgreSQL pg_available_extension_versions 视图。

```
SELECT * FROM pg_available_extension_versions;
```

升级 PostgreSQL 数据库快照引擎版本

借助 Amazon RDS，您可以为 PostgreSQL 数据库实例创建存储卷数据库快照。数据库快照是根据您的 Amazon RDS 实例使用的引擎版本创建的。您可以升级数据库快照的引擎版本。

将升级到新引擎版本的数据库快照还原后，请确保测试升级是否成功。有关主要版本升级的更多信息，请参阅升级 RDS for PostgreSQL 数据库引擎。要了解如何还原数据库快照，请参阅 还原到数据库实例。

您可以升级已加密或未加密的手动数据库快照。

要查看 RDS for PostgreSQL 数据库快照的可用引擎版本，请使用以下 Amazon CLI 示例。

```
aws rds describe-db-engine-versions --engine postgres  --engine-version example-
engine-version --query "DBEngineVersions[*].ValidUpgradeTarget[*].
{EngineVersion:EngineVersion}" --output text --include-all
```

有关 RDS for PostgreSQL 数据库快照的可用引擎版本的更多信息，请参阅 为 RDS for PostgreSQL 升级选择主要版本。

> **Note**
> 您不能升级在自动备份过程中创建的自动数据库快照。

控制台

升级数据库快照

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择快照。
3. 选择要升级的快照。
4. 对于 Actions (操作)，选择 Upgrade snapshot (升级快照)。此时会显示 Upgrade snapshot (升级快照) 页面。
5. 选择要升级到的 New engine version (新引擎版本)。
6. 选择 Save changes (保存更改) 以升级快照。

在升级过程中，将为此数据库快照禁用所有快照操作。此外，数据库快照状态由 available 变为 upgrading，完成后变为 active。如果由于快照损坏导致数据库快照不能升级，状态将变为 unavailable。此状态的快照无法恢复。

> **Note**
>
> 如果数据库快照升级失败，则快照将回滚到具有原始版本的原始状态。

Amazon CLI

要将数据库快照升级到新的数据库引擎版本，请使用 Amazon CLI modify-db-snapshot 命令。

参数

- `--db-snapshot-identifier` – 要升级的数据库快照的标识符。标识符必须是唯一的 Amazon 资源名称 (ARN)。有关更多信息，请参阅“Amazon RDS 中的 Amazon 资源名称（ARN）”。
- `--engine-version` – 升级后的数据库快照的引擎版本。

Example

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-snapshot \
    --db-snapshot-identifier my_db_snapshot \
    --engine-version new_version
```

对于 Windows：

```
aws rds modify-db-snapshot ^
    --db-snapshot-identifier my_db_snapshot ^
    --engine-version new_version
```

RDS API

要将数据库快照升级到新的数据库引擎版本，请调用 Amazon RDS API ModifyDBSnapshot 操作。

- `DBSnapshotIdentifier` – 要升级的数据库快照的标识符。标识符必须是唯一的 Amazon 资源名称 (ARN)。有关更多信息，请参阅“Amazon RDS 中的 Amazon 资源名称（ARN）”。

- EngineVersion – 升级后的数据库快照的引擎版本。

使用 Amazon RDS for PostgreSQL 只读副本

您可以通过向实例添加只读副本来扩展 Amazon RDS for PostgreSQL 数据库实例的读取。与其它 Amazon RDS 数据库引擎一样，RDS for PostgreSQL 使用 PostgreSQL 的原生复制机制来使只读副本与对源数据库的更改保持同步。有关只读副本和 Amazon RDS 的一般信息，请参阅使用数据库实例只读副本。

在下文中，您可以找到使用 RDS for PostgreSQL 只读副本的特定信息。

只读副本上的逻辑解码

RDS for PostgreSQL 支持使用 PostgreSQL 16.1 从备用数据库实例进行逻辑复制。这允许您从只读备用数据库实例创建逻辑解码，从而减少主数据库实例上的负载。对于需要在多个系统之间同步数据的应用程序，您可以实现更高的可用性。此功能可提高数据仓库和数据分析的性能。

此外，给定备用数据库实例上的复制插槽会持续将该备用数据库实例提升为主数据库实例。这意味着，如果主数据库实例发生失效转移或将备用数据库实例提升为新的主数据库实例，则复制插槽将持续存在，以前的备用数据库实例订阅用户不会受到影响。

在只读副本上创建逻辑解码

1. 开启逻辑复制 - 要在备用数据库实例上创建逻辑解码，您必须在源数据库实例及其物理副本上启用逻辑复制。有关更多信息，请参阅 PostgreSQL 只读副本配置。

   - 为新创建的 RDS for PostgreSQL 数据库实例启用逻辑复制 – 创建新的数据库自定义参数组，并将静态参数 `rds.logical_replication` 设置为 `1`。然后，将此数据库参数组与源数据库实例及其物理只读副本相关联。有关更多信息，请参阅 在 Amazon RDS 中将数据库参数组与数据库实例关联。
   - 为现有 RDS for PostgreSQL 数据库实例开启逻辑复制 – 修改源数据库实例及其物理只读副本的数据库自定义参数组，以将静态参数 `rds.logical_replication` 设置为 `1`。有关更多信息，请参阅 在 Amazon RDS 中修改数据库参数组中的参数。

   > **Note**
   >
   > 必须重启数据库实例才能应用这些参数更改。

您可以使用以下查询来验证源数据库实例及其物理只读副本上的 wal_level 和 rds.logical_replication 值。

```
Postgres=>SELECT name,setting FROM pg_settings WHERE name IN
 ('wal_level','rds.logical_replication');

 name                      | setting
-------------------------+---------
 rds.logical_replication | on
 wal_level               | logical
(2 rows)
```

2. 在源数据库中创建表 – 连接到源数据库实例中的数据库。有关更多信息，请参阅 连接到运行 PostgreSQL 数据库引擎的数据库实例。

   使用以下查询在源数据库中创建表并插入值：

```
Postgres=>CREATE TABLE LR_test (a int PRIMARY KEY);
CREATE TABLE
```

```
Postgres=>INSERT INTO LR_test VALUES (generate_series(1,10000));
INSERT 0 10000
```

3. 为源表创建发布 - 使用以下查询为源数据库实例上的表创建发布。

```
Postgres=>CREATE PUBLICATION testpub FOR TABLE LR_test;
CREATE PUBLICATION
```

   使用 SELECT 查询来验证在源数据库实例和物理只读副本实例上创建的发布的详细信息。

```
Postgres=>SELECT * from pg_publication;

oid    | pubname | pubowner | puballtables | pubinsert | pubupdate | pubdelete |
 pubtruncate | pubviaroot
-------+---------+----------+--------------+-----------+-----------+-----------
+-------------+------------
 16429 | testpub |    16413 | f            | t         | t         | t         | t
          | f
```

```
(1 row)
```

4. 从逻辑副本实例创建订阅 – 创建另一个 RDS for PostgreSQL 数据库实例作为逻辑副本实例。确保 VPC 设置正确，以确保此逻辑副本实例可以访问物理只读副本实例。有关更多信息，请参阅 Amazon VPC 和 Amazon RDS。如果您的源数据库实例处于空闲状态，则可能会出现连接问题，并且主数据库实例不会将数据发送到备用数据库实例。

```
Postgres=>CREATE SUBSCRIPTION testsub CONNECTION 'host=Physical replica host name
 port=port
                dbname=source_db_name user=user password=password'
 PUBLICATION testpub;
NOTICE:  created replication slot "testsub" on publisher
CREATE SUBSCRIPTION
```

```
Postgres=>CREATE TABLE LR_test (a int PRIMARY KEY);
CREATE TABLE
```

使用 SELECT 查询来验证逻辑副本实例上的订阅详细信息。

```
Postgres=>SELECT oid,subname,subenabled,subslotname,subpublications FROM
 pg_subscription;

oid    | subname | subenabled | subslotname | subpublications
-------+---------+------------+-------------+-----------------
 16429 | testsub | t          | testsub     | {testpub}
(1 row)
postgres=> select count(*) from LR_test;
 count
-------
 10000
(1 row)
```

5. 检查逻辑复制插槽状态 - 您只能看到源数据库实例上的物理复制插槽。

```
Postgres=>select slot_name, slot_type, confirmed_flush_lsn from
 pg_replication_slots;

slot_name                                     | slot_type | confirmed_flush_lsn
----------------------------------------------+-----------+---------------------
 rds_us_west_2_db_dhqfsmo5wbbjqrn3m6b6ivdhu4 | physical  |
```

```
(1 row)
```

但是，在您的只读副本实例上，您可以看到逻辑复制插槽和 confirmed_flush_lsn 值会随着应用程序主动使用逻辑更改而发生变化。

```
Postgres=>select slot_name, slot_type, confirmed_flush_lsn from
 pg_replication_slots;

slot_name | slot_type | confirmed_flush_lsn
-----------+-----------+---------------------
 testsub   | logical   | 0/500002F0
(1 row)
```

```
Postgres=>select slot_name, slot_type, confirmed_flush_lsn from
 pg_replication_slots;

slot_name | slot_type | confirmed_flush_lsn
-----------+-----------+---------------------
 testsub   | logical   | 0/5413F5C0
(1 row)
```

PostgreSQL 只读副本限制

以下是 PostgreSQL 只读副本的限制：

- PostgreSQL 只读副本是只读的。尽管只读副本不是可写的数据库实例，但您可以将其升级为独立的 RDS for PostgreSQL 数据库实例。不过该过程无法撤消。
- 如果 RDS for PostgreSQL 数据库实例运行的是 14.1 之前的 PostgreSQL 版本，则无法从另一个只读副本创建只读副本。RDS for PostgreSQL 仅支持在 RDS for PostgreSQL 版本 14.1 版及更高版本上级联只读副本。有关更多信息，请参阅 将级联只读副本用于 RDS for PostgreSQL。
- 如果您升级 PostgreSQL 只读副本，它将成为可写的数据库实例。它停止从源数据库实例接收预写日志（WAL）文件，并且已不再是只读实例。您可以像对任何 RDS for PostgreSQL 数据库实例一样从升级的数据库实例创建新只读副本。有关更多信息，请参阅 将只读副本提升为独立的数据库实例。
- 如果您从复制链（一系列级联只读副本）中提升 PostgreSQL 只读副本，则任何现有的下游只读副本将继续自动从提升后的实例接收 WAL 文件。有关更多信息，请参阅 将级联只读副本用于 RDS for PostgreSQL。

- 如果源数据库实例上没有运行任何用户事务，则关联的 PostgreSQL 只读副本会报告长达五分钟的复制滞后。副本滞后的计算公式为 `currentTime - lastCommitedTransactionTimestamp`，这意味着，当未处理任何事务时，副本滞后值会在一段时间内增加，直到预写日志（WAL）分段切换。默认情况下，RDS for PostgreSQL 每 5 分钟切换一次 WAL 分段，这会导致事务记录以及报告的滞后减少。
- 您无法为早于 14.1 的 RDS for PostgreSQL 版本的 PostgreSQL 只读副本启用自动备份。仅 RDS for PostgreSQL 14.1 及更高版本支持只读副本的自动备份。对于 RDS for PostgreSQL 13 和更早版本，如果需要只读副本的备份，请从只读副本创建快照。
- 只读副本不支持时间点故障恢复（PITR）。您只能将 PITR 用于主（写入器）实例，而不能用于只读副本。要了解更多信息，请参阅将 Amazon RDS 的数据库实例还原到指定时间。
- 对于 PostgreSQL 版本 12 及更低版本，只读副本会在 60-90 天的维护时段内自动重启以应用密码轮换。如果在安排的重启之前，副本失去了与源的连接，则实例仍将重启以继续复制。对于 PostgreSQL 版本 13 及更高版本，在密码轮换过程中，只读副本可能会出现短暂的复制断开连接和重新连接的情况。

PostgreSQL 只读副本配置

RDS for PostgreSQL 使用 PostgreSQL 本机流式复制来创建源数据库实例的只读副本。此只读副本数据库实例是源数据库实例的异步创建的物理副本。该副本是通过在源数据库实例和只读副本之间传输预写日志（WAL）数据的特殊连接创建。有关更多信息，请参阅 PostgreSQL 文档中的流式复制。

在源数据库实例上进行更改时，PostgreSQL 会通过此安全连接异步流式传输数据库更改。您可以通过将 `ssl` 参数设置为 `1`，来加密从客户端应用程序到源数据库实例或任何只读副本的通信。有关更多信息，请参阅 将 SSL 与 PostgreSQL 数据库实例结合使用。

PostgreSQL 使用复制角色执行流式复制。该角色拥有特权，但不能用于修改任何数据。PostgreSQL 使用处理复制的单个过程。

您可以在不影响源数据库实例的操作或用户的情况下创建 PostgreSQL 只读副本。Amazon RDS 为您在源数据库实例和只读副本上设置必要的参数和权限，而不会影响服务。为源数据库实例拍摄快照，该快照用于创建只读副本。如果您在将来某个时间点删除只读副本，则不会发生中断。

您可以从同一区域内的一个源数据库实例创建最多 15 个只读副本。自 RDS for PostgreSQL 14.1 起，您还可以从源数据库实例在链（级联）中创建最多三个级别的只读副本。有关更多信息，请参阅 将级联只读副本用于 RDS for PostgreSQL。在所有情况下，源数据库实例都需要配置自动备份。为此，您可以将数据库实例上的备份保留期设置为非 0 值。有关更多信息，请参阅 创建只读副本。

您可以在与源数据库实例相同的 Amazon Web Services 区域中为您的 RDS for PostgreSQL 数据库实例创建只读副本。这称为区域内复制。您还可以在与源数据库实例不同的 Amazon Web Services 区域中创建只读副本。这称为跨区域复制。有关设置跨区域只读副本的更多信息，请参阅在不同的 Amazon Web Services 区域中创建只读副本。根据 RDS for PostgreSQL 版本，支持区域内和跨区域复制过程的各种机制略有不同，如流式复制如何适用于不同的 RDS for PostgreSQL 版本中所述。

为了有效地进行复制，每个只读副本具有的计算和存储资源的量应与源数据库实例的一样多。如果扩展源数据库实例，则还应确保扩展只读副本。

如果 Amazon RDS 从一开始就阻止只读副本，它将覆盖只读副本上的任何不兼容参数。例如，假定源数据库实例上的 `max_connections` 参数值高于只读副本上的此参数值。在这种情况下，Amazon RDS 将只读副本上的参数更新为与源数据库实例上的参数相同的值。

RDS for PostgreSQL 只读副本可以访问通过源数据库实例上的外部数据包装器（FDW）提供的外部数据库。例如，假设您的 RDS for PostgreSQL 数据库实例使用 `mysql_fdw` 包装器从 RDS for MySQL 访问数据。如果是这样，只读副本也可以访问该数据。其他支持的 FDW 包括 `oracle_fdw`、`postgres_fdw` 和 `tds_fdw`。有关更多信息，请参阅 使用 Amazon RDS for PostgreSQL 支持的外部数据包装器。

将 RDS for PostgreSQL 只读副本用于多可用区配置

您可从单可用区或多可用区数据库实例中创建只读副本。您可以借助备用副本，使用多可用区部署提高关键数据的持久性和可用性。备用副本是专用的只读副本，可以在源数据库发生故障转移时承担工作负载。您不能使用备用副本来提供读取流量。但是，可以从大流量、多可用区数据库实例创建只读副本以分流只读查询的负载。要了解关于多可用区部署的更多信息，请参阅 Amazon RDS 的多可用区数据库实例部署。

如果多可用区部署的源数据库实例故障转移到备用可用区，则关联的只读副本将切换为使用备用可用区（现在为主可用区）作为其复制源。只读副本可能需要重新启动，具体取决于 RDS for PostgreSQL 版本，如下所示：

- PostgreSQL 13 及更高版本 – 不需要重新启动。只读副本将自动与新的主副本同步。但是，在某些情况下，您的客户端应用程序可能会缓存只读副本的域名服务（DNS）详细信息。如果是这样，请将生存时间（TTL）值设置为小于 30 秒。这样做可以防止只读副本继续留在陈旧的 IP 地址（因此可以防止它与新的主副本同步）。要了解有关此用法和其他最佳实践的更多信息，请参阅Amazon RDS 基本操作指导方针。
- PostgreSQL 12 和所有早期版本 – 只读副本在故障转移到备用副本后自动重新启动，因为备用副本（现为主）具有不同的 IP 地址和不同的实例名称。重启操作会将只读副本与新的主副本同步。

要详细了解失效转移，请参阅对 Amazon RDS 的多可用区数据库实例执行失效转移。要了解关于只读副本如何在多可用区部署中工作的更多信息，请参阅 使用数据库实例只读副本。

要为只读副本提供失效转移支持，您可以将只读副本创建为多可用区数据库实例，以便 Amazon RDS 在另一个可用区（AZ）中创建备用副本。创建您的只读副本作为多可用区数据库实例与源数据库是否为多可用区数据库实例无关。

将级联只读副本用于 RDS for PostgreSQL

自版本 14.1 开始，RDS for PostgreSQL 支持级联只读副本。使用级联只读副本，您可以扩展读取操作，而不会增加源 RDS for PostgreSQL 数据库实例的开销。源数据库实例不会将 WAL 日志的更新发送到每个只读副本。相反，级联系列中的每个只读副本都将 WAL 日志更新发送到该系列中的下一个只读副本。这将减轻源数据库实例的负担。

使用级联只读副本，您的 RDS for PostgreSQL 数据库实例会将 WAL 数据发送到链中的第一个只读副本。然后，该只读副本将 WAL 数据发送到链中的第二个副本，依此类推。最终结果是，链中的所有只读副本都具有 RDS for PostgreSQL 数据库实例中的更改，但不会只在源数据库实例上产生开销。

您可以从源 RDS for PostgreSQL 数据库实例在链中创建最多三个只读副本。例如，假设您具有 RDS for PostgreSQL 14.1 数据库实例 `rpg-db-main`。您可执行以下操作：

- 从 `rpg-db-main` 开始，创建链中的第一个只读副本 `read-replica-1`。
- 接下来，从 `read-replica-1`，创建链中的下一个只读副本 `read-replica-2`。
- 最后，从 `read-replica-2`，创建链中的第三个只读副本 `read-replica-3`。

除了 `rpg-db-main` 系列中的第三个级联只读副本之外，您无法创建另一个只读副本。从 RDS for PostgreSQL 源数据库实例到一系列级联只读副本末尾的完整实例系列最多可以包含四个数据库实例。

要使级联只读副本正常工作，请在 RDS for PostgreSQL 上启用自动备份。首先创建只读副本，然后在 RDS for PostgreSQL 数据库实例上启用自动备份。此过程与其他 Amazon RDS 数据库引擎相同。有关更多信息，请参阅 创建只读副本。

与任何只读副本一样，您可以升级属于级联一部分的只读副本。从只读副本链中升级只读副本将从链中移除该副本。例如，假设您希望将一些工作负载从 `rpg-db-main` 数据库实例转移到新实例，以便仅供会计部门使用。假设该示例中的链有三个只读副本，您决定升级 `read-replica-2`。该链受到如下影响：

- 升级 `read-replica-2` 会将其从复制链中移除。
  - 现在它是一个完全读/写数据库实例。

  - 它继续复制到 `read-replica-3`，就像在升级之前那样。
- 您的 `rpg-db-main` 继续复制到 `read-replica-1`。

有关升级只读副本的更多信息，请参阅[将只读副本提升为独立的数据库实例](#)。

> **Note**
>
> - RDS for PostgreSQL 不支持对级联副本进行主要版本升级。在执行主要版本升级之前，您需要删除级联副本。在完成对源数据库实例和一级副本的升级后，可以重新创建这些副本。
> - 对于级联只读副本，RDS for PostgreSQL 在第一复制级别为每个源数据库实例支持 15 个只读副本，在第二和第三复制级别为每个源数据库实例支持 5 个只读副本。

使用 RDS for PostgreSQL 创建跨区域级联只读副本

RDS for PostgreSQL 支持跨区域级联只读副本。您可以从源数据库实例创建跨区域副本，然后从该实例创建同区域副本。您也可以从源数据库实例创建同区域副本，然后从该实例创建跨区域副本。

创建跨区域副本，然后创建同区域副本

您可以使用版本 14.1 或更高版本的 RDS for PostgreSQL 数据库实例 `rpg-db-main` 执行以下操作：

1. 从 `rpg-db-main`（US-EAST-1）开始，在链中创建第一个跨区域只读副本 `read-replica-1`（US-WEST-2）。
2. 使用第一个跨区域 `read-replica-1`（US-WEST-2），在链中创建第二个只读副本 `read-replica-2`（US-WEST-2）。
3. 使用 `read-replica-2` 在链中创建第三个只读副本 `read-replica-3`（US-WEST-2）。

创建同区域副本，然后创建跨区域副本

您可以使用版本 14.1 或更高版本的 RDS for PostgreSQL 数据库实例 `rpg-db-main` 执行以下操作：

1. 从 `rpg-db-main`（US-EAST-1）开始，在链中创建第一个只读副本 `read-replica-1`（US-EAST-1）。
2. 使用 `read-replica-1`（US-EAST-1），在链中创建第一个跨区域只读副本 `read-replica-2`（US-WEST-2）。

3. 使用 `read-replica-2` (US-WEST-2) 在链中创建第三个只读副本 `read-replica-3`（US-WEST-2）。

有关创建跨区域只读副本的限制

- 数据库副本的跨区域级联链最多可跨两个区域以及包含四个级别。这四个级别包括数据库源和三个只读副本。

使用级联只读副本所带来的好处

- 提高了读取可扩展性：通过将读取查询分布到多个副本，级联复制有助于实现负载平衡。这将减轻写入器数据库的压力，从而提高性能，尤其是在读取密集型应用程序中。
- 地理分布；级联副本可位于不同的地理位置。这将减少远离主数据库的用户的延迟，并提供本地只读副本，从而提升性能和用户体验。
- 高可用性和灾难恢复：如果主服务器发生故障，可以将副本提升为主服务器，从而确保连续性。级联复制可提供多层失效转移选项来进一步增强此能力，从而提高系统的整体韧性。
- 灵活性和模块化增长：着系统规模的增长，可以在不同的级别添加新副本，而无需对主数据库进行重大重新配置。此模块化方法可实现复制设置的可扩展且可管理的增长。

使用跨区域只读副本的最佳实践

- 在提升一个副本之前，请创建其他副本。这既能节省时间，又能有效处理工作负载。

流式复制如何适用于不同的 RDS for PostgreSQL 版本

正如PostgreSQL 只读副本配置中讨论的，RDS for PostgreSQL 使用 PostgreSQL 的原生流式复制协议从源数据库实例发送 WAL 数据。它将源 WAL 数据发送到区域内和跨区域只读副本。在 9.4 版中，PostgreSQL 引入了物理复制槽作为复制过程的支持机制。

物理复制槽可防止源数据库实例在所有只读副本使用 WAL 数据之前将其删除。每个只读副本在源数据库实例上都有自己的物理槽。该槽跟踪副本可能需要的最旧 WAL（按逻辑序列号，LSN）。在所有槽和数据库连接都进展到超出给定 WAL（LSN）之后，该 LSN 将成为在下一个检查点移除的候选项。

Amazon RDS 使用 Amazon S3 来归档 WAL 数据。对于区域内只读副本，您可以在必要时使用此归档数据恢复只读副本。例如，当源数据库和只读副本之间的连接由于任何原因中断时，您就可以这么做。

在下表中，您可以找到 PostgreSQL 版本之间以及 RDS for PostgreSQL 使用的区域内和跨区域支持机制之间的差异的摘要。

版本	区域内	跨区域
PostgreSQL 14.1 及更高版本	• 复制槽 • Amazon S3 归档	• 复制槽
PostgreSQL 13 及更低版本	• Amazon S3 归档	• 复制槽

有关更多信息，请参阅 监控和调整复制过程。

了解控制 PostgreSQL 复制的参数

以下参数会影响复制过程，并确定只读副本与源数据库实例保持同步的程度：

max_wal_senders

`max_wal_senders` 参数指定源数据库实例可以通过流式复制协议同时支持的最大连接数。

默认值因 RDS for PostgreSQL 版本而各不相同：

- 对于版本 13、14 和 15，默认值为 20。
- 对于版本 16 及更高版本，默认值为 35。

此参数应设置为略高于实际只读副本数量。如果此参数设置为对只读副本数来说太少，则复制将停止。

有关更多信息，请参阅 PostgreSQL 文档中的 max_wal_senders。

> **Note**
>
> `max_wal_senders` 是一个静态参数，要求重启数据库实例才能使更改生效。

wal_keep_segments

`wal_keep_segments` 指定源数据库实例在 `pg_wal` 目录中保留的预写日志（WAL）文件的数量。默认设置为 32。

如果 wal_keep_segments 没有针对您的部署设置为足够大的值，只读副本可能远远落后于流式复制停止。如果出现此情况，Amazon RDS 将生成复制错误并开始对只读副本进行恢复。为此，它将重放来自 Amazon S3 的源数据库实例的已归档 WAL 数据。在只读副本赶上进度可继续流式复制之前，该恢复过程将继续。您可以在示例：只读副本如何从复制中断中恢复中看到 PostgreSQL 日志所捕获的这个过程的实际执行情况。

> **Note**
>
> 在 PostgreSQL 版本 13 中，wal_keep_segments 参数名为 wal_keep_size。它的目的与 wal_keep_segments 相同，但其默认值以兆字节（MB），而不是文件数为单位（2048MB）。有关更多信息，请参阅 PostgreSQL 文档中 wal_keep_segments 和 wal_keep_size。

max_slot_wal_keep_size

max_slot_wal_keep_size 参数控制 RDS for PostgreSQL 数据库实例保留在 pg_wal 目录中以服务插槽的 WAL 数据量。此参数用于使用复制槽的配置。此参数的默认值为 -1，这意味着对源数据库实例上保留多少 WAL 数据没有限制。有关监控复制槽的信息，请参阅监控 RDS for PostgreSQL 数据库实例的复制槽。

有关此参数的更多信息，请参阅 PostgreSQL 文档中的 max_slot_wal_keep_size。

每次向只读副本提供数据的 WAL 流中断时，PostgreSQL 都会切换为恢复模式。它使用来自 Amazon S3 的已存档 WAL 数据或使用与复制槽关联的 WAL 数据来恢复只读副本。在此过程完成后，PostgreSQL 会重新建立流式复制。

示例：只读副本如何从复制中断中恢复

在以下示例中，您可以找到演示只读副本恢复过程的日志详细信息。该示例来自与源数据库在同一 Amazon Web Services 区域中运行 PostgreSQL 版本 12.9 的 RDS for PostgreSQL 数据库实例，因此不使用复制槽。对于运行早于 14.1 版的 PostgreSQL 且具有区域内只读副本的其他 RDS for PostgreSQL 数据库实例，恢复过程相同。

当只读副本与源数据库实例失去联系时，Amazon RDS 会在日志中将问题记录为 FATAL: could not receive data from WAL stream 消息以及 ERROR: requested WAL segment ... has already been removed。如粗体行所示，Amazon RDS 通过重播归档的 WAL 文件来恢复副本。

```
2014-11-07 19:01:10 UTC::@:[23180]:DEBUG:  switched WAL source from archive to stream
 after failure
2014-11-07 19:01:10 UTC::@:[11575]:LOG: started streaming WAL from primary at 1A/
D3000000 on timeline 1
2014-11-07 19:01:10 UTC::@:[11575]:FATAL: could not receive data from WAL stream:
ERROR:  requested WAL segment 000000010000001A000000D3 has already been removed
2014-11-07 19:01:10 UTC::@:[23180]:DEBUG: could not restore file "00000002.history"
 from archive: return code 0
2014-11-07 19:01:15 UTC::@:[23180]:DEBUG: switched WAL source from stream to archive
 after failure recovering 000000010000001A000000D3
2014-11-07 19:01:16 UTC::@:[23180]:LOG:  restored log file "000000010000001A000000D3"
 from archive
```

当 Amazon RDS 在副本上重放足够多的已归档 WAL 数据以赶上进度时，将再次开始向只读副本流式传输。恢复流式传输时，Amazon RDS 会向日志文件中写入一个类似以下内容的条目。

```
2014-11-07 19:41:36 UTC::@:[24714]:LOG:started streaming WAL from primary at 1B/
B6000000 on timeline 1
```

设置控制共享内存的参数

您设置的参数确定了用于跟踪事务 ID、锁和已准备事务的共享内存的大小。备用实例的共享内存结构必须等于或大于主实例的共享内存结构。这样可以确保前者在恢复过程中不会耗尽共享内存。如果副本上的参数值小于主实例上的参数值，Amazon RDS 将自动调整副本参数并重启引擎。

受影响的参数有：

- max_connections
- max_worker_processes
- max_wal_senders
- max_prepared_transactions
- max_locks_per_transaction

为避免由于内存不足而导致副本 RDS 重启，我们建议将参数更改作为滚动重启应用于每个副本。在设置参数时，必须应用以下规则：

- 增加参数值：
  - 您应始终先增加所有只读副本的参数值，然后对所有副本执行滚动重启。然后，在主实例上应用参数更改并重启。

- 减小参数值：
  - 您应该首先减少主实例的参数值，然后执行重启。然后，将参数更改应用于所有关联的只读副本并执行滚动重启。

监控和调整复制过程

我们强烈建议您定期监控 RDS for PostgreSQL 数据库实例和只读副本。您需要确保只读副本与源数据库实例上的更改同步。当复制过程中断时，Amazon RDS 可以透明地恢复您的只读副本。但是，最好避免出现需要恢复的情况。使用复制槽进行恢复比使用 Amazon S3 归档快，但是任何恢复过程都可能会影响读取性能。

要确定只读副本与源数据库实例的同步程度，您可以执行以下操作：

- 检查源数据库实例和副本之间的 **ReplicaLag** 数量。副本滞后是只读副本滞后于其源数据库实例的时间量（以秒为单位）。此指标将报告以下查询的结果。

  ```
  SELECT extract(epoch from now() - pg_last_xact_replay_timestamp()) AS "ReplicaLag";
  ```

  副本滞后可指示只读副本与源数据库实例的同步程度。这是源数据库实例和特定只读实例之间的延迟量。副本滞后值较高可能表示源数据库实例使用的数据库实例类或存储类型（或两者）与其只读副本之间不匹配。数据库源实例和所有只读副本的数据库实例类和存储类型应相同。

  副本滞后也可能是间歇性连接问题导致的。您可以通过查看 Amazon RDS `ReplicaLag` 指标，在 Amazon CloudWatch 中监控复制滞后。若要了解有关 `ReplicaLag` 和 Amazon RDS 的其他指标的更多信息，请参阅 Amazon RDS 的 Amazon CloudWatch 指标。
- 查看 PostgreSQL 日志了解可用于调整设置的信息。在每个检查点中，PostgreSQL 日志都会捕获回收事务日志文件的数量，如以下示例所示。

  ```
  2014-11-07 19:59:35 UTC::@:[26820]:LOG:  checkpoint complete: wrote 376 buffers
   (0.2%);
  0 transaction log file(s) added, 0 removed, 1 recycled; write=35.681 s, sync=0.013 s,
   total=35.703 s;
  sync files=10, longest=0.013 s, average=0.001 s
  ```

  您可以使用此信息来确定在给定时间段内回收了多少事务文件，然后可以根据需要更改 `wal_keep_segments` 设置。例如，假设 `checkpoint complete` 时的 PostgreSQL 日志每隔 5 分钟显示 `35 recycled`。在本例中，`wal_keep_segments` 默认值 32 不足以跟上流式传输活动的节奏，因此应增加此参数的值。

- 使用 Amazon CloudWatch 监控可以预测复制问题的指标。您可以使用 Amazon CloudWatch 检查已收集的指标，而不是直接分析 PostgreSQL 日志。例如，您可以检查 `TransactionLogsGeneration` 指标的值，以查看源数据库实例生成了多少 WAL 数据。某些情况下，数据库实例上的工作负载可能会生成大量 WAL 数据。如果是这样，您可能需要更改源数据库实例和只读副本的数据库实例类。使用具有高（10Gbps）网络性能的实例类可以减少副本滞后。

监控 RDS for PostgreSQL 数据库实例的复制槽

所有 RDS for PostgreSQL 版本都将复制槽用于跨区域只读副本。RDS for PostgreSQL 14.1 及更高版本将复制槽用于区域内只读副本。区域内只读副本还使用 Amazon S3 来归档 WAL 数据。换句话说，如果您的数据库实例和只读副本运行 PostgreSQL 14.1 或更高版本，则复制槽和 Amazon S3 归档都可用于恢复只读副本。使用复制槽恢复只读副本比从 Amazon S3 归档中恢复快。因此，我们建议您监控复制槽和相关指标。

您可以通过查询 `pg_replication_slots` 视图来查看 RDS for PostgreSQL 数据库实例上的复制槽，如下所示。

```
postgres=> SELECT * FROM pg_replication_slots;
slot_name                  | plugin | slot_type | datoid | database | temporary |
 active | active_pid | xmin | catalog_xmin | restart_lsn | confirmed_flush_lsn |
 wal_status | safe_wal_size | two_phase
---------------------------+--------+-----------+--------+----------+-----------
+--------+------------+------+--------------+-------------+---------------------
+------------+---------------+-----------
rds_us_west_1_db_555555555 |        | physical  |        |          | f         | t
  |      13194 |      |              | 23/D8000060 |                     | reserved   |
             | f
(1 row)
```

`reserved` 值 `wal_status` 表示插槽持有的 WAL 数据量在 `max_wal_size` 参数的范围内。换句话说，复制槽的大小正确。其他可能的状态值如下所示：

- `extended` – 插槽超过了`max_wal_size`设置，但 WAL 数据会被保留。
- `unreserved` – 插槽不再拥有所有必需的 WAL 数据。其中一些将在下一个检查点移除。
- `lost` – 一些必需的 WAL 数据已删除。插槽不再可用。

`wal_status` 的 `unreserved` 和 `lost` 状态只有在 `max_slot_wal_keep_size` 为非负数时才会显示。

`pg_replication_slots` 视图显示了复制插槽的当前状态。要评估复制插槽的性能，您可以使用 Amazon CloudWatch 并监控以下指标：

- **`OldestReplicationSlotLag`**：显示源上尚未被滞后程度最高的副本占用的预写日志（WAL）数据量。
- **`TransactionLogsDiskUsage`**：显示 WAL 数据使用了多少存储空间。如果只读副本出现明显滞后，此指标的值可能会大幅增加。

若要了解将 Amazon CloudWatch 及其指标用于 RDS for PostgreSQL 的更多信息，请参阅 [使用 Amazon CloudWatch 监控 Amazon RDS 指标]。有关监控 RDS for PostgreSQL 数据库实例上的流复制的更多信息，请参阅 Amazon数据库博客上的 [Amazon RDS PostgreSQL 复制的最佳实践]。

RDS for PostgreSQL 只读副本故障排除

下面，您可以找到一些常见 RDS for PostgreSQL 只读副本问题的故障排除思路。

终止导致只读副本滞后的查询

在数据库中长时间运行的处于活动或空闲事务状态的事务可能会干扰 WAL 复制过程，从而增加复制滞后。因此，请务必使用 PostgreSQL `pg_stat_activity` 视图监控这些事务的运行时。

在主实例上运行类似于以下内容的查询，来查找长时间运行的查询的进程 ID（PID）：

```
SELECT datname, pid,usename, client_addr, backend_start,
xact_start, current_timestamp - xact_start AS xact_runtime, state,
backend_xmin FROM pg_stat_activity WHERE state='active';
```

```
SELECT now() - state_change as idle_in_transaction_duration, now() - xact_start as
 xact_duration,*
FROM  pg_stat_activity
WHERE state  = 'idle in transaction'
AND   xact_start is not null
ORDER BY 1 DESC;
```

确定查询的 PID 后，您可以选择结束此查询。

在主实例上运行类似于以下内容的查询，来终止长时间运行的查询：

```
SELECT pg_terminate_backend(PID);
```

使用 Amazon RDS 优化型读取功能提高 RDS for PostgreSQL 的查询性能

使用 Amazon RDS 优化型读取功能，您可以更快地处理 RDS for PostgreSQL 查询。与不使用 RDS 优化型读取功能的数据库实例或多可用区数据库集群相比，使用此功能的实例或集群可以实现最多快 50% 的查询处理速度。

主题

PostgreSQL 中的 RDS 优化型读取功能概述

默认情况下，在使用基于 NVMe 的数据库实例类时，RDS for PostgreSQL 版本 15.2 及更高版本、14.7 及更高版本以及 13.10 及更高版本上提供优化型读取功能。有关指示哪些实例使用 NVMe 的硬件规格，请参阅数据库实例类的硬件规格。

当您使用开启了 RDS 优化型读取功能的 RDS for PostgreSQL 数据库实例或多可用区数据库集群时，该实例或集群通过使用基于本地非易失性存储规范（NVMe）的固态硬盘（SSD）块级存储，可实现速度最多快 50% 的查询性能。您可以通过将 PostgreSQL 生成的临时表放置在本地存储上来实现更快的查询处理，这可减少通过网络流向 Elastic Block Storage（EBS）的流量。

在 PostgreSQL 中，临时对象分配给临时命名空间，该命名空间在会话结束时会自动删除。删除时的临时命名空间会删除任何依赖于会话的对象，包括模式限定的对象，例如表、函数、运算符甚至扩展。

在 RDS for PostgreSQL 中，为存储临时对象的这一临时工作区配置了 `temp_tablespaces` 参数。

以下查询返回表空间的名称及其位置。

```
postgres=> show temp_tablespaces;
temp_tablespaces
---------------------
```

```
rds_temp_tablespace
(1 row)
```

rds_temp_tablespace 是由 RDS 配置的指向 NVMe 本地存储的表空间。您可以随时切换回 Amazon EBS 存储，方法是使用 Amazon Web Services Management Console修改 Parameter group 中的此参数以指向除 rds_temp_tablespace 之外的任何表空间。有关更多信息，请参阅 [在 Amazon RDS 中修改数据库参数组中的参数]。也可以使用 SET 命令在会话级别将 temp_tablespaces 参数的值修改为 pg_default。修改此参数会将临时工作区重定向到 Amazon EBS。当 RDS 实例或集群的本地存储空间不足以执行特定的 SQL 操作时，切换回 Amazon EBS 会有所帮助。

```
postgres=> SET temp_tablespaces TO 'pg_default';
SET
```

```
postgres=> show temp_tablespaces;

 temp_tablespaces
------------------
 pg_default
```

RDS 优化读取的使用案例

以下是一些可从优化型读取功能中受益的使用案例：

- 包含公用表表达式（CTE）、派生表和分组操作的分析查询。
- 用于处理应用程序的未优化型查询的只读副本。
- 具有复杂操作（如 GROUP BY 和 ORDER BY）的按需或动态报告查询，这些操作无法始终使用适当的索引。
- 使用内部临时表的其他工作负载。
- 用于排序的 CREATE INDEX 或 REINDEX 操作。

RDS 优化读取的最佳实践

对于 RDS 优化读取使用以下最佳实践：

- 为只读查询添加重试逻辑，以防它们在执行过程中由于实例存储已满而失败。

- 使用 CloudWatch 指标 `FreeLocalStorage` 监控实例存储上的可用存储空间。如果由于数据库实例或多可用区数据库集群上的工作负载导致实例存储达到其限制，请修改此实例或集群以使用更大的数据库实例类。

使用 RDS 优化读取

当您在单可用区数据库实例部署、多可用区数据库实例部署或多可用区数据库集群部署中预调配具有基于 NVMe 的数据库实例类之一的 RDS for PostgreSQL 数据库实例时，该数据库实例会自动使用 RDS 优化型读取功能。

有关多可用区部署的更多信息，请参阅配置和管理 Amazon RDS 的多可用区部署。

要启用 RDS 优化读取，请执行以下操作之一：

- 使用其中一个基于 NVMe 的数据库实例类创建 RDS for PostgreSQL 数据库实例或多可用区数据库集群。有关更多信息，请参阅 创建 Amazon RDS 数据库实例。
- 修改现有 RDS for PostgreSQL 数据库实例或多可用区数据库集群，以使用其中一个基于 NVMe 的数据库实例类。有关更多信息，请参阅 修改 Amazon RDS 数据库实例。

在支持其中一个或多个数据库实例类（具有本地 NVMe SSD 存储）的所有 Amazon Web Services 区域中，均可使用 RDS 优化型读取功能。有关更多信息，请参阅 数据库实例类。

要切换回未优化读取功能的 RDS 实例，请将 RDS 实例或集群的数据库实例类修改为类似的实例类，该实例类对于数据库工作负载仅支持 EBS 存储。例如，如果当前数据库实例类是 db.r6gd.4xlarge，请选择 db.r6g.4xlarge 以切换回该实例类。有关更多信息，请参阅 修改 Amazon RDS 数据库实例。

监控使用 RDS 优化读取的数据库实例

您可以通过以下 CloudWatch 指标监控使用 RDS 优化型读取功能的数据库实例：

- `FreeLocalStorage`
- `ReadIOPSLocalStorage`
- `ReadLatencyLocalStorage`
- `ReadThroughputLocalStorage`
- `WriteIOPSLocalStorage`
- `WriteLatencyLocalStorage`
- `WriteThroughputLocalStorage`

这些指标提供有关可用实例存储的存储空间、IOPS 和吞吐量的数据。有关这些指标的更多信息，请参阅 Amazon RDS 的 Amazon CloudWatch 实例级指标。

要监控本地存储的当前使用情况，请登录数据库并运行以下查询：

```
SELECT
    spcname AS "Name",
    pg_catalog.pg_size_pretty(pg_catalog.pg_tablespace_size(oid)) AS "size"
FROM
    pg_catalog.pg_tablespace
WHERE
    spcname IN ('rds_temp_tablespace');
```

有关临时文件及其用法的更多信息，请参阅使用 PostgreSQL 管理临时文件。

PostgreSQL 中 RDS 优化型读取功能的限制

以下限制适用于 PostgreSQL 中的 RDS 优化型读取功能：

- 实例存储已满时，事务可能会失败。

将数据导入 Amazon RDS 上的 PostgreSQL

假设您要将现有的 PostgreSQL 部署移动到 Amazon RDS 中。任务的复杂性取决于数据库大小以及要传输的数据库对象类型。例如，考虑包含 GB 级别数据集以及存储过程和触发器的数据库。相比只有几个 MB 测试数据并且没有触发器或存储过程的简单数据库，此类数据库要复杂得多。

我们建议您在以下条件下使用本机 PostgreSQL 数据库迁移工具：

- 您具有一个同类迁移，其中，您正在从一个具有与目标数据库相同的数据库引擎的数据库中迁移。
- 您正在迁移整个数据库。
- 利用这些本机工具，您可以在最少的停机时间下迁移您的系统。

在其他大多数情况下，使用 Amazon Database Migration Service（Amazon DMS）是最佳方法。AmazonDMS 可以在不停机的情况下迁移数据库，对于许多数据库引擎，将继续进行持续复制，直到您已准备好切换到目标数据库。您可以使用 Amazon DMS 迁移到相同的数据库引擎或其他数据库引擎。如果要迁移到与源数据库不同的数据库引擎，您可以使用 Amazon Schema Conversion Tool (Amazon SCT)。您可以使用 Amazon SCT 迁移 Amazon DMS 未迁移的架构对象。有关 Amazon DMS 的更多信息，请参阅什么是 Amazon Database Migration Service？。

修改数据库参数组以包括以下仅供您导入 的设置。您应测试参数设置以查找对数据库实例最有效的设置：在导入完成后，您需要将这些参数还原为生产值。

对数据库实例设置进行以下修改：

- 禁用数据库实例备份 (将 backup_retention 设置为 0)。
- 禁用多可用区。

修改数据库参数组以包括以下设置。您只应在导入数据时使用这些设置。您应测试参数设置以查找对数据库实例最有效的设置：在导入完成后，您需要将这些参数还原为生产值。

参数	导入时的建议值	描述
`maintenance_work_mem`	524288、1048576、2097152 或 4194304 (KB)。这些设置相当于 512	此设置的只取决于您主机的大小。该参数在 CREATE INDEX 语句中使用，每个并行命令可以使用这个数量的内存。计算最佳值，这样就不会将该值设置得过高而出现内存不足的情况。

参数	导入时的建议值	描述
	MB、1 GB、2 GB 和 4 GB。	
`max_wal_size`	256（适用于 9.6 版本）、4096（适用于 10 及更高版本）	允许 WAL 在自动检查点期间增长的最大大小。增大此参数会增加崩溃恢复所需的时间。对于 PostgreSQL 9.6 及更高版本，此参数替代 `checkpoint_segments` 。 对于 PostgreSQL 9.6，此值的单位为 16 MB。对于更高版本，此值单位为 1 MB。例如，在 9.6 版本中，128 表示 128 个块，每个块大小为 16 MB。在 12.4 版本中，2048 代表 2048 个块，每个块大小为 1 MB。
`checkpoint_timeout`	1800	此设置的值可以减少 WAL 轮换的频率。
`synchronous_commit`	Off	禁用此设置可加快写入。禁用此参数会在服务器崩溃时增加数据丢失的风险（不禁用 FSYNC）。
`wal_buffers`	8192	该值以 8 KB 为单位。这也有助于您的 WAL 生成速度
`autovacuum`	0	在您加载数据时禁用 PostgreSQL auto vacuum 参数，这样它就不使用资源

将 `pg_dump -Fc` (压缩) 或 `pg_restore -j` (并行) 命令与这些设置结合使用。

> **Note**
>
> PostgreSQL 命令 `pg_dumpall` 所需要的 super_user 权限在创建数据库实例时并不授予，因此，无法使用该命令导入数据。

主题

- 从 Amazon EC2 实例导入 PostgreSQL 数据库
- 使用 \copy 命令将数据导入 PostgreSQL 数据库实例上的表
- 将 Amazon S3 中的数据导入到 RDS for PostgreSQL 数据库实例
- 在数据库实例之间传输 PostgreSQL 数据库

从 Amazon EC2 实例导入 PostgreSQL 数据库

如果您在 Amazon EC2 实例上的 PostgreSQL 服务器中有数据，并要将其移至 PostgreSQL 数据库实例，则可以按照此过程迁移数据。

1. 使用 pg_dump 创建一个文件，其中包含要加载的数据
2. 创建目标数据库实例
3. 使用 psql 在数据库实例上创建数据库，然后加载数据
4. 创建该数据库实例的数据库快照

以下各节提供了上述每个步骤的更多详细信息。

步骤 1：使用包含要加载的数据的 pg_dump 创建文件

pg_dump 实用程序使用 COPY 命令创建 PostgreSQL 数据库的架构和数据转储。pg_dump 生成的转储脚本将数据加载到一个同名的数据库中，然后重新创建表、索引和外键。您可以使用 `pg_restore` 命令和 -d 参数将数据恢复到具有不同的名称的数据库。

在创建数据转储之前，应查询要转储的表以获取行数，以使您可确认目标数据库实例上的数量。

以下命令为一个名为 mydb2 的数据库创建一个名为 mydb2dump.sql 的转储文件。

```
prompt>pg_dump dbname=mydb2 -f mydb2dump.sql
```

步骤 2：创建目标数据库实例

使用 Amazon RDS 控制台、Amazon CLI 或 API 创建目标 PostgreSQL 数据库实例。在备份保留设置为 0 的情况下创建该实例，然后禁用多可用区。这样做可更快地导入数据。您必须先在该实例上创建一个数据库，然后才能转储数据。数据库可以与包含转储数据的数据库具有相同名称。或者，您可以使用其他名称创建数据库。在这种情况下，您使用 `pg_restore` 命令和 -d 参数将数据还原到新命名的数据库中。

例如，以下命令可用于转储、还原和重命名数据库。

```
pg_dump -Fc -v -h [endpoint of instance] -U [master username] [database]
 > [database].dump
createdb [new database name]
pg_restore -v -h [endpoint of instance] -U [master username] -d [new database
 name] [database].dump
```

步骤 3：使用 psql 在数据库实例上创建数据库，然后加载数据

可使用用于运行 pg_dump 命令的同一连接，连接到目标数据库实例并重新创建数据库。使用 psql 时，可使用主用户名和主密码在数据库实例上创建数据库

以下示例使用 psql 和一个名为 mydb2dump.sql 的转储文件，在一个名为 mypginstance 的 PostgreSQL 数据库实例上创建一个名为 mydb2 的数据库：

对于 Linux、macOS 或 Unix：

```
psql \
   -f mydb2dump.sql \
   --host mypginstance.555555555555.aws-region.rds.amazonaws.com \
   --port 8199 \
   --username myawsuser \
   --password password \
   --dbname mydb2
```

对于 Windows：

```
psql ^
   -f mydb2dump.sql ^
   --host mypginstance.555555555555.aws-region.rds.amazonaws.com ^
   --port 8199 ^
   --username myawsuser ^
   --password password ^
   --dbname mydb2
```

> **Note**
>
> 作为安全最佳实践，请指定除此处所示提示以外的密码。

步骤 4：创建数据库实例的数据库快照

确认已将数据加载到数据库实例中后，建议创建目标 PostgreSQL 数据库实例的数据库快照。数据库快照是数据库实例的完整备份，可用于将数据库实例还原到已知状态。通过使用在加载后立即创建的数据库快照，您可以在遇到问题时不必再次加载数据。您也可以使用此类快照创建新的数据库实例。有关创建数据库快照的更多信息，请参阅[为 Amazon RDS 的单可用区数据库实例创建数据库快照]()。

使用 \copy 命令将数据导入 PostgreSQL 数据库实例上的表

PostgreSQL `\copy` 命令是可从 `psql` 交互式客户端工具获取的元命令，。您可以使用 `\copy` 将数据导入 RDS for PostgreSQL 数据库实例上的表中。若想使用 `\copy` 命令，您需要先在目标数据库实例上创建表结构，以便 `\copy` 具有待复制数据的目标。

您可以使用 `\copy` 从逗号分隔值 (CSV) 文件（如已导出并保存到客户端工作站的文件）中加载数据。

要将 CSV 数据导入目标 RDS for PostgreSQL 数据库实例，请先使用 `psql` 连接至目标数据库实例。

```
psql --host=db-instance.111122223333.aws-region.rds.amazonaws.com --port=5432 --
username=postgres --password --dbname=target-db
```

然后使用以下参数运行 `\copy` 命令来标识数据目标及其格式。

- `target_table`：应接收从 CSV 文件复制的数据的表名称。
- `column_list`：表的列规范。
- `'filename'`：本地工作站上 CSV 文件的完整路径。

```
 \copy target_table from '/path/to/local/filename.csv' WITH DELIMITER ',' CSV;
```

如果 CSV 文件包含列标题信息，则可以使用此版本的命令和参数。

```
\copy target_table (column-1, column-2, column-3, ...)
    from '/path/to/local/filename.csv' WITH DELIMITER ',' CSV HEADER;
```

如果 `\copy` 命令失败，PostgreSQL 会输出错误消息。

使用 `\copy` 元命令，在数据库预览环境中通过 `psql` 命令创建新的数据库实例，如以下示例所示。该示例使用 source-table 作为源表名称，使用 source-table.csv 作为 .csv 文件，使用 target-db 作为目标数据库：

对于 Linux、macOS 或 Unix：

```
$psql target-db \
    -U <admin user> \
    -p <port> \
    -h <DB instance name> \
    -c "\copy source-table from 'source-table.csv' with DELIMITER ','"
```

对于 Windows：

```
$psql target-db ^
    -U <admin user> ^
    -p <port> ^
    -h <DB instance name> ^
    -c "\copy source-table from 'source-table.csv' with DELIMITER ','"
```

有关 \copy 命令的完整详细信息，请参阅 PostgreSQL 文档中 psql 页面内的元命令部分。

将 Amazon S3 中的数据导入到 RDS for PostgreSQL 数据库实例

您可以将已使用 Amazon Simple Storage Service 存储的数据导入到 RDS for PostgreSQL 数据库实例上的表中。为此，您首先要安装 RDS for PostgreSQL aws_s3 扩展。该扩展提供用于从 AmazonS3 桶导入数据的函数。桶 是用于存储对象和文件的 Amazon S3 容器。数据可以位于逗号分隔值（CSV）文件、文本文件或压缩（gzip）文件中。接下来，您可以了解如何安装扩展以及如何将数据从 Amazon S3 导入到表中。

要从 Amazon S3 导入到 RDS for PostgreSQL，您的数据库必须运行 PostgreSQL 版本 10.7 或更高版本。。

如果您没有将数据存储在 Amazon S3 上，则需要先创建桶并存储数据。有关更多信息，请参阅《Amazon Simple Storage Service 用户指南》中的以下主题。

- 创建存储桶
- 向存储桶添加对象

支持从 Amazon S3 跨账户导入。有关更多信息，请参阅《Amazon Simple Storage Service 用户指南》中的授予跨账户权限。

从 S3 导入数据时，您可以使用客户托管式密钥进行加密。有关更多信息，请参阅《Amazon Simple Storage Service 用户指南》中的 Amazon KMS 中存储的 KMS 密钥。

主题

安装 aws_s3 扩展名

在将 Amazon S3 用于 RDS for PostgreSQL 数据库实例之前，您需要安装 aws_s3 扩展。此扩展提供从 Amazon S3 导入数据的函数。它还提供将数据从 RDS for PostgreSQL 数据库实例中导出到 Amazon S3 桶的函数。有关更多信息，请参阅 将数据从 RDS for PostgreSQL 数据库实例导出到 Amazon S3。aws_s3 扩展依赖于 aws_commons 扩展（需要时自动安装）中的一些帮助程序函数。

安装 **aws_s3** 扩展

1. 使用 psql（或 pgAdmin）以具有 rds_superuser 权限的用户身份，连接到 RDS for PostgreSQL 数据库实例。如果您在设置过程中保留原定设置名称，则以 postgres 进行连接。

```
psql --host=111122223333.aws-region.rds.amazonaws.com --port=5432 --
username=postgres --password
```

2. 要安装扩展，请运行以下命令。

```
postgres=> CREATE EXTENSION aws_s3 CASCADE;
NOTICE: installing required extension "aws_commons"
CREATE EXTENSION
```

3. 要验证扩展是否已安装，可以使用 psql \dx 元命令。

```
postgres=> \dx
       List of installed extensions
    Name     | Version |   Schema   |                 Description
-------------+---------+------------+---------------------------------------------
 aws_commons | 1.2     | public     | Common data types across Amazon services
 aws_s3      | 1.1     | public     | Amazon S3 extension for importing data from
 S3
 plpgsql     | 1.0     | pg_catalog | PL/pgSQL procedural language
(3 rows)
```

从 Amazon S3 导入数据和将数据导出到 Amazon S3 的函数现在可供使用。

从 Amazon S3 数据导入数据概述

将 S3 数据导入到 Amazon RDS

首先，收集您需要为该函数提供的详细信息。其中包括 RDS for PostgreSQL 数据库实例上的表名称、桶名称、文件路径、文件类型以及存储 Amazon S3 数据的 Amazon Web Services 区域。有关更多信息，请参阅《Amazon Simple Storage Service 用户指南》中的查看对象。

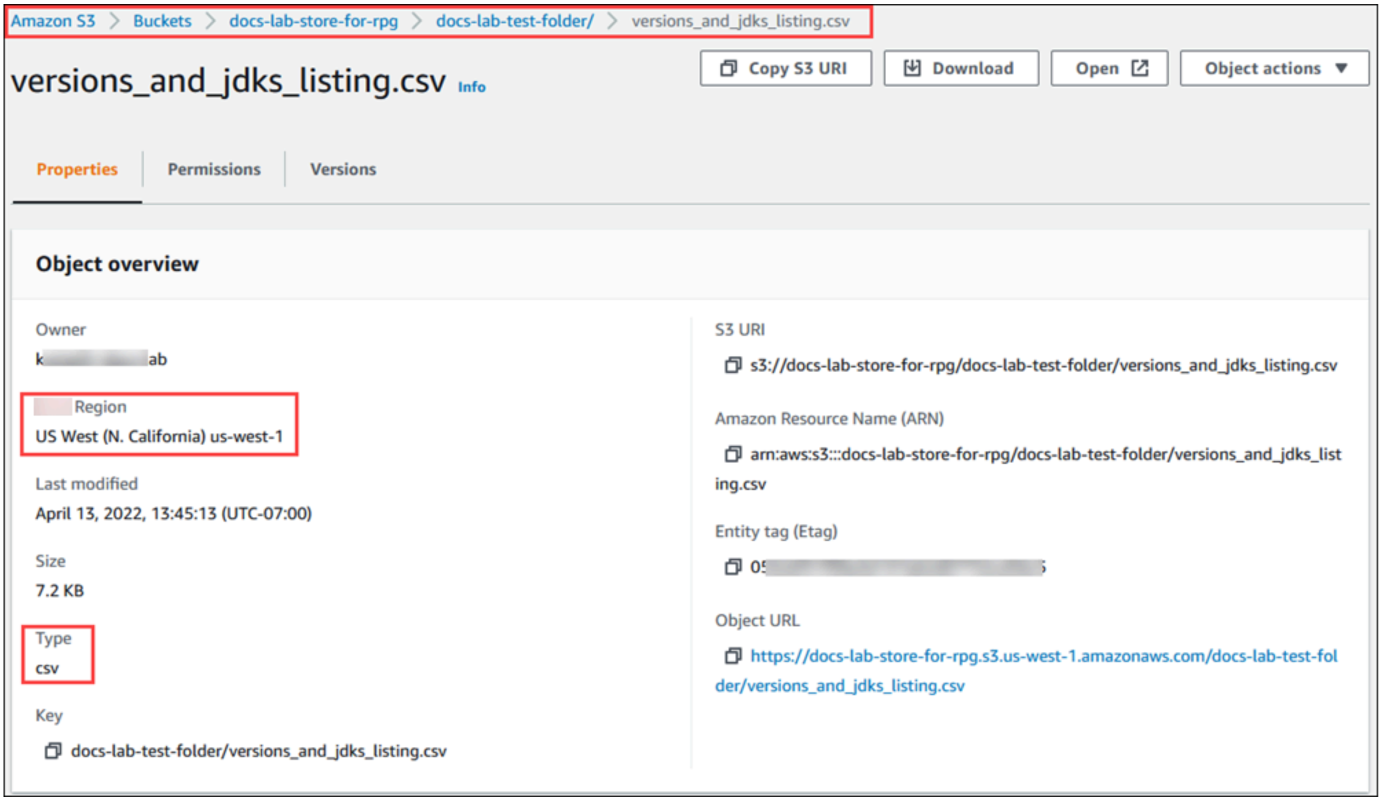
> **Note**
> 目前不支持从 Amazon S3 导入分段数据。

1. 获取 aws_s3.table_import_from_s3 函数要向其中导入数据的表的名称。例如，以下命令创建表 t1，供在后面的步骤中使用。

```
postgres=> CREATE TABLE t1
    (col1 varchar(80),
    col2 varchar(80),
    col3 varchar(80));
```

2. 获取有关 Amazon S3 桶和要导入的数据的详细信息。为此，请通过以下网址打开 Amazon S3 控制台：https://console.aws.amazon.com/s3/，然后选择 Buckets（桶）。在列表中找到包含您的数据的桶。选择桶，打开其 Object overview（对象概述）页面，然后选择 Properties（属性）。

   记下桶名称、路径、Amazon Web Services 区域和文件类型。您稍后需要提供 Amazon 资源名称（ARN），以便通过 IAM 角色设置对 Amazon S3 的访问权限。有关更多信息，请参阅设置 Amazon S3 存储桶的访问权限。下图显示了一个示例。



3. 您可以使用 Amazon CLI 命令 `aws s3 cp` 验证 AmazonS3 桶上数据的路径。如果该信息正确无误，该命令将下载 Amazon S3 文件的副本。

```
aws s3 cp s3://amzn-s3-demo-bucket/sample_file_path ./
```

4. 在 RDS for PostgreSQL 数据库实例上设置权限，以允许访问 Amazon S3 桶中的文件。为此，您可以使用 Amazon Identity and Access Management（IAM）角色或安全凭证。有关更多信息，请参阅 设置 Amazon S3 存储桶的访问权限。

5. 将收集的路径和其他 Amazon S3 对象详细信息（请参阅步骤 2）提供给用于构造 Amazon S3 URI 对象的 `create_s3_uri` 函数。要了解有关此函数的更多信息，请参阅aws_commons.create_s3_uri。以下是在 psql 会话期间构造此对象的示例。

```
postgres=> SELECT aws_commons.create_s3_uri(
   'docs-lab-store-for-rpg',
   'versions_and_jdks_listing.csv',
   'us-west-1'
) AS s3_uri \gset
```

在下一步中，您将此对象（`aws_commons._s3_uri_1`）传递到 `aws_s3.table_import_from_s3` 函数，以便将数据导入表中。

6. 调用 `aws_s3.table_import_from_s3` 函数，以将数据从 Amazon S3 导入到您的表中。有关参考信息，请参阅aws_s3.table_import_from_s3。有关示例，请参阅 将数据从 Amazon S3 导入到 RDS for PostgreSQL 数据库实例。

设置 Amazon S3 存储桶的访问权限

要从 Amazon S3 文件中导入数据，请为 RDS for PostgreSQL 数据库实例提供权限以访问包含该文件的 Amazon S3 存储桶。您可以通过两种方式提供 Amazon S3 存储桶的访问权限，如以下主题中所述。

主题

- 使用 IAM 角色访问 Amazon S3 存储桶
- 使用安全凭证访问 Amazon S3 存储桶
- Amazon S3 访问故障排除

使用 IAM 角色访问 Amazon S3 存储桶

从 Amazon S3 文件中加载数据之前，请为 RDS for PostgreSQL 数据库实例提供权限以访问该文件所在的 Amazon S3 存储桶。这样，您无需管理其他凭证信息或在 aws_s3.table_import_from_s3 函数调用中提供该信息。

为此，请创建一个 IAM 策略以提供 Amazon S3 存储桶的访问权限。创建一个 IAM 角色并将策略附加到该角色。然后，将该 IAM 角色分配给数据库实例。

通过 IAM 角色授予 RDS for PostgreSQL 数据库实例访问 Amazon S3 的权限

1. 创建一个 IAM 策略。

   该策略提供存储桶和对象权限，以允许 RDS for PostgreSQL 数据库实例访问 Amazon S3。

   在策略中包含以下必需操作，以允许将文件从 Amazon S3 存储桶传输到 Amazon RDS：

   - `s3:GetObject`
   - `s3:ListBucket`

   在策略中包含以下资源以标识 Amazon S3 存储桶以及存储桶中的对象。这会显示用于访问 Amazon S3 的 Amazon Resource Name (ARN) 格式。

- arn:aws:s3:::*amzn-s3-demo-bucket*
- arn:aws:s3:::*amzn-s3-demo-bucket*/*

有关为 RDS for PostgreSQL 创建 IAM policy 的更多信息，请参阅创建和使用适用于 IAM 数据库访问的 IAM 策略。另请参阅 IAM 用户指南中的教程：创建和附加您的第一个客户托管策略。

以下 Amazon CLI 命令使用这些选项创建一个名为 `rds-s3-import-policy` 的 IAM 策略。该策略授予对名为 *amzn-s3-demo-bucket* 的存储桶的访问权限。

> **Note**
>
> 记下此命令返回的策略的 Amazon 资源名称（ARN）。在后续步骤中将策略附加到 IAM 角色时，您需要此 ARN。

Example

对于 Linux、macOS 或 Unix：

```
aws iam create-policy \
   --policy-name rds-s3-import-policy \
   --policy-document '{
     "Version": "2012-10-17",
     "Statement": [
       {
         "Sid": "s3import",
         "Action": [
           "s3:GetObject",
           "s3:ListBucket"
         ],
         "Effect": "Allow",
         "Resource": [
           "arn:aws:s3:::amzn-s3-demo-bucket",
           "arn:aws:s3:::amzn-s3-demo-bucket/*"
         ]
       }
     ]
   }'
```

对于 Windows：

```
aws iam create-policy ^
   --policy-name rds-s3-import-policy ^
   --policy-document '{
     "Version": "2012-10-17",
     "Statement": [
       {
         "Sid": "s3import",
         "Action": [
           "s3:GetObject",
           "s3:ListBucket"
         ],
         "Effect": "Allow",
         "Resource": [
           "arn:aws:s3:::amzn-s3-demo-bucket",
           "arn:aws:s3:::amzn-s3-demo-bucket/*"
         ]
       }
     ]
   }'
```

2. 创建一个 IAM 角色。

   这样，Amazon RDS 就可以担任该 IAM 角色以访问您的 Amazon S3 存储桶。有关更多信息，请参阅《IAM 用户指南》中的创建向 IAM 用户委派权限的角色。

   我们建议在基于资源的策略中使用 `aws:SourceArn` 和 `aws:SourceAccount` 全局条件上下文键，以限制对特定资源的服务权限。这是防范混淆代理人问题最有效的方法。

   如果同时使用全局条件上下文键和包含账户 ID 的 `aws:SourceArn` 值，则 `aws:SourceAccount` 值和 `aws:SourceArn` 值中的账户在同一策略语句中使用时，必须使用相同的账户 ID。

   - 如果您想对单个资源进行跨服务访问，请使用 `aws:SourceArn`。
   - 如果您想允许该账户中的任何资源与跨服务使用操作相关联，请使用 `aws:SourceAccount`。

   在策略中，确保使用具有资源的完整 ARN 的 `aws:SourceArn` 全局条件上下文键。以下示例说明了如何使用 Amazon CLI 命令创建一个名为 `rds-s3-import-role` 的角色来实现该目的。

Example

对于 Linux、macOS 或 Unix：

```
aws iam create-role \
   --role-name rds-s3-import-role \
   --assume-role-policy-document '{
     "Version": "2012-10-17",
     "Statement": [
       {
         "Effect": "Allow",
         "Principal": {
            "Service": "rds.amazonaws.com"
          },
         "Action": "sts:AssumeRole",
         "Condition": {
             "StringEquals": {
                "aws:SourceAccount": "111122223333",
                "aws:SourceArn": "arn:aws:rds:us-east-1:111122223333:db:dbname"
                }
             }
       }
     ]
   }'
```

对于 Windows：

```
aws iam create-role ^
   --role-name rds-s3-import-role ^
   --assume-role-policy-document '{
     "Version": "2012-10-17",
     "Statement": [
       {
         "Effect": "Allow",
         "Principal": {
            "Service": "rds.amazonaws.com"
          },
         "Action": "sts:AssumeRole",
         "Condition": {
             "StringEquals": {
                "aws:SourceAccount": "111122223333",
                "aws:SourceArn": "arn:aws:rds:us-east-1:111122223333:db:dbname"
```

```
                 }
              }
        }
      ]
    }'
```

3. 将您创建的 IAM 策略附加到您创建的 IAM 角色。

   以下 Amazon CLI 命令将上一步中创建的策略附加到一个名为 `rds-s3-import-role` 的角色。请将 *`your-policy-arn`* 替换为您在前面的步骤中记下的策略 ARN。

   Example

   对于 Linux、macOS 或 Unix：

   ```
   aws iam attach-role-policy \
      --policy-arn your-policy-arn \
      --role-name rds-s3-import-role
   ```

   对于 Windows：

   ```
   aws iam attach-role-policy ^
      --policy-arn your-policy-arn ^
      --role-name rds-s3-import-role
   ```

4. 将该 IAM 角色添加到数据库实例中。

   您可以使用 Amazon Web Services Management Console 或 Amazon CLI 执行该操作，如下所述。

控制台

使用控制台为 PostgreSQL 数据库实例添加 IAM 角色

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 选择 PostgreSQL 数据库实例名称以显示其详细信息。
3. 在连接性和安全性选项卡上的管理 IAM 角色部分中，在向此实例添加 IAM 角色下选择要添加的角色。
4. 在 Feature (功能) 下，选择 s3Import。

5. 选择 Add role (添加角色)。

Amazon CLI

使用 CLI 为 PostgreSQL 数据库实例添加 IAM 角色

- 使用以下命令将角色添加到名为 my-db-instance 的 PostgreSQL 数据库实例中。将 *your-role-arn* 替换为您在上一步中记下的角色 ARN。使用 s3Import 作为 --feature-name 选项的值。

  Example

  对于 Linux、macOS 或 Unix：

```
aws rds add-role-to-db-instance \
   --db-instance-identifier my-db-instance \
   --feature-name s3Import \
   --role-arn your-role-arn   \
   --region your-region
```

  对于 Windows：

```
aws rds add-role-to-db-instance ^
   --db-instance-identifier my-db-instance ^
   --feature-name s3Import ^
   --role-arn your-role-arn ^
   --region your-region
```

RDS API

要使用 Amazon RDS API 为 PostgreSQL 数据库实例添加 IAM 角色，请调用 AddRoleToDBInstance 操作。

使用安全凭证访问 Amazon S3 存储桶

如果愿意，您可以使用安全凭证提供 Amazon S3 存储桶的访问权限，而不是使用 IAM 角色提供访问权限。这可以通过在 aws_s3.table_import_from_s3 函数调用中指定 credentials 参数来实现。

credentials 参数是 aws_commons._aws_credentials_1 类型的结构，其中包含 Amazon 凭证。使用 aws_commons.create_aws_credentials 函数设置 aws_commons._aws_credentials_1 结构中的访问密钥和私有密钥，如下所示。

```
postgres=> SELECT aws_commons.create_aws_credentials(
   'sample_access_key', 'sample_secret_key', '')
AS creds \gset
```

在创建 aws_commons._aws_credentials_1 结构后，请将 aws_s3.table_import_from_s3 函数与 credentials 参数结合使用来导入数据，如下所示。

```
postgres=> SELECT aws_s3.table_import_from_s3(
   't', '', '(format csv)',
   :'s3_uri',
   :'creds'
);
```

也可以在 aws_commons.create_aws_credentials 函数调用中以内联方式包括 aws_s3.table_import_from_s3 函数调用。

```
postgres=> SELECT aws_s3.table_import_from_s3(
   't', '', '(format csv)',
   :'s3_uri',
   aws_commons.create_aws_credentials('sample_access_key', 'sample_secret_key', '')
);
```

Amazon S3 访问故障排除

如果在尝试从 Amazon S3 导入数据时遇到连接问题，请参阅以下内容以了解相应的建议：

- 对 Amazon RDS 身份和访问权限问题进行故障排除
- Amazon Simple Storage Service 用户指南中的排查 Amazon S3 的问题
- IAM 用户指南中的排查 Amazon S3 和 IAM 的问题

将数据从 Amazon S3 导入到 RDS for PostgreSQL 数据库实例

您可以使用 aws_S3 扩展的 table_import_from_s3 函数从 Amazon S3 桶导入数据。有关参考信息，请参阅aws_s3.table_import_from_s3。

> **Note**
>
> 以下示例使用 IAM 角色方法以允许访问 Amazon S3 桶。因此，aws_s3.table_import_from_s3 函数调用不包括凭据参数。

下面显示典型示例。

```
postgres=> SELECT aws_s3.table_import_from_s3(
   't1',
   '',
   '(format csv)',
   :'s3_uri'
);
```

下面是参数：

- t1 – 将数据复制到的 PostgreSQL 数据库实例中的表的名称。
- '' – 数据库表中的列的可选列表。您可以使用此参数来指示哪些 S3 数据列进入哪些表列中。如果未指定任何列，则会将所有列复制到表中。有关使用列列表的示例，请参阅 导入使用自定义分隔符的 Amazon S3 文件。
- (format csv) – PostgreSQL COPY 参数。复制过程使用 PostgreSQL COPY 命令的参数和格式以导入数据。格式选择包括本例中所示的逗号分隔值（CSV），以及文本和二进制。原定设置为文本。
- s3_uri – 包含标识 Amazon S3 文件的信息的结构。有关使用 aws_commons.create_s3_uri 函数创建 s3_uri 结构的示例，请参阅 从 Amazon S3 数据导入数据概述。

有关此函数的更多信息，请参阅 aws_s3.table_import_from_s3。

aws_s3.table_import_from_s3 函数返回文本。要指定要从 Amazon S3 存储桶中导入的其他类型的文件，请参阅以下示例之一。

> **Note**
>
> 导入 0 字节文件将导致错误。

主题

- 导入使用自定义分隔符的 Amazon S3 文件
- 导入 Amazon S3 压缩 (gzip) 文件
- 导入编码的 Amazon S3 文件

导入使用自定义分隔符的 Amazon S3 文件

以下示例说明了如何导入使用自定义分隔符的文件。它还说明如何使用 column_list 函数的 aws_s3.table_import_from_s3 参数来控制将数据放置在数据库表中的哪个位置。

在此示例中，假定将以下信息组织到 Amazon S3 文件中的竖线分隔列中。

```
1|foo1|bar1|elephant1
2|foo2|bar2|elephant2
3|foo3|bar3|elephant3
4|foo4|bar4|elephant4
...
```

导入使用自定义分隔符的文件

1. 在数据库中为导入的数据创建一个表。

   ```
   postgres=> CREATE TABLE test (a text, b text, c text, d text, e text);
   ```

2. 使用以下形式的 aws_s3.table_import_from_s3 函数从 Amazon S3 文件导入数据。

   您可以在 aws_commons.create_s3_uri 函数调用中以内联方式包括 aws_s3.table_import_from_s3 函数调用来指定文件。

   ```
   postgres=> SELECT aws_s3.table_import_from_s3(
      'test',
      'a,b,d,e',
      'DELIMITER ''|''',
      aws_commons.create_s3_uri('amzn-s3-demo-bucket', 'pipeDelimitedSampleFile', 'us-
   east-2')
   );
   ```

数据现在位于表的以下列中。

```
postgres=> SELECT * FROM test;
```

```
a | b | c | d | e
---+------+---+---+------+-----------
1 | foo1 | | bar1 | elephant1
2 | foo2 | | bar2 | elephant2
3 | foo3 | | bar3 | elephant3
4 | foo4 | | bar4 | elephant4
```

导入 Amazon S3 压缩 (gzip) 文件

以下示例说明如何从 Amazon S3 导入使用 gzip 压缩的文件。导入的文件需要具有以下 Amazon S3 元数据：

- 键：`Content-Encoding`
- 值：`gzip`

如果使用 Amazon Web Services Management Console 上传文件，则元数据通常由系统应用。有关使用 Amazon Web Services Management Console、Amazon CLI 或 API 将文件上传到 Amazon S3 的信息，请参阅《Amazon Simple Storage Service 用户指南》中的上传对象。

有关 Amazon S3 元数据的更多信息以及有关系统提供的元数据的详细信息，请参阅《Amazon Simple Storage Service 用户指南》中的在 Amazon S3 控制台中编辑对象元数据。

将 gzip 文件导入到 RDS for PostgreSQL 数据库实例，如下所示。

```
postgres=> CREATE TABLE test_gzip(id int, a text, b text, c text, d text);
postgres=> SELECT aws_s3.table_import_from_s3(
 'test_gzip', '', '(format csv)',
 'amzn-s3-demo-bucket', 'test-data.gz', 'us-east-2'
);
```

导入编码的 Amazon S3 文件

以下示例说明如何从 Amazon S3 导入具有 Windows-1252 编码的文件。

```
postgres=> SELECT aws_s3.table_import_from_s3(
 'test_table', '', 'encoding ''WIN1252''',
 aws_commons.create_s3_uri('amzn-s3-demo-bucket', 'SampleFile', 'us-east-2')
);
```

函数参考

函数

- aws_s3.table_import_from_s3
- aws_commons.create_s3_uri
- aws_commons.create_aws_credentials

aws_s3.table_import_from_s3

将 Amazon S3 数据导入到 Amazon RDS 表中。aws_s3 扩展提供 aws_s3.table_import_from_s3 函数。返回值为文本。

语法

必需的参数为 table_name、column_list 和 options。这些标识数据库表并指定如何将数据复制到表中。

您还可以使用以下参数：

- s3_info 参数指定要导入的 Amazon S3 文件。在您使用此参数时，IAM 角色为 PostgreSQL 数据库实例提供访问 Amazon S3 的权限。

```
aws_s3.table_import_from_s3 (
   table_name text,
   column_list text,
   options text,
   s3_info aws_commons._s3_uri_1
)
```

- credentials 参数指定凭证以访问 Amazon S3。在您使用此参数时，不使用 IAM 角色。

```
aws_s3.table_import_from_s3 (
   table_name text,
   column_list text,
   options text,
   s3_info aws_commons._s3_uri_1,
   credentials aws_commons._aws_credentials_1
)
```

参数

table_name

包含要将数据导入到的 PostgreSQL 数据库表的名称的必需文本字符串。

column_list

包含要将数据复制到的 PostgreSQL 数据库表列的可选列表的必需文本字符串。如果此字符串为空，将使用表的所有列。有关示例，请参阅导入使用自定义分隔符的 Amazon S3 文件。

options

包含 PostgreSQL `COPY` 命令的参数的必需文本字符串。这些参数指定如何将数据复制到 PostgreSQL 表中。有关更多详细信息，请参阅 PostgreSQL COPY 文档。

s3_info

包含有关 S3 对象的以下信息的 `aws_commons._s3_uri_1` 复合类型：

- `bucket` – 包含文件的 Amazon S3 存储桶的名称。
- `file_path` – 包含文件路径的 Amazon S3 文件名。
- `region` – 文件所在的Amazon区域。有关 Amazon 区域名称和关联值的列表，请参阅区域、可用区和 Local Zones。

凭证

包含以下用于导入操作的凭证的 `aws_commons._aws_credentials_1` 复合类型：

- 访问密钥
- 私有密钥
- 会话令牌

有关创建 `aws_commons._aws_credentials_1` 复合结构的信息，请参阅 aws_commons.create_aws_credentials。

替代语法

为帮助进行测试，您可以使用一组扩展的参数而非 `s3_info` 和 `credentials` 参数。下面是 `aws_s3.table_import_from_s3` 函数的其他语法变化。

- 不使用 `s3_info` 参数来标识 Amazon S3 文件，而使用 `bucket`、`file_path` 和 `region` 参数的组合。使用此形式的函数，IAM 角色在 PostgreSQL 数据库实例上提供访问 Amazon S3 的权限。

```
aws_s3.table_import_from_s3 (
   table_name text,
   column_list text,
   options text,
   bucket text,
   file_path text,
   region text
)
```

- 不使用 `credentials` 参数来指定 Amazon S3 访问权限，而使用 `access_key`、`session_key` 和 `session_token` 参数的组合。

```
aws_s3.table_import_from_s3 (
   table_name text,
   column_list text,
   options text,
   bucket text,
   file_path text,
   region text,
   access_key text,
   secret_key text,
   session_token text
)
```

替代参数

bucket

包含 Amazon S3 存储桶（其中包含文件）的名称的文本字符串。

file_path

包含 Amazon S3 文件名（包含文件路径）的文本字符串。

region

标识文件的 Amazon Web Services 区域位置的文本字符串。有关 Amazon Web Services 区域名称和关联值的列表，请参阅区域、可用区和 Local Zones。

access_key

包含用于导入操作的访问密钥的文本字符串。默认值为 NULL。

secret_key

包含用于导入操作的私有密钥的文本字符串。默认值为 NULL。

session_token

（可选）包含用于导入操作的会话密钥的文本字符串。默认值为 NULL。

aws_commons.create_s3_uri

创建 `aws_commons._s3_uri_1` 结构来保存 Amazon S3 文件信息。在 `aws_commons.create_s3_uri` 函数的 `s3_info` 参数中使用 [aws_s3.table_import_from_s3](#) 函数的结果。

语法

```
aws_commons.create_s3_uri(
   bucket text,
   file_path text,
   region text
)
```

参数

bucket

包含文件的 Amazon S3 存储桶名称的必需文本字符串。

file_path

包含 Amazon S3 文件名（包含文件路径）的必填文本字符串。

region

一个包含文件所在的 Amazon Web Services 区域的必需文本字符串。有关 Amazon Web Services 区域名称和关联值的列表，请参阅[区域、可用区和 Local Zones](#)。

aws_commons.create_aws_credentials

在 `aws_commons._aws_credentials_1` 结构中设置访问密钥和私有密钥。在 `aws_commons.create_aws_credentials` 函数的 `credentials` 参数中使用 [aws_s3.table_import_from_s3](#) 函数的结果。

语法

```
aws_commons.create_aws_credentials(
    access_key text,
    secret_key text,
    session_token text
)
```

参数

access_key

包含用于导入 Amazon S3 文件的访问密钥的必需文本字符串。默认值为 NULL。

secret_key

包含用于导入 Amazon S3 文件的私有密钥的必需文本字符串。默认值为 NULL。

session_token

包含用于导入 Amazon S3 文件的会话令牌的可选文本字符串。默认值为 NULL。如果您提供了可选的 `session_token`，则可以使用临时凭证。

在数据库实例之间传输 PostgreSQL 数据库

通过将 PostgreSQL 可传输数据库用于 Amazon RDS，您可以在两个数据库实例之间传输 PostgreSQL 数据库。这是在不同数据库实例之间迁移大型数据库的一种非常快的方法。要使用该方法，两个数据库实例必须运行相同的 PostgreSQL 主要版本。

此功能要求您在源和目标数据库实例上均安装 `pg_transport` 扩展。`pg_transport` 扩展提供了一种以最少的处理来移动数据库文件的物理传输机制。该机制移动数据的速度比传统的转储和加载过程快得多，并减少了停机时间。

> **Note**
>
> PostgreSQL 可传输数据库在 RDS for PostgreSQL 11.5 和更高版本以及 RDS for PostgreSQL 版本 10.10 和更高版本中提供。

要将 PostgreSQL 数据库实例从一个 RDS for PostgreSQL 数据库实例传输到另一个，首先设置源实例和目标实例，如 设置数据库实例以进行传输 中详述。然后，您可以使用 传输 PostgreSQL 数据库 中所述的函数传输数据库。

主题

在数据库传输期间发生的情况

PostgreSQL 可传输数据库功能使用拉取模式，将数据库从源数据库实例导入到目标数据库实例。`transport.import_from_server` 函数在目标数据库实例上创建传输中的数据库。在传输期间，无法在目标数据库实例上访问传输中的数据库。

在传输开始时，源数据库上的所有当前会话将会结束。源数据库实例上的源数据库以外的任何数据库不受传输影响。

源数据库被置于特殊只读模式。在处于该模式时，您可以连接到源数据库并运行只读查询。不过，阻止启用了写入的查询和某些其他类型的命令。仅正在传输的特定源数据库受这些限制的影响。

在传输期间，您无法将目标数据库实例还原到某个时间点。这是因为传输不是事务性的，并且不使用 PostgreSQL 预写日志记录更改。如果目标数据库实例启用了自动备份，则会在传输完成后自动创建备份。时间点还原适用于备份完成后的时间。

如果传输失败，`pg_transport` 扩展将尝试撤消对源和目标数据库实例的所有更改。这包括删除目标的部分传输的数据库。根据失败类型，源数据库可能会继续拒绝启用了写入的查询。如果发生这种情况，请使用以下命令允许启用了写入的查询。

```
ALTER DATABASE db-name SET default_transaction_read_only = false;
```

使用 PostgreSQL 可传输数据库的限制

可传输数据库具有以下限制：

- 只读副本 – 您无法在只读副本或只读副本的父实例上使用可传输数据库。
- 不支持的列类型 – 您不能在打算使用该方法传输的任何数据库表中使用 `reg` 数据类型。这些类型取决于系统目录对象 ID (OID)，这些 ID 在传输过程中经常发生变化。
- 表空间 – 所有源数据库对象必须位于默认 `pg_default` 表空间中。

- 兼容性 – 源和目标数据库实例必须运行相同的 PostgreSQL 主要版本。
- 扩展 - 源数据库实例只能安装 `pg_transport`。
- 角色和 ACL – 源数据库的访问权限和所有权信息不会传输到目标数据库。所有数据库对象是由传输的本地目标用户创建和拥有的。
- 并发传输 - 如果工件线程进程配置正确，单个数据库实例最多可支持 32 个并发传输，包括导入和导出。
- 仅限 RDS for PostgreSQL 数据库实例 - PostgreSQL 可传输数据库仅在 RDS for PostgreSQL 数据库实例上支持。您不能将其与 Amazon EC2 上运行的本地数据库或数据库一起使用。

设置以传输 PostgreSQL 数据库

在开始之前，请确保您的 RDS for PostgreSQL 数据库实例满足以下要求：

- 源和目标 RDS for PostgreSQL 数据库实例必须运行相同的 PostgreSQL 版本。
- 目标数据库不能拥有与要传输的源数据库同名的数据库。
- 您用来运行传输的账户需要源数据库和目标数据库的 `rds_superuser` 权限。
- 源数据库实例的安全组必须允许来自目标数据库实例的入站访问。如果源和目标数据库实例均位于 VPC 中，则可能已经是这种情况。有关安全组的更多信息，请参阅 使用安全组控制访问权限。

将数据库从源数据库实例传输到目标数据库实例需要对与每个实例关联的数据库参数组进行若干更改。这意味着您必须为源数据库实例创建自定义数据库参数组，同时为目标数据库实例创建自定义数据库参数组。

> **Note**
>
> 如果您的数据库实例已使用自定义数据库参数组进行配置，则可以从以下过程中的步骤 2 开始。

配置用于传输数据库的自定义数据库组参数

对于以下步骤，请使用具有 `rds_superuser` 特权的账户。

1. 如果源和目标数据库实例使用默认的数据库参数组，您需要使用适合您的实例的版本创建自定义数据库参数组。这样便可更改多个参数的值。有关更多信息，请参阅 Amazon RDS 的参数组。
2. 在自定义数据库参数组中，更改以下参数的值：

- shared_preload_libraries - 将 pg_transport 添加到库列表。
- pg_transport.num_workers - 默认值为 3。根据数据库的需要增加或减少该值。对于 200 GB 的数据库，我们建议该值不大于 8。请记住，如果增加该参数的默认值，还应增加 max_worker_processes 的值。
- pg_transport.work_mem - 默认值为 128 MB 或 256 MB，具体视 PostgreSQL 版本而定。默认设置通常可以保持不变。
- max_worker_processes – 需要使用以下计算来设置此参数的值：

  ```
  (3 * pg_transport.num_workers) + 9
  ```

  目标上需要此值来处理传输中涉及的各种后台工作进程。要了解有关 max_worker_processes, 的更多信息，请参阅 PostgreSQL 文档中的资源消耗。

有关 pg_transport 参数的更多信息，请参阅 可传输数据库参数参考 。

3. 重启源 RDS for PostgreSQL 数据库实例和目标实例以使参数的设置生效。
4. 连接到 RDS for PostgreSQL 源数据库实例。

   ```
   psql --host=source-instance.111122223333.aws-region.rds.amazonaws.com --port=5432
    --username=postgres --password
   ```

5. 从数据库实例的公共架构中删除无关的扩展。实际传输操作期间只允许 pg_transport 扩展。
6. 按如下方式安装 pg_transport 扩展：

   ```
   postgres=> CREATE EXTENSION pg_transport;
   CREATE EXTENSION
   ```

7. 连接到 RDS for PostgreSQL 目标数据库实例。删除任何无关的扩展，然后安装 pg_transport 扩展。

   ```
   postgres=> CREATE EXTENSION pg_transport;
   CREATE EXTENSION
   ```

将 PostgreSQL 数据库从源传输到目标

在完成 设置以传输 PostgreSQL 数据库 中所述的过程后，您可以启动传输。为此，请在目标数据库实例上运行 transport.import_from_server 函数。在下面的语法中，您可以找到函数参数。

```
SELECT transport.import_from_server(
   'source-db-instance-endpoint',
    source-db-instance-port,
   'source-db-instance-user',
   'source-user-password',
   'source-database-name',
   'destination-user-password',
   false);
```

示例中显示的 false 值告知函数这不是一次试运行。要测试传输设置，您可以指定调用函数时 dry_run 选项的 true，如下所示：

```
postgres=> SELECT transport.import_from_server(
    'docs-lab-source-db.666666666666aws-region.rds.amazonaws.com', 5432,
    'postgres', '********', 'labdb', '******', true);
INFO:  Starting dry-run of import of database "labdb".
INFO:  Created connections to remote database        (took 0.03 seconds).
INFO:  Checked remote cluster compatibility          (took 0.05 seconds).
INFO:  Dry-run complete                         (took 0.08 seconds total).
 import_from_server
--------------------

(1 row)
```

由于 pg_transport.timing 参数设置为其默认值 true，因此输出 INFO 行。运行该命令时将 dry_run 设置为 false，源数据库将导入到目标数据库，如下所示：

```
INFO:  Starting import of database "labdb".
INFO:  Created connections to remote database        (took 0.02 seconds).
INFO:  Marked remote database as read only           (took 0.13 seconds).
INFO:  Checked remote cluster compatibility          (took 0.03 seconds).
INFO:  Signaled creation of PITR blackout window     (took 2.01 seconds).
INFO:  Applied remote database schema pre-data       (took 0.50 seconds).
INFO:  Created connections to local cluster          (took 0.01 seconds).
INFO:  Locked down destination database              (took 0.00 seconds).
INFO:  Completed transfer of database files          (took 0.24 seconds).
INFO:  Completed clean up                            (took 1.02 seconds).
```

```
INFO:  Physical transport complete              (took 3.97 seconds total).
import_from_server
--------------------
(1 row)
```

该函数要求您提供数据库用户密码。因此，在传输完成后，我们建议您更改使用的用户角色的密码。或者，您可以使用 SQL 绑定变量创建临时用户角色。使用这些临时角色进行传输，然后丢弃这些角色。

如果传输失败，您可能会看到类似于以下内容的错误消息：

```
pg_transport.num_workers=8 25% of files transported failed to download file data
```

“failed to download file data”（下载文件数据失败）错误消息表示没有根据数据库的大小正确设置工件线程进程的数量。您可能需要增加或减少为 `pg_transport.num_workers` 设置的值。每次失败都会报告完成的百分比，因此您可以看到更改的影响。例如，在一个案例中将设置从 8 更改为 4 会导致以下情况：

```
pg_transport.num_workers=4 75% of files transported failed to download file data
```

请记住，在传输过程中还会考虑 `max_worker_processes` 参数。换句话说，您可能需要同时修改 `pg_transport.num_workers` 和 `max_worker_processes` 才能成功传输数据库。图示的例子当 `pg_transport.num_workers` 设置为 2 后最终生效：

```
pg_transport.num_workers=2 100% of files transported
```

有关 `transport.import_from_server` 函数及其参数的更多信息，请参阅 可传输数据库函数参考。

可传输数据库函数参考

`transport.import_from_server` 函数将 PostgreSQL 数据库从源数据库实例导入到目标数据库实例以传输该数据库。它使用物理数据库连接传输机制以执行该操作。

在开始传输之前，此函数将验证源和目标数据库实例的版本是否相同，以及是否与迁移兼容。它还确认目标数据库实例有足够的空间容纳源数据库实例。

语法

```
transport.import_from_server(
```

```
    host text,
    port int,
    username text,
    password text,
    database text,
    local_password text,
    dry_run bool
)
```

返回值

无。

参数

您可以在下表中找到 `transport.import_from_server` 函数参数说明。

参数	描述
`host`	源数据库实例的端点。
`port`	表示源数据库实例端口的整数。 PostgreSQL 数据库实例通常使用端口 5432。
`username`	源数据库实例的用户。该用户必须是 `rds_superuser` 角色的成员。
`password`	源数据库实例的用户密码。
`database`	源数据库实例中要传输的数据库的名称。
`local_password`	目标数据库实例的当前用户的本地密码。该用户必须是 `rds_superuser` 角色的成员。
`dry_run`	可选的布尔值，指定是否执行试运行。默认值为 `false`，这表示传输继续进行。要确认源和目标数据库实例之间的兼容性而不执行实际传输，请将 `dry_run` 设置为 `true`。

示例

有关示例，请参阅 [将 PostgreSQL 数据库从源传输到目标](#)。

可传输数据库参数参考

有多个参数可控制 `pg_transport` 扩展的行为。在下文中，您可以找到这些参数的说明。

`pg_transport.num_workers`

用于传输过程的工件线程数。默认值为 3。有效值为 1-32。即使是最大的数据库，传输通常需要也少于 8 个工件线程。在传输期间，目标数据库实例上此设置的值供目标和源使用。

`pg_transport.timing`

指定是否在传输期间报告计时信息。默认值为 `true`，表示报告计时信息。我们建议您将该参数设置为 `true`，以便监控进度。有关示例输出，请参阅 将 PostgreSQL 数据库从源传输到目标。

`pg_transport.work_mem`

为每个工作线程分配的最大内存量。默认值为 131072 千字节 (KB) 或 262144 KB (256 MB)，具体视 PostgreSQL 版本而定。最小值为 64 兆字节 (65536 KB)。有效值是以二进制单位表示的千字节 (KB)，其中 1 KB = 1024 字节。

传输使用的内存不能超过该参数中指定的值。即使是大型数据库传输，通常所需的每个工件线程内存少于 256 MB (262144 KB)。

将数据从 RDS for PostgreSQL 数据库实例导出到 Amazon S3

您可以从 RDS for PostgreSQL 数据库实例中查询数据，并将数据直接导出到存储在 Amazon S3 存储桶中的文件中。为此，您首先要安装 RDS for PostgreSQL aws_s3 扩展。该扩展为您提供了用于将查询结果导出到 Amazon S3 的函数。接下来，您可以了解如何安装扩展以及如何将数据导出到 Amazon S3。

> **Note**
>
> 不支持跨账户导出到 Amazon S3。

所有当前可用的 RDS for PostgreSQL 版本都支持将数据导出到 Amazon Simple Storage Service。有关详细版本信息，请参阅《Amazon RDS for PostgreSQL 版本注释》中的 Amazon RDS for PostgreSQL 更新。

如果您没有为导出设置桶，请参阅《Amazon Simple Storage Service 用户指南》中的以下主题。

- 设置 Amazon S3
- 创建存储桶

默认情况下，从 RDS for PostgreSQL 导出到 Amazon S3 的数据使用具有 Amazon 托管式密钥的服务器端加密。如果您使用存储桶加密，则 Amazon S3 存储桶必须使用 Amazon Key Management Service（Amazon KMS）密钥（SSE-KMS）进行加密。目前，不支持使用 Amazon S3 托管式密钥（SSE-S3）加密的存储桶。

> **Note**
>
> 您可以使用Amazon Web Services Management Console、Amazon CLI 或 Amazon RDS API 将数据库快照数据保存到 Amazon S3。有关更多信息，请参阅 将 Amazon RDS 的数据库快照数据导出到 Amazon S3。

主题

- 安装 aws_s3 扩展名
- 将数据导出到 Amazon S3 概述
- 指定要导出到的 Amazon S3 文件路径

- 设置 Amazon S3 存储桶的访问权限
- 使用 aws_s3.query_export_to_s3 函数导出查询数据
- 函数参考
- Amazon S3 访问故障排除

安装 aws_s3 扩展名

在将 Amazon Simple Storage Service 用于 RDS for PostgreSQL 数据库实例之前，您需要安装 aws_s3 扩展。此扩展提供将数据从 RDS for PostgreSQL 数据库实例中导出到 Amazon S3 桶的函数。它还提供从 Amazon S3 导入数据的函数。有关更多信息，请参阅 将 Amazon S3 中的数据导入到 RDS for PostgreSQL 数据库实例。aws_s3 扩展依赖于 aws_commons 扩展（需要时自动安装）中的一些帮助程序函数。

安装 **aws_s3** 扩展

1. 使用 psql（或 pgAdmin）以具有 rds_superuser 权限的用户身份，连接到 RDS for PostgreSQL 数据库实例。如果您在设置过程中保留原定设置名称，则以 postgres 进行连接。

```
psql --host=111122223333.aws-region.rds.amazonaws.com --port=5432 --
username=postgres --password
```

2. 要安装扩展，请运行以下命令。

```
postgres=> CREATE EXTENSION aws_s3 CASCADE;
NOTICE: installing required extension "aws_commons"
CREATE EXTENSION
```

3. 要验证扩展是否已安装，可以使用 psql \dx 元命令。

```
postgres=> \dx
       List of installed extensions
    Name     | Version |   Schema   |                 Description
-------------+---------+------------+---------------------------------------------
 aws_commons | 1.2     | public     | Common data types across Amazon services
 aws_s3      | 1.1     | public     | Amazon S3 extension for importing data from
 S3
 plpgsql     | 1.0     | pg_catalog | PL/pgSQL procedural language
(3 rows)
```

从 Amazon S3 导入数据和将数据导出到 Amazon S3 的函数现在可供使用。

验证您的 RDS for PostgreSQL 版本支持导出到 Amazon S3

您可以使用 `describe-db-engine-versions` 命令验证 RDS for PostgreSQL 版本是否支持导出到 Amazon S3。以下示例验证对版本 10.14 的支持。

```
aws rds describe-db-engine-versions --region us-east-1
--engine postgres --engine-version 10.14 | grep s3Export
```

如果输出包含字符串 `"s3Export"`，则该引擎支持 Amazon S3 导出。否则，该引擎不支持它们。

将数据导出到 Amazon S3 概述

要将存储在 RDS for PostgreSQL 数据库中的数据导出到 Amazon S3 存储桶，请使用以下过程。

将 RDS for PostgreSQL 数据导出到 S3

1. 标识要用于导出数据的 Amazon S3 文件路径。有关此过程的详细信息，请参阅指定要导出到的 Amazon S3 文件路径。
2. 提供权限以访问 Amazon S3 存储桶。

   要将数据导出到 Amazon S3 文件，请为 RDS for PostgreSQL 数据库实例提供权限以访问导出将用于存储的 Amazon S3 存储桶。此操作包括以下步骤：

   1. 创建一个 IAM 策略，该策略提供对要导出到的 Amazon S3 存储桶的访问权限。
   2. 创建一个 IAM 角色。
   3. 将您创建的策略附在您创建的角色上。
   4. 将此 IAM 角色添加到数据库实例。

   有关此过程的详细信息，请参阅设置 Amazon S3 存储桶的访问权限。
3. 标识数据库查询以获取数据。通过调用 `aws_s3.query_export_to_s3` 函数来导出查询数据。

   在完成上述准备任务后，请使用 aws_s3.query_export_to_s3 函数将查询结果导出到 Amazon S3。有关此过程的详细信息，请参阅使用 aws_s3.query_export_to_s3 函数导出查询数据。

指定要导出到的 Amazon S3 文件路径

指定以下信息以标识要将数据导出到的 Amazon S3 中的位置：

- 存储桶名称 – 存储桶是 Amazon S3 对象或文件的容器。

  有关使用 Amazon S3 存储数据的更多信息，请参阅《Amazon Simple Storage Service 用户指南》中的创建存储桶和使用对象。
- 文件路径 – 文件路径标识 Amazon S3 存储桶中存储导出的位置。文件路径包含以下内容：
  - 用于标识虚拟文件夹路径的可选路径前缀。
  - 用于标识要存储的一个或多个文件的文件前缀。较大的导出将存储在多个文件中，每个文件的最大大小约为 6 GB。其他文件名具有相同的文件前缀，但追加了 `_partXX`。*XX* 表示 2，然后表示 3，依此类推。

  例如，具有 `exports` 文件夹和 `query-1-export` 文件前缀的文件路径为 `/exports/query-1-export`。
- Amazon区域（可选）– Amazon S3 存储桶所在的Amazon区域。如果您未指定Amazon区域值，则 Amazon RDS 会将文件保存到导出的数据库实例所在的同一Amazon区域中的 Amazon S3 中。

> **Note**
>
> 目前，Amazon 区域必须与导出的数据库实例的区域相同。

  有关 Amazon 区域名称和关联值的列表，请参阅区域、可用区和 Local Zones。

要保存有关导出的存储位置的 Amazon S3 文件信息，您可以使用 aws_commons.create_s3_uri 函数创建一个 `aws_commons._s3_uri_1` 复合结构，如下所示。

```
psql=> SELECT aws_commons.create_s3_uri(
   'amzn-s3-demo-bucket',
   'sample-filepath',
   'us-west-2'
) AS s3_uri_1 \gset
```

稍后，您可以在对 `s3_uri_1` 函数的调用中将该 aws_s3.query_export_to_s3 值作为参数提供。有关示例，请参阅 使用 aws_s3.query_export_to_s3 函数导出查询数据。

设置 Amazon S3 存储桶的访问权限

要将数据导出到 Amazon S3，请为 PostgreSQL 数据库实例提供权限以访问文件将保存到的 Amazon S3 存储桶。

为此，请使用以下过程。

通过 IAM 角色向 PostgreSQL 数据库实例授予访问 Amazon S3 的权限

1. 创建一个 IAM 策略。

   该策略提供存储桶和对象权限，以允许 PostgreSQL 数据库实例访问 Amazon S3。

   在创建此策略的过程中，请执行以下步骤：

   a. 在策略中包含以下必需操作，以允许将文件从 PostgreSQL 数据库实例传输到 Amazon S3 存储桶：

      - `s3:PutObject`
      - `s3:AbortMultipartUpload`

   b. 包含用于标识 Amazon S3 存储桶以及其中的对象的 Amazon Resource Name (ARN)。用于访问 Amazon S3 的 ARN 格式为：`arn:aws:s3:::`*`amzn-s3-demo-bucket`*`/*`

   有关为 Amazon RDS for PostgreSQL 创建 IAM 策略的更多信息，请参阅 创建和使用适用于 IAM 数据库访问的 IAM 策略。另请参阅 IAM 用户指南中的教程：创建和附加您的第一个客户托管式策略。

   以下 Amazon CLI 命令使用这些选项创建一个名为 `rds-s3-export-policy` 的 IAM 策略。该策略授予对名为 *`amzn-s3-demo-bucket`* 的存储桶的访问权限。

   > **Warning**
   >
   > 我们建议您在一个私有 VPC 中设置数据库，该 VPC 配置了用于访问特定存储桶的端点策略。有关更多信息，请参阅 Amazon VPC 用户指南 中的对 Amazon S3 使用端点策略。
   > 强烈建议您不要创建具有所有资源访问权限的策略。此访问权限可能会对数据安全造成威胁。如果您使用 `S3:PutObject` 创建一个向 `"Resource":"*"` 授予对所有资源的访问权限的策略，则具有导出特权的用户可以将数据导出到您账户中的所有存储桶。此外，用户可以将数据导出到 Amazon 区域内的任何可公开写入的存储桶。

在您创建策略之后，请记下策略的 Amazon Resource Name (ARN)。在将策略附加到 IAM 角色时，您在后面的步骤中需要使用 ARN。

```
aws iam create-policy  --policy-name rds-s3-export-policy  --policy-document '{
     "Version": "2012-10-17",
     "Statement": [
       {
         "Sid": "s3export",
         "Action": [
           "s3:PutObject*",
           "s3:ListBucket",
           "s3:GetObject*",
           "s3:DeleteObject*",
           "s3:GetBucketLocation",
           "s3:AbortMultipartUpload"
         ],
         "Effect": "Allow",
         "Resource": [
           "arn:aws:s3:::amzn-s3-demo-bucket/*"
         ]
       }
     ]
   }'
```

2. 创建一个 IAM 角色。

   这样，Amazon RDS 就可以担任该 IAM 角色以代表您访问 Amazon S3 存储桶。有关更多信息，请参阅 IAM 用户指南中的创建向 IAM 用户委派权限的角色。

   我们建议在基于资源的策略中使用 aws:SourceArn 和 aws:SourceAccount 全局条件上下文键，以限制对特定资源的服务权限。这是防范混淆代理人问题最有效的方法。

   如果同时使用全局条件上下文键和包含账户 ID 的 aws:SourceArn 值，则 aws:SourceAccount 值和 aws:SourceArn 值中的账户在同一策略语句中使用时，必须使用相同的账户 ID。

   - 如果您想对单个资源进行跨服务访问，请使用 aws:SourceArn。
   - 如果您想允许该账户中的任何资源与跨服务使用操作相关联，请使用 aws:SourceAccount。

在策略中，确保使用具有资源的完整 ARN 的 `aws:SourceArn` 全局条件上下文键。以下示例说明了如何使用 Amazon CLI 命令创建一个名为 `rds-s3-export-role` 的角色来实现该目的。

Example

对于 Linux、macOS 或 Unix：

```
aws iam create-role  \
    --role-name rds-s3-export-role  \
    --assume-role-policy-document '{
     "Version": "2012-10-17",
     "Statement": [
       {
         "Effect": "Allow",
         "Principal": {
            "Service": "rds.amazonaws.com"
          },
         "Action": "sts:AssumeRole",
         "Condition": {
             "StringEquals": {
                "aws:SourceAccount": "111122223333",
                "aws:SourceArn": "arn:aws:rds:us-east-1:111122223333:db:dbname"
                }
             }
       }
     ]
   }'
```

对于 Windows：

```
aws iam create-role  ^
    --role-name rds-s3-export-role  ^
    --assume-role-policy-document '{
     "Version": "2012-10-17",
     "Statement": [
       {
         "Effect": "Allow",
         "Principal": {
            "Service": "rds.amazonaws.com"
          },
         "Action": "sts:AssumeRole",
```

```
            "Condition": {
                "StringEquals": {
                   "aws:SourceAccount": "111122223333",
                   "aws:SourceArn": "arn:aws:rds:us-east-1:111122223333:db:dbname"
                   }
                }
          }
        ]
     }'
```

3. 将您创建的 IAM 策略附加到您创建的 IAM 角色。

   以下 Amazon CLI 命令将以前创建的策略附加到一个名为 `rds-s3-export-role.` 的角色。请将 *`your-policy-arn`* 替换为您在前面的步骤中记下的策略 ARN。

   ```
   aws iam attach-role-policy  --policy-arn your-policy-arn  --role-name rds-s3-
   export-role
   ```

4. 将该 IAM 角色添加到数据库实例中。您可以使用 Amazon Web Services Management Console 或 Amazon CLI 执行该操作，如下所述。

控制台

使用控制台为 PostgreSQL 数据库实例添加 IAM 角色

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 选择 PostgreSQL 数据库实例名称以显示其详细信息。
3. 在 Connectivity & security (连接性和安全性) 选项卡上的 Manage IAM roles (管理 IAM 角色) 部分中，在 Add IAM roles to this instance (向此实例添加 IAM 角色) 下选择要添加的角色。
4. 在 Feature (功能) 下，选择 s3Export。
5. 选择 Add role (添加角色)。

Amazon CLI

使用 CLI 为 PostgreSQL 数据库实例添加 IAM 角色

- 使用以下命令将角色添加到名为 `my-db-instance` 的 PostgreSQL 数据库实例中。将 *`your-role-arn`* 替换为您在上一步中记下的角色 ARN。使用 `s3Export` 作为 `--feature-name` 选项的值。

  Example

  对于 Linux、macOS 或 Unix：

```
aws rds add-role-to-db-instance \
   --db-instance-identifier my-db-instance \
   --feature-name s3Export \
   --role-arn your-role-arn   \
   --region your-region
```

  对于 Windows：

```
aws rds add-role-to-db-instance ^
   --db-instance-identifier my-db-instance ^
   --feature-name s3Export ^
   --role-arn your-role-arn ^
   --region your-region
```

使用 aws_s3.query_export_to_s3 函数导出查询数据

通过调用 aws_s3.query_export_to_s3 函数将 PostgreSQL 数据导出到 Amazon S3。

主题

先决条件

在使用 aws_s3.query_export_to_s3 函数之前，请确保完成以下先决条件：

- 安装所需的 PostgreSQL 扩展，如将数据导出到 Amazon S3 概述中所述。
- 确定要将数据导出到 Amazon S3 的位置，如指定要导出到的 Amazon S3 文件路径中所述。
- 确保数据库实例具有 设置 Amazon S3 存储桶的访问权限 中所述的对 Amazon S3 的访问权限。

以下示例使用一个称作 sample_table 的数据库表。这些示例将数据导出到称为 *amzn-s3-demo-bucket* 的存储桶中。使用以下 SQL 语句在 psql 中创建示例表和数据。

```
psql=> CREATE TABLE sample_table (bid bigint PRIMARY KEY, name varchar(80));
psql=> INSERT INTO sample_table (bid,name) VALUES (1, 'Monday'), (2,'Tuesday'), (3,
 'Wednesday');
```

调用 aws_s3.query_export_to_s3

下面说明了调用 aws_s3.query_export_to_s3 函数的基本方法。

这些示例使用变量 s3_uri_1 确定包含用于标识 Amazon S3 文件的信息的结构。使用 aws_commons.create_s3_uri 函数可创建结构。

```
psql=> SELECT aws_commons.create_s3_uri(
   'amzn-s3-demo-bucket',
   'sample-filepath',
   'us-west-2'
) AS s3_uri_1 \gset
```

尽管以下两个 aws_s3.query_export_to_s3 函数调用的参数不同，但这些示例的结果是相同的。sample_table 表的所有行都将导出到称为 *amzn-s3-demo-bucket* 的存储桶中。

```
psql=> SELECT * FROM aws_s3.query_export_to_s3('SELECT * FROM
 sample_table', :'s3_uri_1');

psql=> SELECT * FROM aws_s3.query_export_to_s3('SELECT * FROM
 sample_table', :'s3_uri_1', options :='format text');
```

参数如下所述：

- 'SELECT * FROM sample_table' – 第一个参数是包含 SQL 查询的必需文本字符串。PostgreSQL 引擎将运行此查询。查询的结果将复制到其他参数中标识的 S3 存储桶。
- :'s3_uri_1' – 此参数是标识 Amazon S3 文件的结构。此示例使用变量来标识先前创建的结构。相反，您可通过内联方式在 aws_commons.create_s3_uri 函数调用中包含 aws_s3.query_export_to_s3 函数调用来创建结构，如下所示。

```
SELECT * from aws_s3.query_export_to_s3('select * from sample_table',
   aws_commons.create_s3_uri('amzn-s3-demo-bucket', 'sample-filepath', 'us-west-2')
);
```

- options :='format text' – options 参数是包含 PostgreSQL COPY 参数的可选文本字符串。复制过程使用 PostgreSQL COPY 命令的参数和格式。

如果指定的文件在 Amazon S3 存储桶中不存在，则会创建该文件。如果该文件已存在，则将覆盖该文件。以下是用于访问 Amazon S3 中的导出数据的语法。

```
s3-region://bucket-name[/path-prefix]/file-prefix
```

较大的导出将存储在多个文件中，每个文件的最大大小约为 6 GB。其他文件名具有相同的文件前缀，但追加了 _part*XX*。*XX* 表示 2，然后表示 3，依此类推。例如，假设您指定用于存储数据文件的路径，如下所示。

```
s3-us-west-2://amzn-s3-demo-bucket/my-prefix
```

如果导出必须创建三个数据文件，则 Amazon S3 存储桶将包含以下数据文件。

```
s3-us-west-2://amzn-s3-demo-bucket/my-prefix
s3-us-west-2://amzn-s3-demo-bucket/my-prefix_part2
s3-us-west-2://amzn-s3-demo-bucket/my-prefix_part3
```

有关此函数的完整参考以及其他调用方法，请参阅aws_s3.query_export_to_s3。有关访问 Amazon S3 中的文件的更多信息，请参阅 Amazon Simple Storage Service 用户指南中的查看对象。

导出到使用自定义分隔符的 CSV 文件

以下示例说明如何调用 aws_s3.query_export_to_s3 函数以将数据导出到使用自定义分隔符的文件。此示例使用 PostgreSQL COPY 命令的参数来指定逗号分隔值 (CSV) 格式和冒号 (:) 分隔符。

```
SELECT * from aws_s3.query_export_to_s3('select * from basic_test', :'s3_uri_1',
 options :='format csv, delimiter $$:$$');
```

导出到具有编码的二进制文件

以下示例说明如何调用 aws_s3.query_export_to_s3 函数以将数据导出到具有 Windows-1253 编码的二进制文件。

```
SELECT * from aws_s3.query_export_to_s3('select * from basic_test', :'s3_uri_1',
 options :='format binary, encoding WIN1253');
```

函数参考

函数

- aws_s3.query_export_to_s3
- aws_commons.create_s3_uri

aws_s3.query_export_to_s3

将 PostgreSQL 查询结果导出到 Amazon S3 存储桶。`aws_s3` 扩展提供 `aws_s3.query_export_to_s3` 函数。

两个必需的参数为 `query` 和 `s3_info`。这两个参数定义了要导出的查询，并标识了要导出到的 Amazon S3 存储桶。一个称作 `options` 的可选参数，用于定义各种导出参数。有关使用 `aws_s3.query_export_to_s3` 函数的示例，请参阅使用 aws_s3.query_export_to_s3 函数导出查询数据。

语法

```
aws_s3.query_export_to_s3(
    query text,
    s3_info aws_commons._s3_uri_1,
    options text,
    kms_key text
)
```

输入参数

query

一个必需的文本字符串，其中包含 PostgreSQL 引擎运行的 SQL 查询。此查询的结果将复制到 s3_info 参数中标识的 S3 存储桶。

s3_info

包含有关 S3 对象的以下信息的 aws_commons._s3_uri_1 复合类型：

- bucket – 要包含文件的 Amazon S3 存储桶的名称。
- file_path – Amazon S3 文件名和路径。
- region – 存储桶所在的Amazon区域。有关 Amazon 区域名称和关联值的列表，请参阅区域、可用区和 Local Zones。

  目前，此值必须是导出的数据库实例的同一个 Amazon 区域。默认值为导出的数据库实例的 Amazon 区域。

要创建 aws_commons._s3_uri_1 复合结构，请参阅 aws_commons.create_s3_uri 函数。

options

一个包含 PostgreSQL COPY 命令的参数的可选文本字符串。这些参数指定了在导出时复制数据的方式。有关更多详细信息，请参阅 PostgreSQL COPY 文档。

kms_key 文本

一个可选文本字符串，其中包含要将数据导出到的 S3 存储桶的客户托管 KMS 密钥。

备用输入参数

为帮助进行测试，您可以使用一组扩展的参数而非 s3_info 参数。下面是 aws_s3.query_export_to_s3 函数的其他语法变化。

不使用 s3_info 参数来标识 Amazon S3 文件，而使用 bucket、file_path 和 region 参数的组合。

```
aws_s3.query_export_to_s3(
    query text,
    bucket text,
    file_path text,
    region text,
```

```
    options text,
    kms_key text
)
```

query

一个必需的文本字符串，其中包含 PostgreSQL 引擎运行的 SQL 查询。此查询的结果将复制到 `s3_info` 参数中标识的 S3 存储桶。

bucket

一个包含 Amazon S3 存储桶（其中包含文件）的名称的必需文本字符串。

file_path

包含 Amazon S3 文件名（包含文件路径）的必填文本字符串。

region

一个包含存储桶所在的 Amazon 区域的可选文本字符串。有关 Amazon 区域名称和关联值的列表，请参阅[区域、可用区和 Local Zones](#)。

目前，此值必须是导出的数据库实例的同一个 Amazon 区域。默认值为导出的数据库实例的 Amazon 区域。

options

一个包含 PostgreSQL `COPY` 命令的参数的可选文本字符串。这些参数指定了在导出时复制数据的方式。有关更多详细信息，请参阅 [PostgreSQL COPY 文档](#)。

kms_key 文本

一个可选文本字符串，其中包含要将数据导出到的 S3 存储桶的客户托管 KMS 密钥。

输出参数

```
aws_s3.query_export_to_s3(
    OUT rows_uploaded bigint,
    OUT files_uploaded bigint,
    OUT bytes_uploaded bigint
)
```

rows_uploaded

针对给定查询成功上传到 Amazon S3 的表行的数目。

files_uploaded

已上传到 Amazon S3 的文件的数目。以约 6 GB 的大小创建文件。创建的每个附加文件的名称都追加了 `_partXX`。根据需要，*XX* 表示 2，然后表示 3，依此类推。

bytes_uploaded

已上传到 Amazon S3 的字节的总数。

示例

```
psql=> SELECT * from aws_s3.query_export_to_s3('select * from sample_table', 'amzn-s3-demo-bucket', 'sample-filepath');
psql=> SELECT * from aws_s3.query_export_to_s3('select * from sample_table', 'amzn-s3-demo-bucket', 'sample-filepath','us-west-2');
psql=> SELECT * from aws_s3.query_export_to_s3('select * from sample_table', 'amzn-s3-demo-bucket', 'sample-filepath','us-west-2','format text');
```

aws_commons.create_s3_uri

创建 `aws_commons._s3_uri_1` 结构来保存 Amazon S3 文件信息。在 `aws_commons.create_s3_uri` 函数的 `s3_info` 参数中使用 aws_s3.query_export_to_s3 函数的结果。有关使用 `aws_commons.create_s3_uri` 函数的示例，请参阅指定要导出到的 Amazon S3 文件路径。

语法

```
aws_commons.create_s3_uri(
   bucket text,
   file_path text,
   region text
)
```

输入参数

bucket

包含文件的 Amazon S3 存储桶名称的必需文本字符串。

file_path

包含 Amazon S3 文件名（包含文件路径）的必填文本字符串。

region

一个包含文件所在的 Amazon 区域的必需文本字符串。有关 Amazon 区域名称和关联值的列表，请参阅区域、可用区和 Local Zones。

Amazon S3 访问故障排除

如果在尝试将数据导出到 Amazon S3 时遇到连接问题，请首先确认与数据库实例关联的 VPC 安全组的出站访问规则是否允许网络连接。具体而言，安全组必须有一条规则允许数据库实例将 TCP 流量发送到端口 443 和任何 IPv4 地址（0.0.0.0/0）。有关更多信息，请参阅 通过创建安全组提供对 VPC 中的数据库实例的访问。

另请参阅以下建议：

- 对 Amazon RDS 身份和访问权限问题进行故障排除
- Amazon Simple Storage Service 用户指南中的排查 Amazon S3 的问题
- IAM 用户指南中的排查 Amazon S3 和 IAM 的问题

从 RDS for PostgreSQL 数据库实例中调用 Amazon Lambda 函数

Amazon Lambda 是事件驱动型计算服务，无需您预置或管理服务器即可运行代码。该服务可与许多 Amazon 服务搭配使用，其中包括 RDS for PostgreSQL。例如，您可以使用 Lambda 函数处理来自数据库的事件通知，或者在将新文件上传到 Amazon S3 时从文件中加载数据。若要了解 Lambda 的更多信息，请参阅《Amazon Lambda 开发人员指南》中的什么是 Amazon Lambda？

> **Note**
>
> 这些 RDS for PostgreSQL 版本支持调用 Amazon Lambda 函数：
>
> - 所有 PostgreSQL 17 版本
> - 所有 PostgreSQL 16 版本
> - 所有 PostgreSQL 15 版本
> - PostgreSQL 14.1 及更高的次要版本
> - PostgreSQL 13.2 及更高的次要版本
> - PostgreSQL 12.6 及更高的次要版本

设置 RDS for PostgreSQL 使用 Lambda 函数的过程包含多个步骤，其中涉及 Amazon Lambda、IAM、VPC 和 RDS for PostgreSQL 数据库实例。下文对必要步骤进行了总结。

有关 Lambda 函数的更多信息，请参阅《Amazon 开发人员指南https://docs.amazonaws.cn/lambda/latest/dg/getting-started.html》中的 Lambda 入门和 Amazon Lambda Lambda 函数。

主题

步骤 1：配置 RDS for PostgreSQL 数据库实例，实现与 Amazon Lambda 的出站连接

Lambda 函数始终在 Amazon Lambda 服务拥有的 Amazon VPC 中运行。Lambda 将向此 VPC 应用网络访问和安全规则，并且会自动维护和监控 VPC。RDS for PostgreSQL 数据库实例向 Lambda 服务的 VPC 发送网络流量。其配置方式取决于 数据库实例是公有实例，还是私有实例。

- 公有 RDS for PostgreSQL 数据库实例 – 如果数据库实例位于 VPC 的公有子网中，并且该实例的“PublicyAccessible”属性为 `true`，则该实例是公有的。若要查找此属性的值，您可以使用 describe-db-instances Amazon CLI 命令。您也可以使用 Amazon Web Services Management Console 打开 Connectivity & security（连接和安全性）选项卡，然后检查 Publicly accessible（公开访问）是否为 Yes（是）。要验证实例是否在您的 VPC 的公有子网中，您可以使用 Amazon Web Services Management Console或 Amazon CLI。

  要设置对 Lambda 的访问权限，您可以使用 Amazon Web Services Management Console 或 Amazon CLI 在 VPC 的安全组上创建出站规则。出站规则指定 TCP 可以使用端口 443 将数据包发送到任何 IPv4 地址（0.0.0.0/0）。
- 私有 RDS for PostgreSQL数据库实例 – 在这种情况下，实例的“PublicyAccessible”属性为 `false` 或它位于私有子网中。要允许实例使用 Lambda，您可以使用网络地址转换（NAT）网关。有关更多信息，请参阅 NAT 网关。或者，您可以使用用于 Lambda 的 VPC 端点配置 VPC。有关更多信息，请参阅《Amazon VPC 用户指南》中的 VPC 端点。该端点响应 RDS for PostgreSQL 数据库实例对 Lambda 函数发出的调用。VPC 端点使用自己的私有 DNS 解析。在您将 `rds.custom_dns_resolution` 的值从其默认值 0（未启用）更改为 1 之前，RDS for PostgreSQL 无法使用 Lambda VPC 端点。为此，请执行以下操作：
  - 创建自定义数据库参数组。
  - 将 `rds.custom_dns_resolution` 参数的值由默认值 `0` 更改为 `1`。
  - 修改数据库实例以使用自定义数据库参数组。
  - 重启实例，使修改的参数生效。

您的 VPC 现在可以在网络级别与 Amazon Lambda VPC 交互。接下来，您使用 IAM 配置权限。

步骤 2：为 RDS for PostgreSQL 数据库实例和 Amazon Lambda 配置 IAM

从 RDS for PostgreSQL 数据库实例调用 Lambda 函数需要特定权限。若要配置必要权限，建议创建允许调用 Lambda 函数的 IAM 策略，将该策略分配给一个角色，然后将该角色应用于数据库实例。这种

方法授予数据库实例代表您调用指定 Lambda 函数的权限。以下步骤说明如何使用 Amazon CLI 执行此操作。

配置 IAM 权限以将Amazon RDS 实例与 Lambda 搭配使用

1. 使用 create-policy Amazon CLI 命令创建允许 RDS for PostgreSQL 数据库实例调用指定 Lambda 函数的 IAM 策略。（语句 ID (Sid) 是策略语句的可选描述，对使用没有影响。）此策略授予 数据库实例调用指定 Lambda 函数所需的最低权限。

```
aws iam create-policy  --policy-name rds-lambda-policy --policy-document '{
    "Version": "2012-10-17",
    "Statement": [
        {
        "Sid": "AllowAccessToExampleFunction",
        "Effect": "Allow",
        "Action": "lambda:InvokeFunction",
        "Resource": "arn:aws:lambda:aws-region:444455556666:function:my-function"
        }
    ]
}'
```

   您也可以使用允许您调用任何 Lambda 函数的预定义 AWSLambdaRole 策略。有关更多信息，请参阅 Lambda 的基于身份的 IAM 策略

2. 使用 create-role Amazon CLI 命令创建该策略可在运行时担任的 IAM 角色。

```
aws iam create-role  --role-name rds-lambda-role --assume-role-policy-document '{
    "Version": "2012-10-17",
    "Statement": [
        {
        "Effect": "Allow",
        "Principal": {
            "Service": "rds.amazonaws.com"
        },
        "Action": "sts:AssumeRole"
        }
    ]
}'
```

3. 使用 attach-role-policy Amazon CLI 命令将策略应用于角色。

```
aws iam attach-role-policy \
```

```
    --policy-arn arn:aws:iam::444455556666:policy/rds-lambda-policy \
    --role-name rds-lambda-role --region aws-region
```

4. 使用 add-role-to-db-instance Amazon CLI 命令，将该角色应用于 RDS for PostgreSQL 数据库实例。最后一步允许数据库实例的数据库用户调用 Lambda 函数。

```
aws rds add-role-to-db-instance \
       --db-instance-identifier my-instance-name \
       --feature-name Lambda \
       --role-arn  arn:aws:iam::444455556666:role/rds-lambda-role   \
       --region aws-region
```

完成 VPC 和 IAM 配置后，便可以安装 aws_lambda 扩展。（请注意，您可以随时安装扩展，但只有在您设置正确的 VPC 支持和 IAM 权限之后，aws_lambda 扩展才会对 RDS for PostgreSQL 数据库实例的功能添加内容。）

步骤 3：为 RDS for PostgreSQL 数据库实例安装 **aws_lambda** 扩展

要将 Amazon Lambda 与 RDS for PostgreSQL 数据库实例结合使用，请将 aws_lambda PostgreSQL 扩展添加到 RDS for PostgreSQL 数据库实例。此扩展允许 RDS for PostgreSQL 数据库实例能从 PostgreSQL 调用 Lambda 函数。

为 RDS for PostgreSQL 数据库实例安装 **aws_lambda** 扩展

使用 PostgreSQL psql 命令行或 pgAdmin 工具连接到 RDS for PostgreSQL 数据库实例。

1. 以具有 rds_superuser 权限的用户身份，连接到 RDS for PostgreSQL 数据库实例。默认 postgres 用户如示例所示。

```
psql -h instance.444455556666.aws-region.rds.amazonaws.com -U postgres -p 5432
```

2. 安装 aws_lambda 扩展。aws_commons 扩展也是必要项。其为 PostgreSQL 的 aws_lambda 和许多其他 Aurora 扩展提供帮助程序函数。如果其尚未安装到 RDS for PostgreSQL 数据库实例上，则会按如下所示一并安装该扩展和 aws_lambda。

```
CREATE EXTENSION IF NOT EXISTS aws_lambda CASCADE;
NOTICE:  installing required extension "aws_commons"
CREATE EXTENSION
```

aws_lambda 扩展已安装在 数据库实例中。现在，您可以创建易于使用结构来调用 Lambda 函数。

步骤 4：将 Lambda 帮助程序函数与 RDS for PostgreSQL 数据库实例搭配使用（可选）

您可以使用 aws_commons 扩展中的帮助程序函数来准备实体，以便更轻松地从 PostgreSQL 调用这些实体。为此，您需要获得有关 Lambda 函数的以下信息：

- Function name（函数名称）：Lambda 函数的名称、Amazon Resource Name (ARN)、版本或别名。在 步骤 2：为实例和 Lambda 配置 IAM 中创建的 IAM 策略需要 ARN，所以建议您使用函数的 ARN。
- Amazon 区域 —（可选）如果与 RDS for PostgreSQL 数据库实例不在同一个区域中，则此区域为 Lambda 函数所在的 Amazon 区域。

您可以使用 aws_commons.create_lambda_function_arn 函数保存 Lambda 函数名称信息。此帮助程序函数创建了 aws_commons._lambda_function_arn_1 复合结构，其中包含调用函数所需的详细信息。在下文中，您可以找到三种替代方法来设置此复合结构。

```
SELECT aws_commons.create_lambda_function_arn(
   'my-function',
   'aws-region'
) AS aws_lambda_arn_1 \gset
```

```
SELECT aws_commons.create_lambda_function_arn(
   '111122223333:function:my-function',
   'aws-region'
) AS lambda_partial_arn_1 \gset
```

```
SELECT aws_commons.create_lambda_function_arn(
   'arn:aws:lambda:aws-region:111122223333:function:my-function'
) AS lambda_arn_1 \gset
```

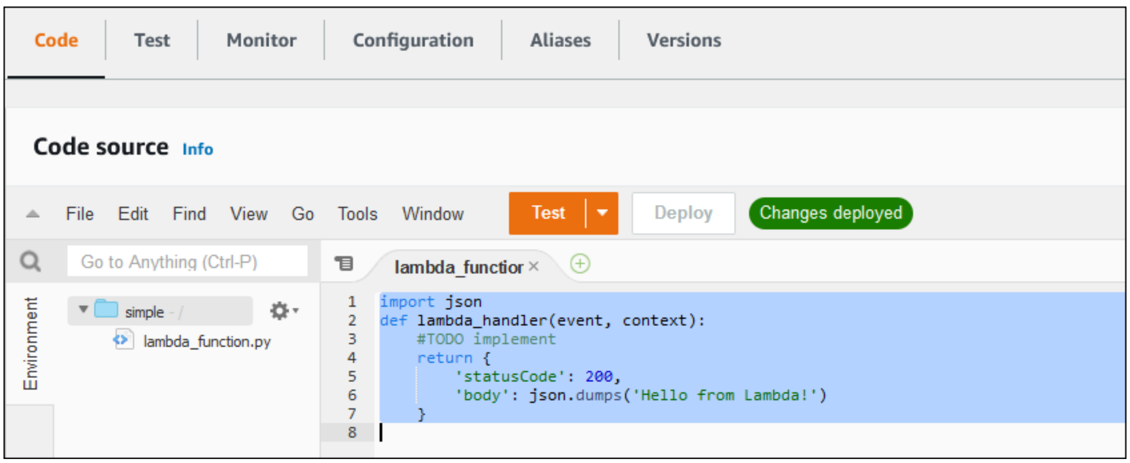
这些值均可用于 aws_lambda.invoke 函数调用。有关示例，请参阅 步骤 5：从 RDS for PostgreSQL 数据库实例调用 Lambda 函数。

步骤 5：从 RDS for PostgreSQL 数据库实例调用 Lambda 函数

`aws_lambda.invoke` 函数采用同步还是异步调用方式取决于 `invocation_type`。此参数的两个可用选项分别为 `RequestResponse`（默认）和 `Event`，如下所示。

- **`RequestResponse`**：此为同步调用类型。如果调用时未指定调用类型，默认此调用方式。响应有效负载包含 `aws_lambda.invoke` 函数的结果。如果工作流程需要接收 Lambda 函数的结果才能继续进行操作，请使用此调用类型。
- **`Event`**：此为异步调用类型。响应不包含包含结果的有效负载。如果工作流程不需要 Lambda 函数的结果即可继续进行操作，请使用此调用类型。

要对设置进行简单测试，您可以使用 `psql` 连接到数据库实例，然后从命令行调用示例函数。假设在 Lambda 服务上设置了一个基本函数，例如下面屏幕截图中所示的简单 Python 函数。



调用示例函数

1. 使用 `psql` 或 pgAdmin 连接到数据库实例。

   ```
   psql -h instance.444455556666.aws-region.rds.amazonaws.com -U postgres -p 5432
   ```

2. 使用函数的 ARN 调用该函数。

   ```
   SELECT * from
    aws_lambda.invoke(aws_commons.create_lambda_function_arn('arn:aws:lambda:aws-region:444455556666:function:simple', 'us-west-1'), '{"body": "Hello from
    Postgres!"}'::json );
   ```

   响应如下所示。

```
status_code |                            payload                            |
 executed_version | log_result
-------------+-------------------------------------------------------
+------------------+------------
         200 | {"statusCode": 200, "body": "\"Hello from Lambda!\""} | $LATEST
     |
(1 row)
```

如果调用尝试不成功，请参阅 Lambda 函数错误消息 。

步骤 6：授予其他用户调用 Lambda 函数的权限

在程序的这一步骤中，仅当您是 rds_superuser 才能调用 Lambda 函数。要允许其他用户调用您创建的任何函数，您需要向他们授予权限。

要授予调用 Lambda 函数的其他权限，请执行以下操作：

1. 使用 psql 或 pgAdmin 连接到数据库实例。

```
psql -h instance.444455556666.aws-region.rds.amazonaws.com -U postgres -p 5432
```

2. 运行以下 SQL 命令：

```
postgres=>  GRANT USAGE ON SCHEMA aws_lambda TO db_username;
GRANT EXECUTE ON ALL FUNCTIONS IN SCHEMA aws_lambda TO db_username;
```

示例：从 RDS for PostgreSQL 数据库实例调用 Lambda 函数

在下文中，您可以找到调用 aws_lambda.invoke 函数的一些示例。大多数示例都使用您在 步骤 4：将 Lambda 帮助程序函数与 RDS for PostgreSQL 数据库实例搭配使用（可选） 中创建的复合结构 aws_lambda_arn_1 来简化函数详细信息的传递。有关异步调用的示例，请参阅 示例：Lambda 函数的异步（事件）调用。列出的其他示例均使用同步调用。

要了解有关 Lambda 调用类型的更多信息，请参阅《Amazon Lambda 开发人员指南》中的调用 Lambda 函数。有关 aws_lambda_arn_1 的更多信息，请参阅 aws_commons.create_lambda_function_arn。

示例列表

- [示例：Lambda 函数的同步 (RequestResponse) 调用]
- [示例：Lambda 函数的异步（事件）调用]
- [示例：在函数响应中捕获 Lambda 执行日志]
- [示例：在 Lambda 函数中包含客户端上下文]
- [示例：调用 Lambda 函数的特定版本]

示例：Lambda 函数的同步 (RequestResponse) 调用

以下是 Lambda 函数同步调用的两个示例。这些 `aws_lambda.invoke` 函数调用的结果相同。

```
SELECT * FROM aws_lambda.invoke('aws_lambda_arn_1', '{"body": "Hello from 
 Postgres!"}'::json);
```

```
SELECT * FROM aws_lambda.invoke('aws_lambda_arn_1', '{"body": "Hello from 
 Postgres!"}'::json, 'RequestResponse');
```

参数如下所述：

- :`'aws_lambda_arn_1'`：此参数使用 `aws_commons.create_lambda_function_arn` 帮助程序函数标识在 [步骤 4：将 Lambda 帮助程序函数与 RDS for PostgreSQL 数据库实例搭配使用（可选）] 中创建的复合结构。您还可以通过内联方式在 `aws_lambda.invoke` 调用中创建此结构，如下所示。

  ```
  SELECT * FROM aws_lambda.invoke(aws_commons.create_lambda_function_arn('my-function',
   'aws-region'),
  '{"body": "Hello from Postgres!"}'::json
  );
  ```
- `'{"body": "Hello from PostgreSQL!"}'::json` – 要传递到 Lambda 函数的 JSON 负载。
- `'RequestResponse'` – Lambda 调用类型。

示例：Lambda 函数的异步（事件）调用

以下是异步 Lambda 函数调用的示例。`Event` 调用类型使用指定的输入负载计划 Lambda 函数调用并立即返回。在某些不依赖于 Lambda 函数结果的工作流程中使用 `Event` 调用类型。

```
SELECT * FROM aws_lambda.invoke('aws_lambda_arn_1', '{"body": "Hello from
 Postgres!"}'::json, 'Event');
```

示例：在函数响应中捕获 Lambda 执行日志

您可以使用 aws_lambda.invoke 函数调用中的 log_type 参数，在函数响应中包含执行日志的最后 4 kB。默认情况下，此参数设置为 None，但您可以指定 Tail 在响应中捕获 Lambda 执行日志的结果，如下所示。

```
SELECT *, select convert_from(decode(log_result, 'base64'), 'utf-8') as log FROM
 aws_lambda.invoke(:'aws_lambda_arn_1', '{"body": "Hello from Postgres!"}'::json,
 'RequestResponse', 'Tail');
```

将 aws_lambda.invoke 函数的 log_type 参数设置为 Tail，以在响应中包含执行日志。log_type 参数的默认值为 None。

返回的 log_result 是 base64 编码的字符串。您可以使用 decode 和 convert_from PostgreSQL 函数的组合来解码内容。

有关 log_type 的更多信息，请参阅 aws_lambda.invoke。

示例：在 Lambda 函数中包含客户端上下文

aws_lambda.invoke 函数具有 context 参数，可用于传递独立于有效负载的信息，如下所示。

```
SELECT *, convert_from(decode(log_result, 'base64'), 'utf-8') as log FROM
 aws_lambda.invoke(:'aws_lambda_arn_1', '{"body": "Hello from Postgres!"}'::json,
 'RequestResponse', 'Tail');
```

要包含客户端上下文，请将 JSON 对象用于 aws_lambda.invoke 函数的 context 参数。

有关 context 参数的更多信息，请参阅 aws_lambda.invoke 参考。

示例：调用 Lambda 函数的特定版本

通过在 aws_lambda.invoke 调用中包含 qualifier 参数，您可以指定 Lambda 函数的特定版本。在下文中，您可以找到一个使用 '*custom_version*' 作为版本别名完成此操作的示例。

```
SELECT * FROM aws_lambda.invoke('aws_lambda_arn_1', '{"body": "Hello from
 Postgres!"}'::json, 'RequestResponse', 'None', NULL, 'custom_version');
```

您还可以改为提供包含 Lambda 函数名称详细信息的 Lambda 函数限定符，如下所示。

```
SELECT * FROM aws_lambda.invoke(aws_commons.create_lambda_function_arn('my-
function:custom_version', 'us-west-2'),
'{"body": "Hello from Postgres!"}'::json);
```

有关 `qualifier` 和其他参数的详细信息，请参阅 aws_lambda.invoke 参考。

Lambda 函数错误消息

在下面的列表中，您可以找到有关错误消息的信息，以及可能的原因和解决方案。

- VPC 配置问题

  在尝试连接时，VPC 配置问题可能会引发以下错误消息：

  ```
  ERROR:  invoke API failed
  DETAIL: Amazon Lambda client returned 'Unable to connect to endpoint'.
  CONTEXT:  SQL function "invoke" statement 1
  ```

  导致此错误的常见原因是 VPC 安全组配置不当。确保在 VPC 安全组的端口 443 上打开 TCP 的出站规则，以便 VPC 能够连接到 Lambda VPC。

  如果数据库实例为私有实例，请检查 VPC 的私有 DNS 设置。确保将 `rds.custom_dns_resolution` 参数设置为 1 并按照 步骤 1：配置 RDS for PostgreSQL 数据库实例，实现与 Amazon Lambda 的出站连接 所述设置 Amazon PrivateLink。有关更多信息，请参阅接口 VPC 端点 (Amazon PrivateLink)。

- 缺乏调用 Lambda 函数所需的权限

  如果看到以下任一错误消息，说明调用此函数的用户（角色）没有适当的权限。

  ```
  ERROR:  permission denied for schema aws_lambda
  ```

  ```
  ERROR:  permission denied for function invoke
  ```

  必须向用户（角色）授予特定权限才能调用 Lambda 函数。有关更多信息，请参阅 步骤 6：授予其他用户调用 Lambda 函数的权限。

- 对 Lambda 函数中的错误处理不当

如果 Lambda 函数在请求处理过程中抛出异常，则 aws_lambda.invoke 会失败并显示 PostgreSQL 错误，如下所示。

```
SELECT * FROM aws_lambda.invoke('aws_lambda_arn_1', '{"body": "Hello from
 Postgres!"}'::json);
ERROR:  lambda invocation failed
DETAIL:  "arn:aws:lambda:us-west-2:555555555555:function:my-function" returned error
 "Unhandled", details: "<Error details string>".
```

务必处理 Lambda 函数或 PostgreSQL 应用程序中的错误。

Amazon Lambda 函数和参数参考

以下是通过 RDS for PostgreSQL 调用 Lambda 所用函数和参数的参考。

函数和参数

- aws_lambda.invoke
- aws_commons.create_lambda_function_arn
- aws_lambda 参数

aws_lambda.invoke

为 RDS for PostgreSQL 数据库实例 运行 Lambda 函数。

有关调用 Lambda 函数的更多详细信息，另请参阅 Amazon Lambda 开发人员指南中的调用。

语法

JSON

```
aws_lambda.invoke(
IN function_name TEXT,
IN payload JSON,
IN region TEXT DEFAULT NULL,
IN invocation_type TEXT DEFAULT 'RequestResponse',
IN log_type TEXT DEFAULT 'None',
IN context JSON DEFAULT NULL,
IN qualifier VARCHAR(128) DEFAULT NULL,
OUT status_code INT,
```

```
OUT payload JSON,
OUT executed_version TEXT,
OUT log_result TEXT)
```

```
aws_lambda.invoke(
IN function_name aws_commons._lambda_function_arn_1,
IN payload JSON,
IN invocation_type TEXT DEFAULT 'RequestResponse',
IN log_type TEXT DEFAULT 'None',
IN context JSON DEFAULT NULL,
IN qualifier VARCHAR(128) DEFAULT NULL,
OUT status_code INT,
OUT payload JSON,
OUT executed_version TEXT,
OUT log_result TEXT)
```

JSONB

```
aws_lambda.invoke(
IN function_name TEXT,
IN payload JSONB,
IN region TEXT DEFAULT NULL,
IN invocation_type TEXT DEFAULT 'RequestResponse',
IN log_type TEXT DEFAULT 'None',
IN context JSONB DEFAULT NULL,
IN qualifier VARCHAR(128) DEFAULT NULL,
OUT status_code INT,
OUT payload JSONB,
OUT executed_version TEXT,
OUT log_result TEXT)
```

```
aws_lambda.invoke(
IN function_name aws_commons._lambda_function_arn_1,
IN payload JSONB,
IN invocation_type TEXT DEFAULT 'RequestResponse',
IN log_type TEXT DEFAULT 'None',
IN context JSONB DEFAULT NULL,
IN qualifier VARCHAR(128) DEFAULT NULL,
OUT status_code INT,
OUT payload JSONB,
OUT executed_version TEXT,
OUT log_result TEXT
```

```
)
```

输入参数

function_name

Lambda 函数的标识名称。该值可以是函数名称、ARN 或部分 ARN。有关可能的格式的列表，请参阅 Amazon Lambda 开发人员指南中的 Lambda 函数名称格式。

payload

Lambda 函数的输入。格式可以是 JSON 或 JSONB。有关更多信息，请参阅 PostgreSQL 文档中的 JSON 类型。

区域

（可选）函数的 Lambda 区域。默认情况下，RDS 会从 `function_name` 的完整 ARN 中解析 Amazon 区域，或使用 RDS for PostgreSQL 数据库实例区域。如果此区域值与 `function_name` ARN 中提供的值冲突，则会引发错误。

invocation_type

Lambda 函数的调用类型。 值区分大小写。可能的值包括：

- `RequestResponse` – 默认值。Lambda 函数的这种调用类型是同步的，并在结果中返回响应负载。当工作流程依赖于立即接收 Lambda 函数结果时，请使用 `RequestResponse` 调用类型。
- `Event` – Lambda 函数的这种调用类型是异步的，并在没有返回负载的情况下立即返回。如果您不需要在工作流程开始之前了解 Lambda 函数的结果，请使用 `Event` 调用类型。
- `DryRun` – 这种类型的调用在不允许 Lambda 函数的情况下测试访问权限。

log_type

要在 `log_result` 输出参数中返回的 Lambda 日志类型。 值区分大小写。可能的值包括：

- Tail – 返回的 `log_result` 输出参数将包含执行日志的最后 4 kB。
- None – 未返回 Lambda 日志信息。

context

JSON 或 JSONB 格式的客户端上下文。要使用的字段包括 `custom` 和 `env`。

限定符

标识要调用的 Lambda 函数版本的限定符。如果此值与 `function_name` ARN 中提供的值冲突，则会引发错误。

输出参数

status_code

HTTP 状态响应代码。有关更多信息，请参阅 Amazon Lambda 开发人员指南中的 Lambda 调用响应元素。

payload

从运行的 Lambda 函数返回的信息。格式为 JSON 或 JSONB。

executed_version

运行的 Lambda 函数的版本。

log_result

如果 `log_type` 值在调用 Lambda 函数时为 `Tail`，则会返回执行日志信息。结果包含以 Base64 编码的执行日志的最后 4 kB。

aws_commons.create_lambda_function_arn

创建一个 `aws_commons._lambda_function_arn_1` 结构来保存 Lambda 函数名称信息。您可以在 aws_lambda.invoke `aws_commons.create_lambda_function_arn` 函数的 `function_name` 参数中使用 aws_lambda.invoke 函数的结果。

语法

```
aws_commons.create_lambda_function_arn(
    function_name TEXT,
    region TEXT DEFAULT NULL
    )
    RETURNS aws_commons._lambda_function_arn_1
```

输入参数

function_name

包含 Lambda 函数名称的必需文本字符串。该值可以是函数名称、部分 ARN 或完整 ARN。

区域

一个包含 Lambda 函数所在的Amazon区域的可选文本字符串。有关 区域名称和关联值的列表，请参阅区域、可用区和 Local Zones。

aws_lambda 参数

在该表中，您可以找到与 aws_lambda 函数关联的参数。

参数	描述
aws_lambda.connect_timeout_ms	这是一个动态参数，用于设置连接到 Amazon Lambda 时的最大等待时间。默认值为 1000。此参数允许的值为 1 - 900000。
aws_lambda.request_timeout_ms	这是一个动态参数，它设置了等待来自 Amazon Lambda 的响应时的最大等待时间。默认值为 3000。此参数允许的值为 1 - 900000。
aws_lambda.endpoint_override	指定可用于连接到 Amazon Lambda 的端点。空字符串会选择该区域的默认 Amazon Lambda 端点。必须重启数据库，此静态参数更改才能生效。

Amazon RDS for PostgreSQL 的数据库管理员常见任务

数据管理员 (DBA) 在管理 Amazon RDS for PostgreSQL 数据库实例时执行各种任务。如果您是一个已经熟悉 PostgreSQL 的 DBA，那么您需要注意在硬件上运行 PostgreSQL 和 RDS for PostgreSQL 之间的一些重要区别。例如，由于它是托管服务，Amazon RDS 不允许对数据库实例进行 Shell 访问。这意味着您无法直接访问 `pg_hba.conf` 和其他配置文件。对于 RDS for PostgreSQL，通常对本地实例的 PostgreSQL 配置文件所做的更改是针对与 RDS for PostgreSQL 数据库实例关联的自定义数据库参数组做出的。有关更多信息，请参阅 Amazon RDS 的参数组。

您也不能以与访问本地 PostgreSQL 实例相同的方式访问日志文件。要了解有关日志记录的更多信息，请参阅RDS for PostgreSQL 数据库日志文件。

再例如，您无法访问 PostgreSQL `superuser` 账户。在 RDS for PostgreSQL 上，`rds_superuser` 角色是权限最高的角色，它在设置时被授予 `postgres`。无论您是熟悉使用本地 PostgreSQL 还是对 RDS for PostgreSQL 完全陌生，我们都建议您了解 `rds_superuser` 角色，以及如何使用角色、用户、组和权限。有关更多信息，请参阅 了解 PostgreSQL 角色和权限。

以下是 RDS for PostgreSQL 的一些数据库管理员常见任务。

主题

RDS for PostgreSQL 中支持的排序规则

排序规则是一组规则，用于确定如何对存储在数据库中的字符串进行排序和比较。排序规则在计算机系统中起着重要作用，并作为操作系统的一部分包含在其中。当向语言中添加新字符或排序规则发生变化时，排序规则会随着时间的推移而变化。

排序规则库为排序规则定义特定的规则和算法。PostgreSQL 中最常用的排序规则库是 GNU C（Glibc）和 Unicode 国际化组件（ICU）。原定设置情况下，RDS for PostgreSQL 使用 Glibc 排序规则，其中包括适用于多字节字符序列的 Unicode 字符排序顺序。

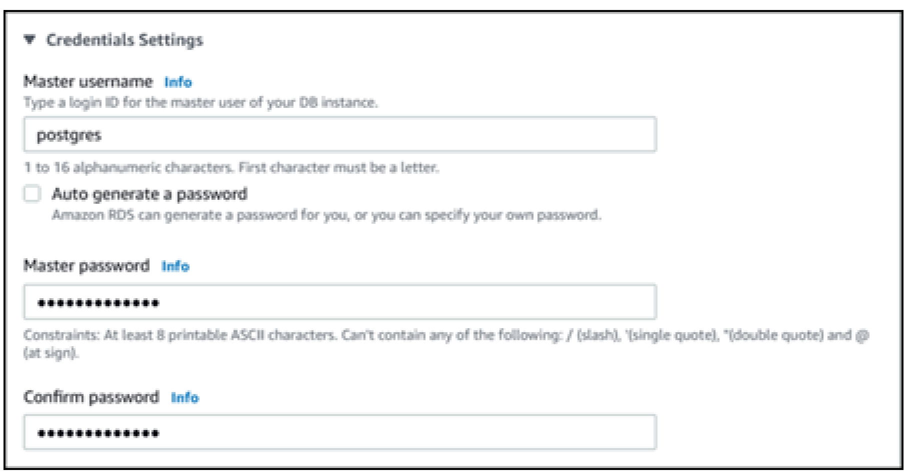
当您在 RDS for PostgreSQL 中创建新的数据库实例时，它将检查操作系统中是否有可用的排序规则。`CREATE DATABASE` 命令的 PostgreSQL 参数 `LC_COLLATE` 和 `LC_CTYPE` 用于指定排序规则，该排序规则是该数据库中的原定设置排序规则。或者，您也可以在 `CREATE DATABASE` 中使用 `LOCALE` 参数来设置这些参数。这决定了数据库中字符串的原定设置排序规则以及将字符分类为字母、数字或符号的规则。您也可以选择用于列、索引或查询的排序规则。

RDS for PostgreSQL 依赖于操作系统中的 Glibc 库来提供排序规则支持。RDS for PostgreSQL 实例定期使用最新版本的操作系统进行更新。这些更新有时包含较新版本的 Glibc 库。较新版本的 Glibc 很少会更改某些字符的排序顺序或排序规则，这可能会导致数据的排序方式不同或生成无效的索引条目。如果您在更新期间发现用于排序规则的排序顺序有问题，则可能需要重建索引。

为了减少 Glibc 更新可能产生的影响，RDS for PostgreSQL 现在包含一个独立的原定设置排序规则库。这个排序规则库在 RDS for PostgreSQL 14.6、13.9、12.13、11.18、10.23 和更高的次要版本中可用。它与 Glibc 2.26-59.amzn2 兼容，并提供排序顺序稳定性以防止错误的查询结果。

了解 PostgreSQL 角色和权限

在使用 Amazon Web Services Management Console 创建 RDS for PostgreSQL 数据库实例时，将同时创建管理员账户。默认情况下，其名称为 `postgres`，如以下屏幕截图所示：



您可以选择其他名称，而不是接受默认值（`postgres`)。如果这样做，您选择的名称必须以字母开头，并且必须介于 1 到 16 个字母数字字符之间。为简单起见，在整个指南中，我们将使用默认值 (`postgres`) 来指代此主用户账户。

如果您使用 `create-db-instance` Amazon CLI 而不是 Amazon Web Services Management Console，则可以通过将名称传递给命令中的 `master-username` 参数来创建名称。有关更多信息，请参阅 创建 Amazon RDS 数据库实例。

无论您是使用 Amazon Web Services Management Console、Amazon CLI 还是 Amazon RDS API，也无论您是使用默认 `postgres` 名称还是选择其他名称，这第一个数据库用户账户都是 `rds_superuser` 组的成员并具有 `rds_superuser` 权限。

主题

- 了解 rds_superuser 角色
- 控制用户对 PostgreSQL 数据库的访问
- 委托和控制用户密码管理
- 使用 SCRAM 进行 PostgreSQL 密码加密

了解 rds_superuser 角色

在 PostgreSQL 中，角色可以针对数据库中的各种对象定义一个用户、一个组或一组授予组或用户的特定权限。PostgreSQL 命令 `CREATE USER` 和 `CREATE GROUP` 已替换为更通用的 `CREATE ROLE`，并使用特定属性来区分数据库用户。数据库用户可以被视为具有 LOGIN 权限的角色。

> **Note**
>
> 仍然可以使用 `CREATE USER` 和 `CREATE GROUP` 命令。有关更多信息，请参阅 PostgreSQL 文档中的数据库角色。

`postgres` 用户是您的 RDS for PostgreSQL 数据库实例上权限最高的数据库用户。它具有以下 `CREATE ROLE` 语句所定义的特征。

```
CREATE ROLE postgres WITH LOGIN NOSUPERUSER INHERIT CREATEDB CREATEROLE NOREPLICATION
 VALID UNTIL 'infinity'
```

属性 `NOSUPERUSER`、`NOREPLICATION`、`INHERIT` 和 `VALID UNTIL 'infinity'` 是 CREATE ROLE 的默认选项，除非另有说明。

默认情况下，`postgres` 拥有授予 `rds_superuser` 角色的权限以及创建角色和数据库的权限。`rds_superuser` 角色允许 `postgres` 用户执行以下操作：

- 添加可用于 Amazon RDS 的扩展。有关更多信息，请参阅 使用 Amazon RDS for PostgreSQL 支持的 PostgreSQL 功能
- 为用户创建角色并向用户授予权限。有关更多信息，请参阅 PostgreSQL 文档中的 CREATE ROLE 和 GRANT。
- 创建数据库。有关更多信息，请参阅 PostgreSQL 文档中的 CREATE DATABASE。
- 将 `rds_superuser` 权限授予没有这些权限的用户角色，并根据需要撤销权限。建议您仅向执行超级用户任务的那些用户授予此角色。换句话说，您可以将此角色授予数据库管理员 (DBA) 或系统管理员。
- 向没有 `rds_superuser` 角色的数据库用户授予（和撤销）`rds_replication` 角色。
- 向没有 `rds_superuser` 角色的数据库用户授予（和撤销）`rds_password` 角色。
- 通过使用 `pg_stat_activity` 视图获取有关所有数据库连接的状态信息。需要时，`rds_superuser` 可以通过使用 `pg_terminate_backend` 或 `pg_cancel_backend` 停止任何连接。

在 `CREATE ROLE postgres...` 语句中，您可以看到 `postgres` 用户角色明确禁止 PostgreSQL `superuser` 权限。RDS for PostgreSQL 是一项托管服务，因此您无法访问主机操作系统，也无法使用 PostgreSQL `superuser` 账户进行连接。许多需要独立 PostgreSQL 上的 `superuser` 访问权限的任务都由 Amazon RDS 自动管理。

有关授权权限的更多信息，请参阅 PostgreSQL 文档中的 GRANT。

`rds_superuser` 角色是 中的几个预定义角色之一。RDS for PostgreSQL 数据库实例。

> **Note**
>
> 在 PostgreSQL 13 和更早版本中，预定义角色称为默认角色。

在下面的列表中，您可以找到为新 自动创建的一些其他预定义角色。RDS for PostgreSQL 数据库实例。无法更改预定义角色及其权限。无法删除、重命名或修改这些预定义角色的权限。此类尝试会导致错误。

- rds_password – 可以为数据库用户更改密码和设置密码限制的角色。默认情况下，`rds_superuser` 角色被授予此角色，并且可以将此角色授予数据库用户。有关更多信息，请参阅 控制用户对 PostgreSQL 数据库的访问。
  - 对于早于 14 的 RDS for PostgreSQL 版本，`rds_password` 角色可以为数据库用户和具有 `rds_superuser` 角色的用户更改密码和设置密码限制。在 RDS for PostgreSQL 14 及

更高版本中，`rds_password` 角色只能为数据库用户更改密码和设置密码限制。只有具有 `rds_superuser` 角色的用户才能对具有 `rds_superuser` 角色的其他用户执行这些操作。

- rdsadmin – 具有 `superuser` 权限的管理员将对独立的 PostgreSQL 数据库执行许多管理任务，此角色专为处理这些管理任务而创建。此角色由 RDS for PostgreSQL 在内部用于许多管理任务。
- rdstopmgr – Amazon RDS 在内部用于支持多可用区部署的角色。
- rds_reserved – Amazon RDS 在内部用于保留数据库连接的角色。

查看角色及其权限

根据您的 PostgreSQL 版本，您可以使用不同的命令在 RDS for PostgreSQL 数据库实例中查看预定义角色及其权限。

对于 PostgreSQL 15 及更早版本

连接到 RDS for PostgreSQL 数据库实例，并在 psql 中使用 \du 命令：

```
postgres=> \du
                                                               List of roles
    Role name    |                         Attributes                         |
                  Member of
-----------------+------------------------------------------------------------
+------------------------------------------------------
 postgres        | Create role, Create DB                                    +|
 {rds_superuser}
                 | Password valid until infinity                              |
 rds_ad          | Cannot login                                               | {}
 rds_iam         | Cannot login                                               | {}
 rds_password    | Cannot login                                               | {}
 rds_replication | Cannot login                                               | {}
 rds_superuser   | Cannot login                                               |
 {pg_monitor,pg_signal_backend,rds_password,rds_replication}
 rdsadmin        | Superuser, Create role, Create DB, Replication, Bypass RLS+| {}
                 | Password valid until infinity                              |
```

对于 PostgreSQL 16 及更高版本

从 PostgreSQL 16 开始，可以使用 `\drg+` 命令来查看详细的角色成员资格信息：

```
postgres=> \drg+
                             List of role grants
   Role name    |           Member of            |       Options       | Grantor
```

```
---------------+-----------------------------+----------------------+----------
 postgres      | rds_superuser               | INHERIT, SET         | rdsadmin
 rds_superuser | pg_checkpoint               | ADMIN, INHERIT, SET | rdsadmin
 rds_superuser | pg_monitor                  | ADMIN, INHERIT, SET | rdsadmin
 rds_superuser | pg_signal_backend           | ADMIN, INHERIT, SET | rdsadmin
 rds_superuser | pg_use_reserved_connections | ADMIN, INHERIT, SET | rdsadmin
 rds_superuser | rds_password                | ADMIN, INHERIT, SET | rdsadmin
 rds_superuser | rds_replication             | ADMIN, INHERIT, SET | rdsadmin
```

要检查角色成员资格，无论 PostgreSQL 版本如何，都可以使用以下 SQL 查询：

```
SELECT m.rolname AS "Role name", r.rolname AS "Member of"
FROM pg_catalog.pg_roles m
JOIN pg_catalog.pg_auth_members pam ON (pam.member = m.oid)
LEFT JOIN pg_catalog.pg_roles r ON (pam.roleid = r.oid)
LEFT JOIN pg_catalog.pg_roles g ON (pam.grantor = g.oid)
WHERE m.rolname !~ '^pg_'
ORDER BY 1, 2;
```

在输出中，您可以看到 `rds_superuser` 不是数据库用户角色（无法登录），但它具有许多其他角色的权限。您还可以看到数据库用户 `postgres` 是 `rds_superuser` 角色的成员。如前所述，`postgres` 是 Amazon RDS 控制台的 Create database（创建数据库）页面中的默认值。如果您选择了另一个名称，则该名称将显示在角色列表中。

控制用户对 PostgreSQL 数据库的访问

PostgreSQL 中的新数据库始终使用数据库 `public` 架构中的一组默认权限创建，允许所有数据库用户和角色创建对象。例如，这些权限使数据库用户能够连接数据库，并在连接后创建临时表格。

为了更好地控制用户对您在 RDS for PostgreSQL 数据库实例上创建的数据库实例的访问，我们建议您撤消这些默认 `public` 权限。撤消后，您可以更精确地为数据库用户授权，如以下过程中所示。

为新数据库实例设置角色和权限

假设您正在新创建的 RDS for PostgreSQL 数据库实例上设置数据库，以供几位研究人员使用，他们都需要对数据库的读写访问权限。

1. 使用 psql（或 pgAdmin）连接到 RDS for PostgreSQL 数据库实例：

   ```
   psql --host=your-db-instance.666666666666.aws-region.rds.amazonaws.com --port=5432
    --username=postgres --password
   ```

出现提示时请输入密码。psql 客户端会建立连接并显示默认的管理连接数据库 postgres=>，作为提示符。

2. 要阻止数据库用户在 public 架构中创建对象，执行以下操作：

```
postgres=> REVOKE CREATE ON SCHEMA public FROM PUBLIC;
REVOKE
```

3. 接下来，创建一个新数据库实例：

```
postgres=> CREATE DATABASE lab_db;
CREATE DATABASE
```

4. 在这个新数据库上，撤消 PUBLIC 架构的所有权限。

```
postgres=> REVOKE ALL ON DATABASE lab_db FROM public;
REVOKE
```

5. 为数据库用户创建角色。

```
postgres=> CREATE ROLE lab_tech;
CREATE ROLE
```

6. 为具有此角色的数据库用户提供连接到数据库的能力。

```
postgres=> GRANT CONNECT ON DATABASE lab_db TO lab_tech;
GRANT
```

7. 向具有 lab_tech 角色的所有用户授予对此数据库的所有权限。

```
postgres=> GRANT ALL PRIVILEGES ON DATABASE lab_db TO lab_tech;
GRANT
```

8. 创建数据库用户，如下所示：

```
postgres=> CREATE ROLE lab_user1 LOGIN PASSWORD 'change_me';
CREATE ROLE
postgres=> CREATE ROLE lab_user2 LOGIN PASSWORD 'change_me';
CREATE ROLE
```

9. 向这两个用户授予与 lab_tech 角色关联的权限：

```
postgres=> GRANT lab_tech TO lab_user1;
GRANT ROLE
postgres=> GRANT lab_tech TO lab_user2;
GRANT ROLE
```

此时，lab_user1 和 lab_user2 可以连接到 lab_db 数据库。此示例未遵循企业使用的最佳实践，其中可能包括创建多个数据库实例、不同的架构和授予有限权限。有关更多完整信息和其他方案，请参阅管理 PostgreSQL 用户和角色。

有关 PostgreSQL 数据库中特权的更多信息，请参阅 PostgreSQL 文档中的 GRANT 命令。

委托和控制用户密码管理

作为 DBA，您可能需要委托用户密码的管理。或者，您可能希望防止数据库用户更改其密码或重新配置密码限制，例如密码生命周期。要确保只有您选择的数据库用户才能更改密码设置，可以启用受限密码管理特征。激活此特征时，只有那些已被授予 rds_password 角色的数据库用户可以管理密码。

> **Note**
>
> 要使用受限密码管理，您的 RDS for PostgreSQL 数据库实例必须运行 PostgreSQL 10.6 或更高版本。

默认情况下，此特征为 off，如下所示：

```
postgres=> SHOW rds.restrict_password_commands;
  rds.restrict_password_commands
--------------------------------
 off
(1 row)
```

要启用此特征，请使用自定义参数组并将 rds.restrict_password_commands 的设置更改为 1。一定要重新启动 RDS for PostgreSQL 数据库实例，此设置才能生效。

激活此特征后，以下 SQL 命令需要 rds_password 权限：

```
CREATE ROLE myrole WITH PASSWORD 'mypassword';
CREATE ROLE myrole WITH PASSWORD 'mypassword' VALID UNTIL '2023-01-01';
```

```
ALTER ROLE myrole WITH PASSWORD 'mypassword' VALID UNTIL '2023-01-01';
ALTER ROLE myrole WITH PASSWORD 'mypassword';
ALTER ROLE myrole VALID UNTIL '2023-01-01';
ALTER ROLE myrole RENAME TO myrole2;
```

如果密码使用 MD5 哈希算法，重命名角色 (`ALTER ROLE myrole RENAME TO newname`) 也会受到限制。

激活此特征后，在没有 `rds_password` 角色权限的情况下尝试这些 SQL 命令中的任何一个都会生成以下错误：

```
ERROR: must be a member of rds_password to alter passwords
```

我们建议您仅将 `rds_password` 授予少数几个仅用于密码管理的角色。如果您将 `rds_password` 权限授予没有 `rds_superuser` 权限的数据库用户，则还需要授他们 `CREATEROLE` 属性。

请确保您验证了密码要求，例如客户端上的过期时间以及所需的复杂性。如果您使用自己的客户端实用程序进行与密码相关的更改，则该实用程序需要是 `rds_password` 的成员并具有 `CREATE ROLE` 权限。

使用 SCRAM 进行 PostgreSQL 密码加密

在对密码进行加密时，加盐质询响应身份验证机制（SCRAM）是 PostgreSQL 的默认消息摘要（MD5）算法的替代方案。–般认为 SCRAM 身份验证机制比 MD5 更安全。要了解有关这两种不同的密码保护方法的更多信息，请参阅 PostgreSQL 文档中的密码身份验证。

我们建议您使用 SCRAM 而不是 MD5 作为您的 RDS for PostgreSQL 数据库实例的密码加密方案。这是一种加密质询-响应机制，它使用 scram-sha-256 算法进行密码身份验证和加密。

要支持 SCRAM，您可能需要更新客户端应用程序的库。例如，42.2.0 之前的 JDBC 版本不支持 SCRAM。有关更多信息，请参阅 PostgreSQL JDBC 驱动程序文档中的 PostgreSQL JDBC 驱动程序。有关其他 PostgreSQL 驱动程序和 SCRAM 支持的列表，请参阅 PostgreSQL 文档中的驱动程序列表。

RDS for PostgreSQL 版本 13.1 和更高版本支持 scram-sha-256。这些版本还可让您将数据库实例配置为需要 SCRAM，如以下过程所述。

设置 RDS for PostgreSQL 数据库实例以要求使用 SRAM

您可以要求 RDS for PostgreSQL 数据库实例仅接受使用 scram-sha-256 算法的密码。

> **Important**
>
> 对于带有 PostgreSQL 数据库的现有 RDS 代理，如果您将数据库身份验证修改为仅使用 SCRAM，代理将在长达 60 秒的时间内不可用。要避免此问题，请执行以下操作之一：
>
> - 确保数据库同时允许 SCRAM 和 MD5 身份验证。
> - 要仅使用 SCRAM 身份验证，请创建一个新代理，将应用程序流量迁移到新代理，然后删除先前与数据库关联的代理。

在对系统进行更改之前，请务必了解完整的过程，如下所示：

- 获取有关所有数据库用户的所有角色和密码加密的信息。
- 仔细检查 RDS for PostgreSQL 数据库实例的参数设置，以了解用于控制密码加密的参数。
- 如果您的 RDS for PostgreSQL 数据库实例使用默认参数组，您需要创建自定义数据库参数组，然后将其应用到您的 RDS for PostgreSQL 数据库实例，以便您可以在需要时修改参数。如果您的 RDS for PostgreSQL 数据库实例使用自定义参数组，您可以稍后根据需要在此过程中修改必要的参数。
- 将 `password_encryption` 参数更改为 `scram-sha-256`。
- 通知所有数据库用户他们需要更新密码。对您的 `postgres` 账户执行相同的操作。使用 scram-sha-256 算法对新密码进行加密和存储。
- 验证是否使用加密类型对所有密码加密。
- 如果所有密码都使用 scram-sha-256，您可以将 `rds.accepted_password_auth_method` 参数从 `md5+scram` 更改为 `scram-sha-256`。

> **Warning**
>
> 仅将 `rds.accepted_password_auth_method` 更改为 scram-sha-256 之后，使用 md5 加密的密码的任何用户（角色）都将无法连接。

做好准备，以要求 RDS for PostgreSQL 数据库实例使用 SCRAM

在对 RDS for PostgreSQL 数据库实例进行任何更改之前，检查所有现有的数据库用户账户。另外，请检查用于密码的加密类型。您可以使用 `rds_tools` 扩展来执行这些任务。要了解哪些 PostgreSQL 版本支持 `rds_tools`，请参阅 Extension versions for Amazon RDS for PostgreSQL。

获取数据库用户（角色）和密码加密方法的列表

1. 使用 psql 连接到 RDS for PostgreSQL 数据库实例，如下所示。

```
psql --host=db-name.111122223333.aws-region.rds.amazonaws.com --port=5432 --
username=postgres --password
```

2. 安装 rds_tools 扩展。

```
postgres=> CREATE EXTENSION rds_tools;
CREATE EXTENSION
```

3. 获取角色和加密的列表。

```
postgres=> SELECT * FROM 
      rds_tools.role_password_encryption_type();
```

您将看到类似以下内容的输出。

```
       rolname        | encryption_type
----------------------+-----------------
 pg_monitor           |
 pg_read_all_settings |
 pg_read_all_stats    |
 pg_stat_scan_tables  |
 pg_signal_backend    |
 lab_tester           | md5
 user_465             | md5
 postgres             | md5
(8 rows)
```

创建自定义数据库参数组

> **Note**
> 如果您的 RDS for PostgreSQL 数据库实例已使用自定义参数组，您不需要创建新参数组。

有关 Amazon RDS 的参数组的概述，请参阅在 RDS for PostgreSQL 数据库实例上使用参数。

用于密码的密码加密类型在一个参数（即 password_encryption）中设置。 RDS for PostgreSQL 数据库实例允许的加密在另一个参数 rds.accepted_password_auth_method 中设置。更改其中任何一个的默认值都要求您创建自定义数据库参数组，然后将其应用到您的实例。

也可以使用 Amazon Web Services Management Console 或 RDS API 创建自定义数据库参数组。有关更多信息，请参阅。

现在可以将自定义参数组与数据库实例关联。

创建自定义数据库参数组

1. 使用 CLI 命令 create-db-parameter-group 创建自定义数据库参数组。此示例使用 postgres13 作为此自定义参数组的来源。

   对于 Linux、macOS 或 Unix：

   ```
   aws rds create-db-parameter-group --db-parameter-group-name 'docs-lab-scram-passwords' \
     --db-parameter-group-family postgres13  --description 'Custom parameter group for SCRAM'
   ```

   对于 Windows：

   ```
   aws rds create-db-parameter-group --db-parameter-group-name "docs-lab-scram-passwords" ^
     --db-parameter-group-family postgres13  --description "Custom DB parameter group for SCRAM"
   ```

2. 使用 CLI 命令 modify-db-instance 将此自定义参数组应用于 RDS for PostgreSQL 数据库集群。

   对于 Linux、macOS 或 Unix：

   ```
   aws rds modify-db-instance --db-instance-identifier 'your-instance-name' \
           --db-parameter-group-name "docs-lab-scram-passwords
   ```

   对于 Windows：

   ```
   aws rds modify-db-instance --db-instance-identifier "your-instance-name" ^
           --db-parameter-group-name "docs-lab-scram-passwords
   ```

要将 RDS for PostgreSQL 数据库实例与您的自定义数据库参数组重新同步，您需要重启集群的主实例和所有其他实例。为了尽量减少对用户的影响，请安排在常规维护时段内执行此操作。

配置密码加密以使用 SCRAM

RDS for PostgreSQL 数据库实例使用的密码加密机制在数据库参数组的 `password_encryption` 参数中设置。允许的值为未设置、`md5` 或 `scram-sha-256`。默认值取决于 RDS for PostgreSQL 版本，如下所示：

- RDS for PostgreSQL 14 和更高版本 – 默认为 `scram-sha-256`
- RDS for PostgreSQL 13 – 默认为 `md5`

使用附加到 RDS for PostgreSQL 数据库实例的自定义数据库参数组，您可以修改密码加密参数的值。

Name	Values	Allowed values	Modifiable	Source	Apply type
password_encryption	md5	md5, scram-sha-256	true	system	dynamic
rds.accepted_password_auth_method	md5+scram	md5+scram, scram	true	system	dynamic

将密码加密设置更改为 scram-sha-256

- 将密码加密的值更改为 scram-sha-256，如下所示。可以立即应用更改，因为参数是动态的，这样，无需重新启动即可使更改生效。

  对于 Linux、macOS 或 Unix：

  ```
  aws rds modify-db-parameter-group --db-parameter-group-name \
    'docs-lab-scram-passwords' --parameters
   'ParameterName=password_encryption,ParameterValue=scram-
  sha-256,ApplyMethod=immediate'
  ```

  对于 Windows：

  ```
  aws rds modify-db-parameter-group --db-parameter-group-name ^
  ```

```
  "docs-lab-scram-passwords" --parameters
 "ParameterName=password_encryption,ParameterValue=scram-
sha-256,ApplyMethod=immediate"
```

将用户角色的密码迁移到 SCRAM

您可以将用户角色的密码迁移到 SCRAM，如下所述。

将数据库用户（角色）密码从 MD5 迁移到 SCRAM

1. 以管理员用户身份（默认用户名 `postgres`）登录，如下所示。

```
psql --host=db-name.111122223333.aws-region.rds.amazonaws.com --port=5432 --
username=postgres --password
```

2. 通过使用以下命令，在 RDS for PostgreSQL 数据库实例上检查 `password_encryption` 参数的设置。

```
postgres=> SHOW password_encryption;
 password_encryption
---------------------
 md5
 (1 row)
```

3. 将此参数的值更改为 scram-sha-256。这是一个动态参数，因此您不需要在进行此更改后重新启动该实例。再次检查该值，以确保它现在已设置为 `scram-sha-256`，如下所示。

```
postgres=> SHOW password_encryption;
 password_encryption
---------------------
 scram-sha-256
 (1 row)
```

4. 通知所有数据库用户更改其密码。请确保还要更改您自己的用于 `postgres` 账户的密码（具有 `rds_superuser` 权限的数据库用户）。

```
labdb=> ALTER ROLE postgres WITH LOGIN PASSWORD 'change_me';
ALTER ROLE
```

5. 对于您的 RDS for PostgreSQL 数据库实例上的所有数据库重复此过程。

更改参数以要求使用 SCRAM

这是该过程的最后一步。在以下过程中进行更改后，任何仍对密码使用 md5 加密的用户账户（角色）都将无法登录到 RDS for PostgreSQL 数据库实例。

`rds.accepted_password_auth_method` 指定 RDS for PostgreSQL 数据库实例在登录过程中接受的用户密码加密方法。默认值为 `md5+scram`，这意味着可以接受任一种方法。在下面的图中，您可以找到此参数的默认设置。

	Name	Values	Allowed values	Modifiable	Source	Apply type
	password_encryption	scram-sha-256	md5, scram-sha-256	true	system	dynamic
	rds.accepted_password_auth_method	md5+scram	md5+scram, scram	true	system	dynamic

此参数允许的值仅为 `md5+scram` 或 `scram`。如果将此参数值更改为 `scram`，则必须使用这种方法。

更改参数值以要求对密码进行 SCRAM 身份验证

1. 验证 RDS for PostgreSQL 数据库实例上所有数据库的所有数据库用户密码是否使用 `scram-sha-256` 进行密码加密。为此，查询 `rds_tools` 以获得角色（用户）和加密类型，如下所示。

```
postgres=> SELECT * FROM rds_tools.role_password_encryption_type();
  rolname        | encryption_type
  ----------------------+-----------------
  pg_monitor            |
  pg_read_all_settings  |
  pg_read_all_stats     |
  pg_stat_scan_tables   |
  pg_signal_backend     |
  lab_tester            | scram-sha-256
  user_465              | scram-sha-256
  postgres              | scram-sha-256
  ( rows)
```

2. 在您的 RDS for PostgreSQL 数据库实例中的所有数据库实例中重复此查询。

   如果所有密码都使用 scram-sha-256，您可以继续操作。

3. 将接受的密码身份验证的值更改为 scram-sha-256，如下所示。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-parameter-group --db-parameter-group-name 'docs-lab-scram-passwords' \
  --parameters
 'ParameterName=rds.accepted_password_auth_method,ParameterValue=scram,ApplyMethod=immediat
```

对于 Windows：

```
aws rds modify-db-parameter-group --db-parameter-group-name "docs-lab-scram-passwords" ^
  --parameters
 "ParameterName=rds.accepted_password_auth_method,ParameterValue=scram,ApplyMethod=immediat
```

在 Amazon RDS for PostgreSQL 上使用 PostgreSQL autovacuum

我们强烈建议您使用 Autovacuum 功能来保持 PostgreSQL 数据库实例正常运行。Autovacuum 自动启动 VACUUM 和 ANALYZE 命令。它会检查包含大量插入的、更新的或删除的元组的表。进行此检查后，它会通过从 PostgreSQL 数据库中删除过时的数据或元组来回收存储。

在使用任何默认 PostgreSQL 数据库参数组创建的 RDS for PostgreSQL 数据库实例上，会默认启用 autovacuum。默认情况下，还会设置与 Autovacuum 功能关联的其他配置参数。这些默认值是通用值，因此可以针对特定工作负载优化与 Autovacuum 功能关联的某些参数。

在下文中，您可以了解有关 autovacuum 功能以及如何为 RDS for PostgreSQL 数据库实例优化其部分参数的更多信息。有关高级信息，请参阅使用 PostgreSQL 的最佳实践。

主题

- [使用大型索引管理 autovacuum]
- [其他影响 Autovacuum 的参数]
- [设置表级别 Autovacuum 参数]
- [记录 autovacuum 和 vacuum 活动]
- [了解对无效数据库使用 autovacuum 的行为]
- [识别并解决 RDS for PostgreSQL 中的积极真空拦截器]

为 Autovacuum 分配内存

影响Autovacuum 性能的最重要参数之一是 `autovacuum_work_mem` 参数。在 RDS for PostgreSQL 版本 14 及更早版本中，`autovacuum_work_mem` 参数设置为 -1，表示改用 `maintenance_work_mem` 的设置。对于所有其它版本，`autovacuum_work_mem` 由 GREATEST({DBInstanceClassMemory/32768}, 65536) 决定。

手动 vacuum 操作始终使用 `maintenance_work_mem` 设置，默认设置为 GREATEST({DBInstanceClassMemory/63963136*1024}, 65536)，也可以使用 `SET` 命令在会话级别对其进行调整，以进行更有针对性的手动 `VACUUM` 操作。

`autovacuum_work_mem` 确定 autovacuum 在对索引执行 vacuum 操作时，用于容纳无效元组 (`pg_stat_all_tables.n_dead_tup`) 的标识符的内存。

当执行计算以确定 `autovacuum_work_mem` 参数的值时，请注意以下事项：

- 如果将此参数设得过低，则 vacuum 过程可能必须扫描表多次才能完成其工作。此类多次扫描可能会对性能产生负面影响。对于较大的实例，将 `maintenance_work_mem` 或 `autovacuum_work_mem` 设置为至少 1 GB，可以提高对包含大量无效元组的表执行 vacuum 操作的性能。但是，在 PostgreSQL 版本 16 及更早版本中，vacuum 的内存使用量上限为 1 GB，这足以单次处理大约 1.79 亿个无效元组。如果表的无效元组比这更多，vacuum 将需要多次遍历表的索引，从而显著增加所需的时间。从 PostgreSQL 版本 17 开始，没有 1 GB 的限制，自动真空可以使用基数树处理超过 1.79 亿个元组。

  元组标识符的大小为 6 个字节。要估算对表的索引执行 vacuum 操作所需的内存，请查询 `pg_stat_all_tables.n_dead_tup` 来找到无效元组的数量，然后将该数字乘以 6，以确定单次对索引执行 vacuum 操作所需的内存。您可以使用以下查询：

  ```
  SELECT
      relname AS table_name,
  ```

```
    n_dead_tup,
    pg_size_pretty(n_dead_tup * 6) AS estimated_memory
FROM
    pg_stat_all_tables
WHERE
    relname = 'name_of_the_table';
```

- autovacuum_work_mem 参数可与 autovacuum_max_workers 参数结合使用。autovacuum_max_workers 中的每个工作进程均可使用您分配的内存。如果您有多个小型表，请分配更多的 autovacuum_max_workers 和更少的 autovacuum_work_mem。如果您拥有大型表（大于 100 GB），请分配更多内存和更少的工作进程。您需要分配有足够的内存才能对最大的表成功完成操作。因此，请确保工作进程和内存的组合等于要分配的总内存。

减少事务 ID 重叠的可能性

在一些情况下，与 Autovacuum 相关的参数组设置可能不够积极，无法阻止事务 ID 重叠。为解决此问题，RDS for PostgreSQL 提供了自动调整 Autovacuum 参数值的机制。适应性 autovacuum 是 RDS for PostgreSQL 的一项功能。在 PostgreSQL 文档中找到了 TransactionID 重叠的详细说明。

默认情况下，动态参数 rds.adaptive_autovacuum 设置为 ON 的 RDS for PostgreSQL 实例启用适应性 autovacuum。强烈建议您保持启用此选项。不过，要关闭适应性 Autovacuum 参数优化，请将 rds.adaptive_autovacuum 参数设置为 0 或 OFF。

即使在 Amazon RDS 优化 Autovacuum 参数时，仍可能出现事务 ID 重叠。鼓励您为事务 ID 重叠实施 Amazon CloudWatch 警报。有关更多信息，请参阅 Amazon 数据库博客上的贴子在 RDS for PostgreSQL 中为事务 ID 重叠实施预警系统。

启用适应性 autovacuum 参数优化后，当 CloudWatch 指标 MaximumUsedTransactionIDs 达到 autovacuum_freeze_max_age 参数值或 500000000 中的较大值时，Amazon RDS 将开始调整 autovacuum 参数。

如果表继续倾向于事务 ID 重叠，则 Amazon RDS 将继续调整 autovacuum 的参数。其中每次调整都会将更多资源专用于 Autovacuum 以避免重叠。Amazon RDS 更新以下与 Autovacuum 相关的参数：

- Autovacuum_vacuum_cost_delay
- Autovacuum_vacuum_cost_limit
- autovacuum_work_mem
- Autovacuum_naptime

仅当新值使 Autovacuum 更积极时，RDS 才会修改这些参数。在数据库实例上的内存中修改参数。不会更改参数组中的值。要查看当前内存中的设置，请使用 PostgreSQL SHOW SQL 命令。

当 Amazon RDS 修改其中任何 autovacuum 参数时，它会为受影响的数据库实例生成事件。此事件在 Amazon Web Services Management Console 上和通过 Amazon RDS API 显示。在 `MaximumUsedTransactionIDs` CloudWatch 指标返回的值低于阈值后，Amazon RDS 会将内存中与 autovacuum 相关的参数重置回参数组中指定的值。然后，它会生成另一个与此更改对应的事件。

确定数据库中的表是否需要 vacuum 操作

您可以使用以下查询显示数据库中解冻事务的数目。数据库的 `datfrozenxid` 行的 `pg_database` 列是显示在该数据库中的正常事务 ID 的下限。此列是数据库中每个表的 `relfrozenxid` 值的最小值。

```
SELECT datname, age(datfrozenxid) FROM pg_database ORDER BY age(datfrozenxid) desc
 limit 20;
```

例如，运行上述查询的结果可能如下所示。

```
datname     | age
mydb        | 1771757888
template0   | 1721757888
template1   | 1721757888
rdsadmin    | 1694008527
postgres    | 1693881061
(5 rows)
```

当数据库的期限达到 20 亿个事务 ID 时，事务 ID (XID) 重叠将出现，并且数据库将变成只读状态。您可以使用此查询来生成指标，并且一天可运行几次。默认情况下，将设置 Autovacuum 以确保事务期限不超过 200000000 ()。`autovacuum_freeze_max_age`

示例监控策略可能类似于：

- 将 `autovacuum_freeze_max_age` 值设置为 2 亿个事务。
- 如果表中的解冻事务达到 5 亿个，则会触发低严重性警报。这不是一个不合理的值，但它可能指示 Autovacuum 未保持同步。
- 如果表期限为 10 亿，这应被视为要采取操作的警报。通常，您出于性能原因，需要使期限更接近 `autovacuum_freeze_max_age`。建议您使用以下建议进行调查。
- 如果表达到 15 亿个未执行 vacuum 操作的事务，则这会触发高严重性警报。根据数据库使用事务 ID 的频率，此警报将指示系统运行 Autovacuum 的时间不多了。在这种情况下，建议您立即解决此问题。

如果表持续违反这些阈值，请进一步修改 autovacuum 参数。默认情况下，手动使用 VACUUM（已禁用基于成本的延迟）比使用默认的 Autovacuum 更积极，但对整个系统来说也更具侵入性。

我们建议执行下列操作：

- 了解和启用监控机制，以便您了解最早的事务的期限。

  有关创建提醒您事务 ID 重叠的过程的信息，请参阅 Amazon 数据库博客帖子 Implement an early warning system for transaction ID wraparound in Amazon RDS for PostgreSQL。
- 对于更复杂的表，在维护时段内定期执行手动 vacuum 冻结操作，并依赖 Autovacuum。有关执行手动 vacuum 冻结的信息，请参阅执行手动 vacuum 冻结。

确定哪些表当前符合 Autovacuum 条件

通常，它是需要执行 vacuum 操作的一个或两个表。其 `relfrozenxid` 值大于 `autovacuum_freeze_max_age` 中的事务数的表始终是 Autovacuum 的目标。否则，如果元组数因上一个 VACUUM 超出 vacuum 阈值而变得过时，则对表执行 vacuum 操作。

Autovacuum 阈值的定义如下：

```
Vacuum-threshold = vacuum-base-threshold + vacuum-scale-factor * number-of-tuples
```

其中，`vacuum base threshold` 为 `autovacuum_vacuum_threshold`，`vacuum scale factor` 为 `autovacuum_vacuum_scale_factor`，而 `number of tuples` 为 `pg_class.reltuples`。

在连接到数据库时，运行以下查询可查看 autovacuum 认为有资格执行 vacuum 操作的表的列表。

```
WITH vbt AS (SELECT setting AS autovacuum_vacuum_threshold FROM
pg_settings WHERE name = 'autovacuum_vacuum_threshold'),
vsf AS (SELECT setting AS autovacuum_vacuum_scale_factor FROM
pg_settings WHERE name = 'autovacuum_vacuum_scale_factor'),
fma AS (SELECT setting AS autovacuum_freeze_max_age FROM pg_settings WHERE name =
 'autovacuum_freeze_max_age'),
sto AS (select opt_oid, split_part(setting, '=', 1) as param,
split_part(setting, '=', 2) as value from (select oid opt_oid, unnest(reloptions)
 setting from pg_class) opt)
SELECT '"'||ns.nspname||'"."'||c.relname||'"' as relation,
pg_size_pretty(pg_table_size(c.oid)) as table_size,
age(relfrozenxid) as xid_age,
```

```
coalesce(cfma.value::float, autovacuum_freeze_max_age::float)
 autovacuum_freeze_max_age,
(coalesce(cvbt.value::float, autovacuum_vacuum_threshold::float) +
coalesce(cvsf.value::float,autovacuum_vacuum_scale_factor::float) * c.reltuples)
AS autovacuum_vacuum_tuples, n_dead_tup as dead_tuples FROM
pg_class c join pg_namespace ns on ns.oid = c.relnamespace
join pg_stat_all_tables stat on stat.relid = c.oid join vbt on (1=1) join vsf on (1=1)
 join fma on (1=1)
left join sto cvbt on cvbt.param = 'autovacuum_vacuum_threshold' and c.oid =
 cvbt.opt_oid
left join sto cvsf on cvsf.param = 'autovacuum_vacuum_scale_factor' and c.oid =
 cvsf.opt_oid
left join sto cfma on cfma.param = 'autovacuum_freeze_max_age' and c.oid = cfma.opt_oid
WHERE c.relkind = 'r' and nspname <> 'pg_catalog'
AND (age(relfrozenxid) >= coalesce(cfma.value::float, autovacuum_freeze_max_age::float)
OR coalesce(cvbt.value::float, autovacuum_vacuum_threshold::float) +
coalesce(cvsf.value::float,autovacuum_vacuum_scale_factor::float) *
c.reltuples <= n_dead_tup)
ORDER BY age(relfrozenxid) DESC LIMIT 50;
```

确定 Autovacuum 当前是否正在运行以及运行时长

如果需要手动对表执行 vacuum 操作，确保确定 autovacuum 当前是否正在运行。如果它当前正在运行，则您可能需要调整参数以使其更高效地运行，或者暂时禁用 autovacuum 以便手动运行 VACUUM。

使用以下查询来确定 Autovacuum 是否正在运行、它已运行的时长以及它是否正在等待其他会话。

```
SELECT datname, usename, pid, state, wait_event, current_timestamp - xact_start AS
 xact_runtime, query
FROM pg_stat_activity
WHERE upper(query) LIKE '%VACUUM%'
ORDER BY xact_start;
```

在运行查询后，您应看到类似以下内容的输出。

```

 datname | usename  |  pid  | state  | wait_event |       xact_runtime       | query
 --------+----------+-------+--------+------------+-------------------------
+---------------------------------------------------------------------------------------
 mydb    | rdsadmin | 16473 | active |            | 33 days 16:32:11.600656 |
 autovacuum: VACUUM ANALYZE public.mytable1 (to prevent wraparound)
```

```
mydb     | rdsadmin | 22553 | active |              | 14 days 09:15:34.073141 |
autovacuum: VACUUM ANALYZE public.mytable2 (to prevent wraparound)
mydb     | rdsadmin | 41909 | active |              | 3 days 02:43:54.203349  |
autovacuum: VACUUM ANALYZE public.mytable3
mydb     | rdsadmin |   618 | active |              | 00:00:00                |
SELECT datname, usename, pid, state, wait_event, current_timestamp - xact_start AS
xact_runtime, query+
         |          |       |        |              |                         | FROM
pg_stat_activity
           +
         |          |       |        |              |                         | WHERE
query like '%VACUUM%'
          +
         |          |       |        |              |                         | ORDER BY
xact_start;
      +
```

有多个问题可能会导致长时间运行 Autovacuum 会话（即，多天）。最常见的问题是，对于表的大小或更新速率来说，设置的 maintenance_work_mem 参数值太小。

建议您使用以下公式来设置 maintenance_work_mem 参数值。

```
GREATEST({DBInstanceClassMemory/63963136*1024},65536)
```

短时间运行的 Autovacuum 会话还可以指示以下问题：

- 它可以指示，对于工作负载而言，autovacuum_max_workers 不足。在这种情况下，您将需要指示工作线程数。
- 它可以指示存在索引损坏（autovacuum 将发生崩溃并在同一关系上重新启动，但毫无进展）。在这种情况下，运行手动 vacuum freeze verbose *table* 以查看准确原因。

执行手动 vacuum 冻结

您可能需要对已具有正在运行的 vacuum 进程的表执行手动 vacuum 操作。如果您已使用接近 20 亿个事务（或高于您监控的任何阈值）的期限标识表，则这会很有用。

以下步骤是指导原则，此过程存在几种变化。例如，在测试期间，假设您发现设定的 maintenance_work_mem 参数值过小，并且您需要立即对表采取措施。不过，可能您不希望此时恢复实例。通过使用前几节中的查询，您可以确定哪个表存在问题，并找到长时间运行的 Autovacuum

会话。您知道您需要更改 maintenance_work_mem 参数设置，但您还需要立即采取行动，对有问题的表执行 vacuum 操作。以下过程说明了在此情况下应采取的措施。

手动执行 vacuum 冻结

1. 打开针对包含要执行 vacuum 操作的表的数据库的两个会话。对于第二个会话，使用“screen”或其他维护会话的实用工具 (如果您的连接已中断)。
2. 在第一个会话中，获取正在表上运行的 autovacuum 会话的进程 ID (PID)。

   运行以下查询可获取 Autovacuum 会话的 PID。

   ```
   SELECT datname, usename, pid, current_timestamp - xact_start
   AS xact_runtime, query
   FROM pg_stat_activity WHERE upper(query) LIKE '%VACUUM%' ORDER BY
   xact_start;
   ```
3. 在第二个会话中，计算该操作所需的内存量。在此示例中，我们确定自己最多可以为该操作使用 2GB 的内存，因此，我们将当前会话的 maintenance_work_mem 设置为 2 GB。

   ```
   SET maintenance_work_mem='2 GB';
   SET
   ```
4. 在第二个会话中，为表发出 vacuum freeze verbose 命令。详细设置很有用，因为虽然 PostgreSQL 中当前没有进度报告，但您可以查看活动。

   ```
   \timing on
   Timing is on.
   vacuum freeze verbose pgbench_branches;
   ```

   ```
   INFO:  vacuuming "public.pgbench_branches"
   INFO:  index "pgbench_branches_pkey" now contains 50 row versions in 2 pages
   DETAIL:  0 index row versions were removed.
   0 index pages have been deleted, 0 are currently reusable.
   CPU 0.00s/0.00u sec elapsed 0.00 sec.
   INFO:  index "pgbench_branches_test_index" now contains 50 row versions in 2 pages
   DETAIL:  0 index row versions were removed.
   0 index pages have been deleted, 0 are currently reusable.
   CPU 0.00s/0.00u sec elapsed 0.00 sec.
   INFO:  "pgbench_branches": found 0 removable, 50 nonremovable row versions
        in 43 out of 43 pages
   DETAIL:  0 dead row versions cannot be removed yet.
   ```

```
There were 9347 unused item pointers.
0 pages are entirely empty.
CPU 0.00s/0.00u sec elapsed 0.00 sec.
VACUUM
Time: 2.765 ms
```

5. 在第一个会话中，如果 autovacuum 阻止 vacuum 会话，`pg_stat_activity` 显示 vacuum 会话的等待为 T。在此情况下，您需要终止 autovacuum 进程，如下所示。

```
SELECT pg_terminate_backend('the_pid');
```

> **Note**
>
> 某些较低版本的 Amazon RDS 无法使用上述命令终止 autovacuum 进程，并且会失败，显示以下错误：`ERROR: 42501: must be a superuser to terminate superuser process LOCATION: pg_terminate_backend, signalfuncs.c:227`。

此时，您的会话将开始。由于此表可能位于其工作列表中的最高位置，因此 Autovacuum 将立即重新启动。

6. 在第二个会话中启动您的 `vacuum freeze verbose` 命令，然后终止第一个会话中的 autovacuum 过程。

在 Autovacuum 正在运行时重新为表建立索引

如果索引已损坏，Autovacuum 将继续处理表并失败。在此情况下，如果您尝试执行手动 vacuum 操作，您将收到一条与以下内容类似的错误消息。

```
postgres=>  vacuum freeze pgbench_branches;
ERROR: index "pgbench_branches_test_index" contains unexpected
   zero page at block 30521
HINT: Please REINDEX it.
```

当索引损坏并且 autovacuum 尝试在表上运行时，您将处理已经正在运行的 autovacuum 会话。在您发出 REINDEX 命令时，将删除表上的排他锁。将阻止写入操作以及使用该特定索引的读取操作。

在对表运行 Autovacuum 时重新为表建立索引

1. 打开针对包含要执行 vacuum 操作的表的数据库的两个会话。对于第二个会话，使用“screen”或其他维护会话的实用工具 (如果您的连接已中断)。
2. 在第一个会话中，获取正在表上运行的 Autovacuum 会话的 PID。

   运行以下查询可获取 Autovacuum 会话的 PID。

   ```
   SELECT datname, usename, pid, current_timestamp - xact_start
   AS xact_runtime, query
   FROM pg_stat_activity WHERE upper(query) like '%VACUUM%' ORDER BY
   xact_start;
   ```
3. 在第二个会话中，发出 reindex 命令。

   ```
   \timing on
   Timing is on.
   reindex index pgbench_branches_test_index;
   REINDEX
     Time: 9.966 ms
   ```
4. 在第一个会话中，如果 autovacuum 阻止该过程，您将在 `pg_stat_activity` 中看到 vacuum 会话的等待为“T”。在此情况下，您将终止 autovacuum 过程。

   ```
   SELECT pg_terminate_backend('the_pid');
   ```

   此时，您的会话将开始。由于此表可能位于其工作列表中的最高位置，因此，了解 Autovacuum 将立即重新启动很重要。
5. 在第二个会话中启动您的命令，然后终止第一个会话中的 autovacuum 过程。

使用大型索引管理 autovacuum

作为其操作的一部分，在对表运行 autovacuum 时会执行多个 vacuum 阶段。在清理表之前，首先对所有索引执行 vacuum 操作。删除多个大型索引时，此阶段会消耗大量的时间和资源。因此，作为最佳实践，请务必控制表上的索引数量并消除未使用的索引。

对于此过程，请先检查总体索引大小。然后，确定是否存在可以删除的潜在未使用索引，如以下示例所示。

检查表及其索引的大小

```
postgres=> select pg_size_pretty(pg_relation_size('pgbench_accounts'));
pg_size_pretty
6404 MB
(1 row)
```

```
postgres=> select pg_size_pretty(pg_indexes_size('pgbench_accounts'));
pg_size_pretty
11 GB
(1 row)
```

在此示例中，索引的大小大于表的大小。这种差异可能会导致性能问题，因为索引膨胀或未使用，这会影响 autovacuum 以及插入操作。

检查是否有未使用的索引

使用 pg_stat_user_indexes 视图，您可以检查 idx_scan 列使用索引的频率。在以下示例中，未使用索引的 idx_scan 值为 0。

```
postgres=> select * from pg_stat_user_indexes where relname = 'pgbench_accounts' order
 by idx_scan desc;

relid  | indexrelid | schemaname | relname          | indexrelname          | idx_scan
 | idx_tup_read | idx_tup_fetch
-------+------------+------------+------------------+-----------------------+----------
+--------------+---------------
16433  | 16454      | public     | pgbench_accounts | index_f               | 6
 | 6            | 0
16433  | 16450      | public     | pgbench_accounts | index_b               | 3
 | 199999       | 0
16433  | 16447      | public     | pgbench_accounts | pgbench_accounts_pkey | 0
 | 0            | 0
16433  | 16452      | public     | pgbench_accounts | index_d               | 0
 | 0            | 0
16433  | 16453      | public     | pgbench_accounts | index_e               | 0
 | 0            | 0
16433  | 16451      | public     | pgbench_accounts | index_c               | 0
 | 0            | 0
16433  | 16449      | public     | pgbench_accounts | index_a               | 0
 | 0            | 0
(7 rows)
```

```
postgres=> select schemaname, relname, indexrelname, idx_scan from pg_stat_user_indexes
 where relname = 'pgbench_accounts' order by idx_scan desc;

schemaname  | relname          | indexrelname          | idx_scan
------------+------------------+-----------------------+----------
public      | pgbench_accounts | index_f               | 6
public      | pgbench_accounts | index_b               | 3
public      | pgbench_accounts | pgbench_accounts_pkey | 0
public      | pgbench_accounts | index_d               | 0
public      | pgbench_accounts | index_e               | 0
public      | pgbench_accounts | index_c               | 0
public      | pgbench_accounts | index_a               | 0
(7 rows)
```

> **Note**
>
> 这些统计数据自统计数据重置之时起开始递增。假设您的索引仅用于业务季度末或仅用于特定报告。自从统计数据重置后，此索引可能就没有使用过。有关更多信息，请参阅统计数据函数。用于强制唯一性的索引不会执行扫描，也不应被标识为未使用的索引。要识别未使用的索引，您应该对应用程序及其查询有深入的了解。

要检查数据库上次重置统计数据的时间，请使用 `pg_stat_database`

```
postgres=> select datname, stats_reset from pg_stat_database where datname =
 'postgres';

datname   | stats_reset
----------+-------------------------------
postgres  | 2022-11-17 08:58:11.427224+00
(1 row)
```

尽快对表执行 vacuum 操作

RDS for PostgreSQL 12 及更高版本

如果大型表中的索引过多，则您的数据库实例可能接近事务 ID 重叠（XID），也就是 XID 计数器变为零时。如果不进行检查，这种情况可能导致数据丢失。但是，您可以在不清理索引的情况下快速对表执

行 vacuum 操作。在 RDS for PostgreSQL 12 及更高版本中，可以将 VACUUM 与 INDEX_CLEANUP 子句结合使用。

```
postgres=> VACUUM (INDEX_CLEANUP FALSE, VERBOSE TRUE) pgbench_accounts;

INFO: vacuuming "public.pgbench_accounts"
INFO: table "pgbench_accounts": found 0 removable, 8 nonremovable row versions in 1 out
 of 819673 pages
DETAIL: 0 dead row versions cannot be removed yet, oldest xmin: 7517
Skipped 0 pages due to buffer pins, 0 frozen pages.
CPU: user: 0.01 s, system: 0.00 s, elapsed: 0.01 s.
```

如果 autovacuum 会话已在运行，则必须将其终止才能开始手动 VACUUM。有关执行手动 vacuum 冻结的信息，请参阅执行手动 vacuum 冻结。

Note

定期跳过索引清理会导致索引膨胀，这会降低扫描性能。索引保留死行，而表保留死行指针。因此，pg_stat_all_tables.n_dead_tup 会一直增加，直到 autovacuum 或带有索引清理功能的手动 VACUUM 运行。作为最佳实践，请仅使用此过程来防止事务 ID 重叠。

RDS for PostgreSQL 11 及更低版本

但是，在 RDS for PostgreSQL 11 及更低版本中，让 vacuum 过程更快地完成的唯一方法是减少表上的索引数量。删除索引可能会影响查询计划。我们建议您先删除未使用的索引，然后在 XID 重叠非常接近时删除索引。vacuum 过程完成后，您可以重新创建这些索引。

其他影响 Autovacuum 的参数

以下查询将显示直接影响 Autovacuum 及其行为的一些参数的值。PostgreSQL 文档中完整介绍了 Autovacuum 参数。

```
SELECT name, setting, unit, short_desc
FROM pg_settings
WHERE name IN (
'autovacuum_max_workers',
'autovacuum_analyze_scale_factor',
'autovacuum_naptime',
'autovacuum_analyze_threshold',
```

```
'autovacuum_analyze_scale_factor',
'autovacuum_vacuum_threshold',
'autovacuum_vacuum_scale_factor',
'autovacuum_vacuum_threshold',
'autovacuum_vacuum_cost_delay',
'autovacuum_vacuum_cost_limit',
'vacuum_cost_limit',
'autovacuum_freeze_max_age',
'maintenance_work_mem',
'vacuum_freeze_min_age');
```

所有这些参数都会影响 Autovacuum，其中一些最重要的参数为：

- maintenance_work_mem
- Autovacuum_freeze_max_age
- Autovacuum_max_workers
- Autovacuum_vacuum_cost_delay
- Autovacuum_vacuum_cost_limit

设置表级别 Autovacuum 参数

您可以在表级别设置与 autovacuum 相关的存储参数，这可能优于更改整个数据库的行为。对于大型表，您可能需要设置主动设置，并且可能不希望 Autovacuum 对所有表的行为都相同。

以下查询将显示哪些表当前拥有表级别选项。

```
SELECT relname, reloptions
FROM pg_class
WHERE reloptions IS NOT null;
```

例如，对于比您的其他表大得多的表，这可能会很有用。假设您具有一个 300GB 表和另外 30 个小于 1GB 的表。在这种情况下，可以为大型表设置一些特定的参数，这样便无需更改整个系统的行为。

```
ALTER TABLE mytable set (autovacuum_vacuum_cost_delay=0);
```

这样便可通过使用系统中的更多资源来禁用此表的基于成本的 autovacuum 延迟。通常，每次达到 `autovacuum_cost_limit` 时，autovacuum 会暂停 `autovacuum_vacuum_cost_delay`。有关更多详细信息，请参阅 PostgreSQL 文档中有关基于成本的 vacuum 操作的内容。

记录 autovacuum 和 vacuum 活动

根据 `rds.force_autovacuum_logging_level` 参数中指定的级别，有关 autovacuum 活动的信息将发送到 `postgresql.log`。以下是此参数允许的值以及该值为默认设置的 PostgreSQL 版本：

- `disabled`（PostgreSQL 10、PostgreSQL 9.6）
- `debug5`, `debug4`, `debug3`, `debug2`, `debug1`
- `info`（PostgreSQL 12、PostgreSQL 11）
- `notice`
- `warning`（PostgreSQL 13 及更高版本）
- `error`、日志、`fatal`、`panic`

`rds.force_autovacuum_logging_level` 与 `log_autovacuum_min_duration` 参数结合使用。`log_autovacuum_min_duration` 参数的值为阈值（以毫秒为单位），超出该值后将记录 autovacuum 操作。设置为 `-1` 不会记录任何内容，而设置 0 将记录所有操作。和 `rds.force_autovacuum_logging_level` 一样，`log_autovacuum_min_duration` 的默认值取决于版本，如下所示：

- `10000 ms` - PostgreSQL 14、PostgreSQL 13、PostgreSQL 12 和 PostgreSQL 11
- `(empty)` - PostgreSQL 10 和 PostgreSQL 9.6 没有默认值

建议您将 `rds.force_autovacuum_logging_level` 设置为 `WARNING`。我们还建议您将 `log_autovacuum_min_duration` 设置为 1000 到 5000 之间的值。设置为 5000 的记录活动耗时将超过 5000 毫秒。如果由于冲突锁定或并行删除关系跳过了 autovacuum 操作，则并非 -1 的任何其他设置也会记录消息。有关更多信息，请参阅 PostgreSQL 文档中的 [Automatic Vacuuming]。

要对问题进行故障排除，可以将 `rds.force_autovacuum_logging_level` 参数更改为调试级别之一，从 `debug1` 到最高 `debug5` 以获取最详细的信息。我们建议您在短时间内使用调试设置，并且仅用于故障排除目的。要了解更多信息，请参阅 PostgreSQL 文档中的[何时记录]。

> **Note**
>
> PostgreSQL 允许 `rds_superuser` 账户查看 `pg_stat_activity` 中的 Autovacuum 会话。例如，您可识别并结束阻止命令运行或运行速度慢于手动发出的 vacuum 命令的 Autovacuum 会话。

了解对无效数据库使用 autovacuum 的行为

在 `pg_database` 目录的 `datconnlimit` 列中引入了一个新值 `-2`，来指示在 DROP DATABASE 操作过程中已被中断的数据库为无效的。

以下 RDS for PostgreSQL 版本中提供了这个新值：

- 15.4 及所有更高版本
- 14.9 及更高版本
- 13.12 及更高版本
- 12.16 及更高版本
- 11.21 及更高版本

无效的数据库不会影响 autovacuum 冻结有效数据库的功能的能力。Autovacuum 会忽略无效的数据库。因此，对于 PostgreSQL 环境中的所有有效数据库，常规 vacuum 操作将继续正常且高效地运行。

主题

- 监控事务 ID
- 调整监控查询
- 解决无效的数据库问题

监控事务 ID

`age(datfrozenxid)` 函数常用于监控数据库的事务 ID（XID）期限，以防止事务 ID 重叠。

由于无效数据库被排除在 autovacuum 之外，因此其事务 ID（XID）计数器可以达到最大值 `2 billion`，重叠到 `- 2 billion`，并无限期地继续此循环。监控事务 ID 重叠的典型查询可能如下所示：

```
SELECT max(age(datfrozenxid)) FROM pg_database;
```

但是，在为 `datconnlimit` 引入 -2 值后，无效的数据库可能会扭曲此查询的结果。由于这些数据库无效，而不应作为定期维护检查的一部分，因此它们可能会导致误报，致使您认为 `age(datfrozenxid)` 高于实际值。

调整监控查询

为确保准确的监控，应调整监控查询来排除无效的数据库。请遵循这一建议的查询：

```
SELECT
    max(age(datfrozenxid))
FROM
    pg_database
WHERE
    datconnlimit <> -2;
```

此查询可确保在 `age(datfrozenxid)` 计算中只考虑有效的数据库，从而真实地反映整个 PostgreSQL 环境中的事务 ID 期限。

解决无效的数据库问题

当尝试连接到无效的数据库时，可能会遇到类似于以下内容的错误消息：

```
postgres=> \c db1
connection to server at "mydb.xxxxxxxxxx.us-west-2.rds.amazonaws.com" (xx.xx.xx.xxx),
 port xxxx failed: FATAL:  cannot connect to invalid database "db1"
HINT:  Use DROP DATABASE to drop invalid databases.
Previous connection kept
```

此外，如果 `log_min_messages` 参数设置为 `DEBUG2` 或更高，您可能会注意到以下日志条目，表明 autovacuum 进程正在跳过无效的数据库：

```
2024-07-30 05:59:00 UTC::@:[32000]:DEBUG:  autovacuum: skipping invalid database "db6"
2024-07-30 05:59:00 UTC::@:[32000]:DEBUG:  autovacuum: skipping invalid database "db1"
```

要解决此问题，请按照尝试连接期间提供的 `HINT` 进行操作。使用 RDS 主账户或具有 `rds_superuser` 角色的数据库账户连接到任何有效的数据库，然后删除无效的数据库。

```
SELECT
    'DROP DATABASE ' || quote_ident(datname) || ';'
FROM
    pg_database
WHERE
    datconnlimit = -2 \gexec
```

识别并解决 RDS for PostgreSQL 中的积极真空拦截器

在 PostgreSQL 中，清理对于确保数据库良好运行状况至关重要，因为它可以回收存储并防止出现事务 ID 重叠问题。但是，有时可能会阻止真空清理操作按预期运行，这样可能会导致性能下降、存储膨胀，甚至会因事务 ID 重叠而影响您的数据库实例的可用性。因此，识别和解决这些问题对于实现最佳数据库性能和可用性具有至关重要的意义。阅读了解 Amazon RDS for PostgreSQL 环境中的自动真空功能，以了解有关自动真空的更多信息。

`postgres_get_av_diag()` 函数可用于识别会阻止或延迟积极 vacuum 进度的问题。提供了建议，其中可能包括用于解决可识别问题的命令，或者在无法识别问题时提供进一步诊断的指导。当龄期超过 RDS 的 5 亿个事务 ID 的自适应自动真空阈值时，将报告积极真空拦截器。

事务 ID 的龄期是多少？

事务 ID 的 `age()` 函数会计算自数据库 (`pg_database.datfrozenxid`) 或表 (`pg_class.relfrozenxid`) 的最早解冻事务 ID 以来发生的事务数。此值表示自上次积极真空操作以来的数据库活动，并突出显示即将到来的真空进程可能出现的工作负载。

什么是积极真空？

积极真空操作会对表中的所有页面进行全面扫描，包括在常规真空期间通常跳过的页面。这种彻底的扫描旨在“冻结”接近其最大龄期的事务 ID，从而有效地防止出现被称为事务 ID 重叠的情况。

要使 `postgres_get_av_diag()` 报告拦截器，拦截器必须至少有 5 亿个事务。

主题

在 RDS for PostgreSQL 中安装 autovacuum 监控和诊断工具

`postgres_get_av_diag()` 函数目前在以下 RDS for PostgreSQL 版本中可用：

- 17.2 及更高的 17 版本
- 16.7 及更高的 16 版本
- 15.11 及更高的 15 版本
- 14.16 及更高的 14 版本
- 13.19 及更高的 13 版本

要使用 `postgres_get_av_diag()`，请创建 `rds_tools` 扩展。

```
postgres=> CREATE EXTENSION rds_tools ;
CREATE EXTENSION
```

验证是否已安装扩展。

```
postgres=> \dx rds_tools
             List of installed extensions
   Name    | Version |  Schema   |                    Description
 ----------+---------+-----------
+-----------------------------------------------------------
 rds_tools |   1.8   | rds_tools | miscellaneous administrative functions for RDS
 PostgreSQL
 1 row
```

验证是否已创建函数。

```
postgres=> SELECT
    proname function_name,
    pronamespace::regnamespace function_schema,
    proowner::regrole function_owner
FROM
    pg_proc
WHERE
    proname = 'postgres_get_av_diag';
    function_name     | function_schema | function_owner
----------------------+-----------------+----------------
 postgres_get_av_diag | rds_tools       | rds_superuser
(1 row)
```

RDS for PostgreSQL 中的 postgres_get_av_diag() 函数

`postgres_get_av_diag()` 函数会检索有关 RDS for PostgreSQL 数据库中阻止或滞后的 autovacuum 进程的诊断信息。为了获得准确的结果，需要在具有最早事务 ID 的数据库中执行查询。有关使用具有最早事务 ID 的数据库的更多信息，请参阅[未连接到具有最早事务 ID 龄期的数据库](#)

```
SELECT
    blocker,
    DATABASE,
    blocker_identifier,
    wait_event,
    TO_CHAR(autovacuum_lagging_by, 'FM9,999,999,999') AS autovacuum_lagging_by,
    suggestion,
    suggested_action
FROM (
    SELECT
        *
    FROM
        rds_tools.postgres_get_av_diag ()
    ORDER BY
        autovacuum_lagging_by DESC) q;
```

`postgres_get_av_diag()` 函数将返回一个包含以下信息的表：

blocker

指定阻止真空的数据库活动的类别。

- 活动语句
- 事务中空闲
- 准备好的事务
- 逻辑复制槽
- 带有物理复制槽的只读副本
- 使用流式复制的只读副本
- 临时表

database

在适用和支持的情况下指定数据库的名称。这是其中活动正在进行并正在阻止或将阻止自动真空的数据库。这是您需要连接并执行操作的数据库。

blocker_identifier

指定正在阻止或将阻止自动真空的活动的标识符。该标识符可以是进程 ID 以及 SQL 语句、准备好的事务、只读副本的 IP 地址以及复制槽的名称（逻辑或物理）。

wait_event

指定阻止会话的等待事件，适用于以下拦截器：

- 活动语句
- 事务中空闲

autovacum_lagging_by

指定自动真空在每个类别的积压工作中滞后的事务数。

suggestion

指定解除拦截器的建议。这些说明包括活动所在的数据库的名称（如果适用）、会话的进程 ID（PID）（如果适用）以及要执行的操作。

suggested_action

建议需要执行哪些操作来解除拦截器。

在 RDS for PostgreSQL 中解决可识别的真空拦截器

自动真空会执行积极真空操作，并将事务 ID 的龄期降低到低于由 RDS 实例的 `autovacuum_freeze_max_age` 参数指定的阈值。您可以使用 Amazon CloudWatch 指标 `MaximumUsedTransactionIDs` 跟踪此龄期。

要查找 Amazon RDS 实例的 `autovacuum_freeze_max_age` 设置（默认为 2 亿个事务 ID），您可以使用以下查询：

```
SELECT
    TO_CHAR(setting::bigint, 'FM9,999,999,999') autovacuum_freeze_max_age
FROM
    pg_settings
WHERE
    name = 'autovacuum_freeze_max_age';
```

请注意，`postgres_get_av_diag()` 仅在龄期超过 Amazon RDS 的 5 亿个事务 ID 的自适应自动真空阈值时检查积极真空拦截器。要使 `postgres_get_av_diag()` 检测拦截器，拦截器必须至少有 5 亿个事务。

postgres_get_av_diag() 函数可识别以下类型的拦截器：

主题

- 活动语句
- 事务中空闲
- 准备好的事务
- 逻辑复制槽
- 只读副本
- 临时表

活动语句

在 PostgreSQL 中，活动语句是数据库当前正在执行的 SQL 语句。这包括查询、事务或任何正在进行的操作。通过 pg_stat_activity 进行监控时，状态列指示具有相应 PID 的进程处于活动状态。

postgres_get_av_diag() 函数在识别系活动语句的语句时会显示类似于以下内容的输出。

```
blocker                  | Active statement
database                 | my_database
blocker_identifier       | SELECT pg_sleep(20000);
wait_event               | Timeout:PgSleep
autovacuum_lagging_by    | 568,600,871
suggestion               | Connect to database "my_database", review carefully and you
 may consider terminating the process using suggested_action. For more information, see
 Working with PostgreSQL autovacuum in the Amazon RDS User Guide.
suggested_action         | {"SELECT pg_terminate_backend (29621);"}
```

建议采取的措施

按照 suggestion 列中的指导，用户可以连接到存在活动语句的数据库，并按照 suggested_action 列中的指定，建议仔细查看终止会话的选项。如果终止是安全的，则可以使用 pg_terminate_backend() 函数终止会话。此操作可由管理员（如 RDS 主账户）或具有所需 pg_terminate_backend() 权限的用户执行。

> **Warning**
>
> 已终止的会话将撤消 (ROLLBACK) 其所做的更改。根据您的要求，您可能需要重新运行相应语句。但是，建议只有在自动真空过程完成其积极真空操作之后才这样做。

事务中空闲

事务中空闲语句是指已打开显式事务（例如通过发出 BEGIN 语句）、执行了一些工作，并且现在正在等待客户端传递更多工作或通过发出 COMMIT、ROLLBACK 或 END（这将导致隐式 COMMIT）来发出事务结束信号的任何会话。

postgres_get_av_diag() 函数在将 idle in transaction 语句识别为拦截器时，显示类似于以下内容的输出。

```
blocker                | idle in transaction
database               | my_database
blocker_identifier     | INSERT INTO tt SELECT * FROM tt;
wait_event             | Client:ClientRead
autovacuum_lagging_by  | 1,237,201,759
suggestion             | Connect to database "my_database", review carefully and you
 may consider terminating the process using suggested_action. For more information, see
 Working with PostgreSQL autovacuum in the Amazon RDS User Guide.
suggested_action       | {"SELECT pg_terminate_backend (28438);"}
```

建议采取的措施

如 suggestion 列中所示，您可以连接到存在事务中空闲会话的数据库，并使用 pg_terminate_backend() 函数终止会话。该用户可以是您的管理员（RDS 主账户）用户或具有 pg_terminate_backend() 权限的用户。

> **Warning**
>
> 已终止的会话将撤消 (ROLLBACK) 其所做的更改。根据您的要求，您可能需要重新运行相应语句。但是，建议只有在自动真空过程完成其积极真空操作之后才这样做。

准备好的事务

PostgreSQL 允许属于称为准备好的事务的两阶段提交策略一部分的事务。通过将 max_prepared_transactions 参数设置为非零值即可启用这些功能。准备好的事务旨在确保事务的持久性，即使在数据库崩溃、重启或客户端断开连接后仍可用。与定期事务一样，它们会被分配一个事务 ID，并且可能会影响自动真空。如果保持已准备状态，则自动真空将无法执行冻结，这样可能会导致事务 ID 重叠。

当事务被无限期地准备而没有被事务管理器解决时，它们将成为孤立的已准备事务。解决此问题的唯一方法是分别使用 COMMIT PREPARED 或 ROLLBACK PREPARED 命令提交或回滚事务。

> **Note**
>
> 请注意，在准备事务期间进行的备份在还原后仍将包含该事务。有关如何查找和关闭此类事务的信息，请参阅以下信息。

postgres_get_av_diag() 函数在识别出系已准备事务的拦截器时显示以下输出。

```
blocker                | Prepared transaction
database               | my_database
blocker_identifier     | myptx
wait_event             | Not applicable
autovacuum_lagging_by  | 1,805,802,632
suggestion             | Connect to database "my_database" and consider either COMMIT
 or ROLLBACK the prepared transaction using suggested_action. For more information, see
 Working with PostgreSQL autovacuum in the Amazon RDS User Guide.
suggested_action       | {"COMMIT PREPARED 'myptx';",[OR],"ROLLBACK PREPARED 'myptx';"}
```

建议采取的措施

如建议列中所述，连接到已准备事务所在的数据库。根据 suggested_action 列，仔细查看是执行 COMMIT 还是 ROLLBACK，并执行适当的操作。

为了总体上监控已准备事务，PostgreSQL 提供了一个名为 pg_prepared_xacts 的目录视图。您可以使用以下查询来查找已准备事务。

```
SELECT
    gid,
    prepared,
    owner,
    database,
    transaction AS oldest_xmin
FROM
    pg_prepared_xacts
ORDER BY
    age(transaction) DESC;
```

逻辑复制槽

复制槽的用途是保存未使用的更改，直到它们被复制到目标服务器。有关更多信息，请参阅 PostgreSQL 的 Logical replication。

有两种类型的逻辑复制槽。

非活动逻辑复制槽

复制终止后，无法删除未使用的事务日志，且复制槽变为非活动状态。尽管订阅用户当前未使用非活动逻辑复制槽，但它仍保留在服务器上，从而保留 WAL 文件并防止删除旧的事务日志。这样会增加磁盘使用量，特别是阻止自动真空清理内部目录表，因为系统必须保护 LSN 信息不被覆盖。如果不加以解决，可能会导致目录臃肿、性能下降和重叠真空风险增加，从而可能导致事务停机。

活动但速度较慢的逻辑复制槽

有时，由于逻辑复制的性能下降，目录中无效元组的删除会延迟。这种复制延迟会减慢 `catalog_xmin` 的更新速度，并可能导致目录臃肿和重叠真空。

`postgres_get_av_diag()` 函数在找到逻辑复制槽作为拦截器时，会显示类似于以下内容的输出。

```
blocker               | Logical replication slot
database              | my_database
blocker_identifier    | slot1
wait_event            | Not applicable
autovacuum_lagging_by | 1,940,103,068
suggestion            | Ensure replication is active and resolve any lag for the slot
 if active. If inactive, consider dropping it using the command in suggested_action.
 For more information, see Working with PostgreSQL autovacuum in the Amazon RDS User
 Guide.
suggested_action      | {"SELECT pg_drop_replication_slot('slot1') FROM
 pg_replication_slots WHERE active = 'f';"}
```

建议采取的措施

要解决此问题，请检查复制配置中是否存在可能终止应用进程的目标架构或数据问题。最常见的原因如下：

- 缺少列
- 数据类型不兼容
- 数据不匹配
- 缺少表

如果该问题与基础设施问题有关：

- 网络问题 – [如何解决 Amazon RDS 数据库处于不兼容网络状态的问题？](#)。

- 数据库或数据库实例不可用的原因如下：
  - 副本实例用尽存储空间 – 有关添加存储的信息，请查看 Amazon RDS DB 实例存储空间不足。
  - 不兼容参数 – 有关如何能够解决该问题的更多信息，请查看如何修复一直处于不兼容参数状态的 Amazon RDS 数据库实例问题？。

如果您的实例位于 Amazon 网络之外或 Amazon EC2 上，请咨询您的管理员，了解如何解决可用性或基础设施相关问题。

删除非活动槽

> **Warning**
>
> 注意：在删除复制槽之前，请仔细确保其没有正在进行的复制、处于非活动状态且处于不可恢复状态。过早删除槽可能会中断复制或导致数据丢失。

确认不再需要复制槽后，将其删除以允许自动真空继续运行。条件 `active = 'f'` 可确保仅删除非活动槽。

```
SELECT pg_drop_replication_slot('slot1') WHERE active ='f'
```

只读副本

为 Amazon RDS 只读副本启用 `hot_standby_feedback` 设置后，它会阻止主数据库上的自动真空删除只读副本上运行的查询可能仍需要的死行。这样会影响所有类型的物理只读副本，包括那些带有或不带由复制槽管理的只读副本。此行为是必需的，因为在备用副本上运行的查询要求这些行在主副本上保持可用，从而防止发生查询冲突和取消。

带有物理复制槽的只读副本

带有物理复制槽的只读副本可显著增强 RDS for PostgreSQL 中复制的可靠性和稳定性。这些槽可确保主数据库保留基本的预写日志文件，直到副本处理它们，即使在网络中断期间也能保持数据一致性。

从 RDS for PostgreSQL 版本 14 开始，所有副本均使用复制槽。在早期版本中，只有跨区域副本使用复制槽。

`postgres_get_av_diag()` 函数在找到以物理复制槽作为拦截器的只读副本时，会显示类似于以下内容的输出。

```
blocker                | Read replica with physical replication slot
database               |
blocker_identifier     | rds_us_west_2_db_xxxxxxxxxxxxxxxxxxxx
wait_event             | Not applicable
autovacuum_lagging_by  | 554,080,689
suggestion             | Run the following query on the replica
 "rds_us_west_2_db_xxxxxxxxxxxxxxxxxxxx" to find the long running query:

                       | SELECT * FROM pg_catalog.pg_stat_activity WHERE
 backend_xmin::text::bigint = 757989377;

                       | Review carefully and you may consdier terminating the query on
 read replica using suggested_action. For more information, see Working with PostgreSQL
 autovacuum in the Amazon RDS User Guide.                                 +
           |
suggested_action       | {"SELECT pg_terminate_backend(pid) FROM
 pg_catalog.pg_stat_activity WHERE backend_xmin::text::bigint = 757989377;","
                                                                       +
                       | [OR]

                                                 +
                       | ","Disable hot_standby_feedback","

                                                 +
                       | [OR]

                                                 +
                       | ","Delete the read replica if not needed"}
```

使用流式复制的只读副本

Amazon RDS 允许在较旧版本（最高为版本 13）中设置不带物理复制槽的只读副本。这种方法通过允许主数据库更积极地回收 WAL 文件来减少开销，这在磁盘空间有限的环境中是有利的，并可以容忍偶尔发生的 ReplicaLag。但是，如果没有槽，备用副本必须保持同步，以免丢失 WAL 文件。Amazon RDS 使用已存档的 WAL 文件来帮助副本在落后时赶上，但是此过程需要仔细监控，而且可能速度很慢。

`postgres_get_av_diag()` 函数在找到作为拦截器的流式只读副本时，会显示类似于以下内容的输出。

```
blocker                | Read replica with streaming replication slot
database               | Not applicable
```

```
blocker_identifier    | xx.x.x.xxx/xx
wait_event            | Not applicable
autovacuum_lagging_by | 610,146,760
suggestion            | Run the following query on the replica "xx.x.x.xxx" to find the
 long running query:
                                                                                              +
                      | SELECT * FROM pg_catalog.pg_stat_activity WHERE
 backend_xmin::text::bigint = 348319343;

               +
                      | Review carefully and you may consdier terminating the query on
 read replica using suggested_action. For more information, see Working with PostgreSQL
 autovacuum in the Amazon RDS User Guide.                                              +
                      |
suggested_action      | {"SELECT pg_terminate_backend(pid) FROM
 pg_catalog.pg_stat_activity WHERE backend_xmin::text::bigint = 348319343;","

                       +
                      | [OR]

                                                                                              +
                      | ","Disable hot_standby_feedback","

                                                                                              +
                      | [OR]

                                                                                              +
                      | ","Delete the read replica if not needed"}
```

建议采取的措施

按照 `suggested_action` 列中的建议，仔细查看用于解除阻止自动真空的这些选项。

- 终止查询 – 按照建议列中的指导，您可以连接到只读副本，如 suggested_action 列中指定，建议仔细查看终止会话的选项。如果终止被视为是安全的，则可以使用 `pg_terminate_backend()` 函数终止会话。此操作可由管理员（如 RDS 主账户）或具有所需 pg_terminate_backend() 权限的用户执行。

  您可以在只读副本上运行以下 SQL 命令来终止用于阻止主数据库上的真空清理旧行的查询。在函数的输出中报告 `backend_xmin` 的值：

  ```
  SELECT
      pg_terminate_backend(pid)
  ```

```
FROM
    pg_catalog.pg_stat_activity
WHERE
    backend_xmin::text::bigint = backend_xmin;
```

- 禁用热备用反馈 – 如果 hot_standby_feedback 参数会导致明显的真空延迟，请考虑禁用该参数。

  hot_standby_feedback 参数允许只读副本将其查询活动通知主数据库，从而防止主数据库对备用数据库上使用的表或行执行清理操作。虽然这样可以确保备用数据库上的查询稳定性，但它可能会显著延迟主数据库上的清理时间。禁用此功能允许主数据库继续进行清理，而无需等待备用数据库赶上。但是，如果备用数据库尝试访问已被主数据库清理的行，则可能导致备用数据库上的查询取消或失败。
- 如果不需要只读副本，请将其删除 – 如果不再需要只读副本，则可以将其删除。这将消除相关的复制开销，并允许主数据库在不受副本阻碍的情况下回收事务日志。

临时表

使用 TEMPORARY 关键字创建的临时表驻留在临时架构中，例如 pg_temp_xxx，并且只能由创建它们的会话访问。会话结束时会删除临时表。但是，这些表对于 PostgreSQL 的自动真空流程是不可见的，必须由创建它们的会话手动清理。尝试从另一个会话中清空临时表是无效的。

在特殊情况下，存在一个临时表，但没有活动会话拥有该表。如果所属会话由于致命崩溃、网络问题或类似事件而意外结束，则可能无法清理临时表，从而会将其留作为“孤立”表。当 PostgreSQL 自动真空进程检测到孤立的临时表时，它会记录以下消息：

```
LOG: autovacuum: found orphan temp table \"%s\".\"%s\" in database \"%s\"
```

postgres_get_av_diag() 函数在将临时表识别为拦截器时，会显示类似于以下内容的输出。要使该函数正确显示与临时表相关的输出，需要在存在这些表的同一数据库中执行该函数。

```
blocker                 | Temporary table
database                | my_database
blocker_identifier      | pg_temp_14.ttemp
wait_event              | Not applicable
autovacuum_lagging_by   | 1,805,802,632
suggestion              | Connect to database "my_database". Review carefully, you
 may consider dropping temporary table using command in suggested_action. For more
 information, see Working with PostgreSQL autovacuum in the Amazon RDS User Guide.
```

```
suggested_action        | {"DROP TABLE ttemp;"}
```

建议采取的措施

按照输出的 suggestion 列中提供的说明识别并删除阻止自动真空运行的临时表。使用以下命令删除由 postgres_get_av_diag() 报告的临时表。根据 postgres_get_av_diag() 函数提供的输出替换表名称。

```
DROP TABLE my_temp_schema.my_temp_table;
```

以下查询可用于识别临时表：

```
SELECT
    oid,
    relname,
    relnamespace::regnamespace,
    age(relfrozenxid)
FROM
    pg_class
WHERE
relpersistence = 't'
ORDER BY
    age(relfrozenxid) DESC;
```

在 RDS for PostgreSQL 中解除无法识别的真空拦截器

本节探讨了可能阻碍清理取得进展的其他原因。postgres_get_av_diag() 函数目前无法直接识别这些问题。

主题

- 页面无效
- 索引不一致
- 事务速率异常高

页面无效

当 PostgreSQL 在访问页面时检测到该页面的校验和不匹配时，就会出现无效页面错误。内容无法读取，从而阻止自动真空冻结元组。这样实际上会停止清理过程。PostgreSQL 的日志中写入了以下错误：

```
WARNING:  page verification failed, calculated checksum YYYYY but expected XXXX
ERROR:  invalid page in block ZZZZZ of relation base/XXXXX/XXXXX
CONTEXT:  automatic vacuum of table myschema.mytable
```

确定对象类型

```
ERROR: invalid page in block 4305910 of relation base/16403/186752608
WARNING: page verification failed, calculated checksum 50065 but expected 60033
```

从错误消息中，路径 base/16403/186752608 提供以下信息：

- “base”是 PostgreSQL 数据目录下的目录名称。
- “16403”是数据库 OID，您可以在 pg_database 系统目录中进行查找。
- “186752608”是 relfilenode，您可以使用它在 pg_class 系统目录中查找架构和对象名称。

通过在受影响的数据库中检查以下查询的输出，可以确定对象类型。以下查询会检索 oid 为 186752608 的对象信息。将该 OID 替换为与您遇到的错误相关的 OID。

```
SELECT
    relname AS object_name,
    relkind AS object_type,
    nspname AS schema_name
FROM
    pg_class c
    JOIN pg_namespace n ON c.relnamespace = n.oid
WHERE
    c.oid = 186752608;
```

有关更多信息，请参阅 PostgreSQL 文档 [pg_class]，以了解所有支持的对象类型，如 pg_class 中的 relkind 列所示。

指南

此问题的最有效解决方案取决于特定 Amazon RDS 实例的配置以及受不一致页面影响的数据类型。

如果对象类型是索引：

建议重建索引。

- 使用 **CONCURRENTLY** 选项 – 在 PostgreSQL 版本 12 之前，重建索引需要独占的表锁定，从而限制对表的访问。在 PostgreSQL 版本 12 及更高版本中，CONCURRENTLY 选项允许行级别锁定，从而显著提高表的可用性。命令如下：

  ```
  REINDEX INDEX ix_name CONCURRENTLY;
  ```

  虽然 CONCURRENTLY 的破坏性较小，但在繁忙的表中其速度可能会变慢。如果可能，可以考虑在流量较低的时段构建索引。

  有关更多信息，请参阅 PostgreSQL REINDEX 文档。

- 使用 **INDEX_CLEANUP FALSE** 选项 – 如果索引很大并且估计需要大量时间才能完成，则您可以通过在排除索引的同时执行手动 VACUUM FREEZE 来解除阻止 autovacuum。PostgreSQL 版本 12 及更高版本中提供了此功能。

  绕过索引将允许您跳过索引不一致的真空过程并缓解重叠问题。但是，这样并不能解决潜在的无效页面问题。要完全应对和解决无效页面问题，您仍然需要重建索引。

如果对象类型为实体化视图：

如果实体化视图上出现无效页面错误，请登录到受影响的数据库并刷新以解决无效的页面：

刷新实体化视图：

```
REFRESH MATERIALIZED VIEW schema_name.materialized_view_name;
```

如果刷新失败，请尝试重新创建：

```
DROP MATERIALIZED VIEW schema_name.materialized_view_name;
CREATE MATERIALIZED VIEW schema_name.materialized_view_name AS query;
```

刷新或重新创建实体化视图可在不影响基础表数据的情况下还原该视图。

对于所有其他对象类型：

对于所有其他对象类型，请联系 Amazon 支持。

索引不一致

逻辑上不一致的索引可能会阻碍自动真空取得进展。在索引的真空阶段或 SQL 语句访问索引时，将记录以下错误或类似错误。

```
ERROR: right sibling's left-link doesn't match:block 5 links to 10 instead of expected
 2 in index ix_name
```

```
ERROR: failed to re-find parent key in index "XXXXXXXXXX" for deletion target page XXX
CONTEXT:  while vacuuming index index_name of relation schema.table
```

指南

在手动 `VACUUM FREEZE` 上使用 `INDEX_CLEANUP` 重建索引或跳过索引。有关如何重建索引的信息，请参阅如果对象类型是索引。

- 使用 CONCURRENTLY 选项 – 在 PostgreSQL 版本 12 之前，重建索引需要独占的表锁定，从而限制对表的访问。在 PostgreSQL 版本 12 及更高版本中，CONCURRENTLY 选项允许行级别锁定，从而显著提高表的可用性。命令如下：

  ```
  REINDEX INDEX ix_name CONCURRENTLY;
  ```

  虽然 CONCURRENTLY 的破坏性较小，但在繁忙的表中其速度可能会变慢。如果可能，可以考虑在流量较低的时段构建索引。有关更多信息，请参阅 PostgreSQL 文档中的 REINDEX。
- 使用 INDEX_CLEANUP FALSE 选项 – 如果索引很大并且估计需要大量时间才能完成，则您可以通过在排除索引的同时执行手动 VACUUM FREEZE 来解除阻止 autovacuum。PostgreSQL 版本 12 及更高版本中提供了此功能。

  绕过索引将允许您跳过索引不一致的真空过程并缓解重叠问题。但是，这样并不能解决潜在的无效页面问题。要完全应对和解决无效页面问题，您仍然需要重建索引。

事务速率异常高

在 PostgreSQL 中，事务速率高会显著影响自动真空的性能，从而导致无效元组的清理速度变慢，并增加事务 ID 重叠的风险。您可以通过衡量两个时间段之间的 `max(age(datfrozenxid))` 差异（通常为每秒）来监控事务速率。此外，您还可以使用 RDS 性能详情中的以下计数器指标来衡量事务速率（xact_commit 与 xact_rollback 之和），即事务总数。

计数器	类型	单位	指标
xact_commit	事务	每秒提交数	db.Transactions.xact_commit

计数器	类型	单位	指标
xact_rollback	事务	每秒回滚数	db.Transactions.xact_rollback

快速增加表示高事务负载，这可能会使自动真空不堪重负，从而导致膨胀、锁争用和潜在的性能问题。这样可能会以多种方式对自动真空进程产生负面影响：

- 表活动：正在经历清理操作的特定表可能会遇到大量事务，从而导致延迟。
- 系统资源：整个系统可能会过载，使自动真空难以访问必要的资源以便高效运行。

请考虑以下策略，以允许自动真空更有效地运行并跟上其任务：

1. 如果可能，请降低事务速率。请考虑在可行的情况下对类似的事务进行批处理或分组。
2. 在非高峰时段，每晚、每周或每两周通过手动 `VACUUM FREEZE` 操作来定位频繁更新的表。
3. 请考虑扩展您的实例类以分配更多系统资源来处理高事务量和自动真空操作。

解决 RDS for PostgreSQL 中的真空性能问题

本节讨论经常导致真空性能变慢的因素以及如何解决这些问题。

主题

- 清理大型索引
- 需要清理的表或数据库太多
- 正在运行积极真空（以防止重叠）

清理大型索引

VACUUM 按以下阶段顺序运行：初始化、堆扫描、索引和堆真空清理、索引清理、堆截断和最终清理。在堆扫描期间，该过程修剪页面、整理碎片并冻结它们。完成堆扫描后，VACUUM 将清理索引，将空页返回到操作系统，并执行最终的清理任务，例如对空闲空间映射表执行真空清理和更新统计信息。

对索引执行真空清理操作时，如果 `maintenance_work_mem`（或 `autovacuum_work_mem`）不足以处理索引，则可能需要多次扫描。在 PostgreSQL 16 及更早版本中，用于存储无效元组 ID 存在 1

GB 内存限制，对于大型索引通常需要多次扫描。PostgreSQL 17 引入了 `TidStore`，这会动态分配内存，而不是使用单一分配数组。此功能消除了 1 GB 限制，可以更高效地利用内存，并减少了每次索引操作中进行多次索引扫描的需求。

在 PostgreSQL 17 中，对于大型索引，如果可用内存无法一次性容纳整个索引的处理需求，那么仍可能需要多次扫描。通常，更大的索引往往包含更多需要多次扫描的无效元组。

检测缓慢的 vacuum 操作

`postgres_get_av_diag()` 函数可以检测 vacuum 操作何时因内存不足而运行缓慢。有关此函数的更多信息，请参阅[在 RDS for PostgreSQL 中安装 autovacuum 监控和诊断工具](#)。

当可用内存不足以一次性完成索引 vacuum 时，`postgres_get_av_diag()` 函数会发出以下通知。

`rds_tools` 1.8

```
NOTICE: Your database is currently running aggressive vacuum to prevent wraparound and
 it might be slow.
```

```
NOTICE: The current setting of autovacuum_work_mem is "XXX" and might not be
 sufficient. Consider increasing the setting, and if necessary, scaling up the Amazon
 RDS instance class for more memory.
        Additionally, review the possibility of manual vacuum with exclusion of indexes
 using (VACUUM (INDEX_CLEANUP FALSE, VERBOSE TRUE) table_name;).
```

`rds_tools` 1.9

```
NOTICE: Your database is currently running aggressive vacuum to prevent wraparound and
 it might be slow.
```

```
NOTICE: The current setting of autovacuum_work_mem is XX might not be sufficient.
 Consider increasing the setting to XXX, and if necessary, scaling up the RDS instance
 class for more
        memory. The suggested value is an estimate based on the current number of dead
 tuples for the table being vacuumed, which might not fully reflect the latest state.
 Additionally, review the possibility of manual
        vacuum with exclusion of indexes using (VACUUM (INDEX_CLEANUP FALSE, VERBOSE
 TRUE) table_name;). For more information, see
        Working with PostgreSQL autovacuum in the Amazon Amazon RDS User Guide
        .
```

> **Note**
>
> postgres_get_av_diag() 函数依靠 pg_stat_all_tables.n_dead_tup 来估计索引清理所需的内存量。

当 postgres_get_av_diag() 函数识别出由于 autovacuum_work_mem 不足而需要多次索引扫描的慢速 vacuum 操作时，它将生成以下消息：

```
NOTICE: Your vacuum is performing multiple index scans due to insufficient
 autovacuum_work_mem:XXX for index vacuuming.
        For more information, see Working with PostgreSQL autovacuum in the Amazon
 Amazon RDS User Guide.
```

指南

您可以使用手动 VACUUM FREEZE 应用以下解决方法来加快表的冻结速度。

增加用于清理的内存

正如 postgres_get_av_diag() 函数所建议的那样，建议增加 autovacuum_work_mem 参数以解决实例级别的潜在内存限制问题。虽然 autovacuum_work_mem 是一个动态参数，但请务必注意，要使新的内存设置生效，自动真空进程守护程序需要重新启动其工作线程。要实现这一点，请执行以下操作：

1. 确认新设置已落实到位。
2. 终止当前运行自动真空的进程。

这种方法可确保将调整后的内存分配应用于新的自动真空操作。

为了获得更直接的结果，请考虑在会话中手动执行 VACUUM FREEZE 操作，并增加 maintenance_work_mem 设置：

```
SET maintenance_work_mem TO '1GB';
VACUUM FREEZE VERBOSE table_name;
```

如果您使用的是 Amazon RDS，并且发现需要额外的内存来支持更高的 maintenance_work_mem 或 autovacuum_work_mem 值，请考虑升级到具有更多内存的实例类。这样可以提供必要的资源来增强手动和自动清理操作，从而提高整体真空和数据库性能。

禁用 INDEX_CLEANUP

PostgreSQL 版本 12 及更高版本中的手动 VACUUM 允许跳过索引清理阶段，而 PostgreSQL 版本 14 及更高版本中的紧急自动真空会根据 vacuum_failsafe_age 参数自动执行此操作。

> **Warning**
>
> 跳过索引清理会导致索引膨胀并对查询性能产生负面影响。为了缓解这种情况，可以考虑在维护时段内对受影响的索引重建索引或清理受影响的索引。

有关处理大型索引的更多指导，请参阅使用大型索引管理 autovacuum 中的文档。

并行索引清理

从 PostgreSQL 13 开始，默认情况下，可以使用手动 VACUUM 并行对索引进行清理，并为每个索引分配一个真空工件进程。但是，要使 PostgreSQL 确定真空操作是否符合并行执行的条件，必须满足特定标准：

- 必须至少有两个索引。
- max_parallel_maintenance_workers 参数应至少设置为 2。
- 索引大小必须超过 min_parallel_index_scan_size 限制，默认为 512 KB。

您可以根据 Amazon RDS 实例上可用的 vCPU 数量和表上的索引数量来调整 max_parallel_maintenance_workers 设置，以优化清理周转时间。

有关更多信息，请参阅 Parallel vacuuming in Amazon RDS for PostgreSQL and Amazon Aurora PostgreSQL。

需要清理的表或数据库太多

如 PostgreSQL 的自动真空进程守护程序文档中所述，自动真空进程守护程序通过多个进程运行。这包括一个持久性自动真空启动程序，负责为系统中的每个数据库启动自动真空工件进程。该启动程序安排这些工件大约每个数据库每 autovacuum_naptime 秒启动一次。

对于“N”个数据库，大约每 [autovacuum_naptime/N 秒] 就会有一个新工件开始工作。但是，并发工件的总数受 autovacuum_max_workers 设置的限制。如果需要清理的数据库或表的数量超过此限制，则一旦有工件可用，就会立即处理下一个数据库或表。

当许多大型表或数据库需要同时执行清理操作时，所有可用的自动真空工件可能会长时间被占用，从而会延迟其他表和数据库的维护。在事务速率较高的环境中，此瓶颈可能会迅速升级，并可能导致 Amazon RDS 实例中出现重叠真空问题。

当 `postgres_get_av_diag()` 检测到大量表或数据库时，它会提供以下建议：

```
NOTICE: Your database is currently running aggressive vacuum to prevent wraparound and
 it might be slow.
```

```
NOTICE: The current setting of autovacuum_max_workers:3 might not be sufficient.
 Consider increasing the setting and, if necessary, consider scaling up the Amazon RDS
 instance class for more workers.
```

指南

增加 autovacuum_max_workers

为了加快清理速度，我们建议调整 `autovacuum_max_workers` 参数以允许更多并发自动真空工件。如果性能瓶颈仍然存在，请考虑将 Amazon RDS 实例扩展到具有更多 vCPU 的类，这样可以进一步提高并行处理能力。

正在运行积极真空（以防止重叠）

PostgreSQL 中数据库（MaximumUsedTransactionIDs）的龄期仅会在成功完成积极真空（以防止重叠）时减少。在此真空结束之前，龄期将继续增加，具体取决于事务速率。

`postgres_get_av_diag()` 函数在检测到积极真空时会生成以下 `NOTICE`。但是，只有在真空处于活动状态至少两分钟后，它才会触发此输出。

```
NOTICE: Your database is currently running aggressive vacuum to prevent wraparound,
 monitor autovacuum performance.
```

有关积极真空的更多信息，请参阅[当积极真空已经在运行时](#)。

您可以使用以下查询验证是否正在进行主动清理：

```
SELECT
    a.xact_start AS start_time,
    v.datname "database",
    a.query,
```

```
    a.wait_event,
    v.pid,
    v.phase,
    v.relid::regclass,
    pg_size_pretty(pg_relation_size(v.relid)) AS heap_size,
    (
        SELECT
            string_agg(pg_size_pretty(pg_relation_size(i.indexrelid)) || ':' ||
 i.indexrelid::regclass || chr(10), ', ')
        FROM
            pg_index i
        WHERE
            i.indrelid = v.relid
    ) AS index_sizes,
    trunc(v.heap_blks_scanned * 100 / NULLIF(v.heap_blks_total, 0)) AS step1_scan_pct,
    v.index_vacuum_count || '/' || (
        SELECT
            count(*)
        FROM
            pg_index i
        WHERE
            i.indrelid = v.relid
    ) AS step2_vacuum_indexes,
    trunc(v.heap_blks_vacuumed * 100 / NULLIF(v.heap_blks_total, 0)) AS
 step3_vacuum_pct,
    age(CURRENT_TIMESTAMP, a.xact_start) AS total_time_spent_sofar
FROM
    pg_stat_activity a
    INNER JOIN pg_stat_progress_vacuum v ON v.pid = a.pid;
```

您可以通过检查输出中的查询列来确定这是否为积极真空（以防止重叠）。“防止重叠”一词表示这是一个积极真空。

```
query                   | autovacuum: VACUUM public.t3 (to prevent wraparound)
```

例如，假设您有一个事务龄期为 10 亿的拦截器，还有一个需要积极真空的表，以防止在相同的事务龄期出现重叠。此外，在事务龄期为 7.5 亿时，还有另一个拦截器。在事务龄期为 10 亿时清除拦截器后，事务龄期不会立即下降到 7.5 亿。它将保持较高的状态，直到需要积极真空或任何龄期超过 7.5 亿的事务完成为止。在此期间，PostgreSQL 集群的事务龄期将持续提高。真空过程完成后，事务龄期将降至 7.5 亿，但在进一步的清理完成之前，事务龄期将再次开始增加。只要这些条件仍然存在，此循环就会继续，直到事务龄期最终下降到为您的 Amazon RDS 实例配置的级别（由 `autovacuum_freeze_max_age` 指定）。

RDS for PostgreSQL 中的通知消息说明

`postgres_get_av_diag()` 函数提供以下通知消息：

当龄期尚未达到监控阈值时

默认情况下，`postgres_get_av_diag()` 识别拦截器的监控阈值为 5 亿个事务。如果 `postgres_get_av_diag()` 生成以下通知，则表示事务龄期尚未达到此阈值。

```
NOTICE: postgres_get_av_diag() checks for blockers that prevent aggressive vacuums
 only, it does so only after exceeding dvb_threshold which is 500,000,000 and age of
 this PostgreSQL cluster is currently at 2.
```

未连接到具有最早事务 ID 龄期的数据库

当连接到具有最早事务 ID 龄期的数据库时，`postgres_get_av_diag()` 函数可提供最准确的输出。`postgres_get_av_diag()` 报告的具有最早事务 ID 龄期的数据库将不同于您的案例中的“my_database”。如果您未连接到正确的数据库，则会生成以下通知：

```
NOTICE: You are not connected to the database with the age of oldest transaction
 ID. Connect to my_database database and run postgres_get_av_diag() for accurate
 reporting.
```

出于以下原因，连接到具有最早事务龄期的数据库非常重要：

- 识别临时表拦截器：由于临时表的元数据特定于每个数据库，因此它们通常位于在其中创建临时表的数据库中。但是，如果临时表恰好是最重要的拦截器，并且位于具有最早事务的数据库中，则可能会产生误导。连接到正确的数据库可确保准确识别临时表拦截器。
- 诊断缓慢真空：索引元数据和表计数信息是特定于数据库的，是诊断缓慢真空问题所必需的。

按龄期计算具有最早事务的数据库位于 rdsadmin 或 template0 数据库中

在某些情况下，`rdsadmin` 或 `template0` 数据库可能会被标识为具有最早事务 ID 龄期的数据库。如果发生这种情况，`postgres_get_av_diag()` 将发出以下通知：

```
NOTICE: The database with the age of oldest transaction ID is rdsadmin or template0,
 reach out to support if the reported blocker is in rdsadmin or template0.
```

验证列出的拦截器是否不是来自这两个数据库中的任何一个。如果据报告 `rdsadmin` 或 `template0` 中存在拦截器，请联系支持部门，因为这些数据库对用户不可访问，需要干预。

`rdsadmin` 或 `template0` 数据库包含最重要拦截器的可能性极小。

当积极真空已经在运行时

postgres_get_av_diag() 函数旨在报告何时正在运行积极真空进程，但它仅在真空处于活动状态至少 1 分钟后触发此输出。这种有意的延迟有助于减少误报的可能性。通过等待，该函数可确保仅报告有效、重要的真空，从而可更准确、更可靠地监控真空活动。

postgres_get_av_diag() 函数在检测到一个或多个正在进行的积极真空时会生成以下通知。

```
NOTICE: Your database is currently running aggressive vacuum to prevent wraparound,
 monitor autovacuum performance.
```

如通知中所述，请继续监控真空的性能。有关积极真空的更多信息，请参阅[正在运行积极真空（以防止重叠）](#)

当自动真空关闭时

如果数据库实例上禁用了自动真空，则 postgres_get_av_diag() 函数会生成以下通知：

```
NOTICE: Autovacuum is OFF, we strongly recommend to enable it, no restart is
 necessary.
```

Autovacuum 是 RDS for PostgreSQL 数据库实例的一项关键功能，可确保数据库顺畅运行。它会自动删除旧的行版本，回收存储空间，并防止表膨胀，从而帮助保持表和索引的高效性，以实现最佳性能。此外，它还可以防止发生事务 ID 重叠，这可能会停止 Amazon RDS 实例上的事务。禁用自动真空会导致数据库性能和稳定性的长期下降。建议您始终将其保持开启状态。有关更多信息，请参阅 [Understanding autovacuum in RDS for PostgreSQL environments](#)。

> **Note**
>
> 关闭自动真空并不会阻止积极真空。一旦您的表达到 autovacuum_freeze_max_age 阈值，仍会发生这种情况。

剩余的事务数严重不足

postgres_get_av_diag() 函数会在即将发生重叠真空时生成以下通知。当您的 Amazon RDS 实例距离可能拒绝新事务还有 1 亿个事务时，将发出此通知。

```
WARNING: Number of transactions remaining is critically low, resolve issues with
 autovacuum or perform manual VACUUM FREEZE before your instance stops accepting
 transactions.
```

需要立即采取行动以免发生数据库停机。您应该密切监控真空操作，并考虑在受影响的数据库上手动启动 `VACUUM FREEZE` 以防止事务失败。

使用 RDS for PostgreSQL 支持的日志记录机制

您可以设置多个参数、扩展和其他可配置项来记录 PostgreSQL 数据库实例中发生的活动。这些功能包括：

- `log_statement` 参数可用于记录 PostgreSQL 数据库中的用户活动。要了解有关 RDS for PostgreSQL 日志记录以及如何监控日志的更多信息，请参阅 RDS for PostgreSQL 数据库日志文件。
- `rds.force_admin_logging_level` 参数记录数据库实例上的数据库中由 Amazon RDS 内部用户 (rdsadmin) 执行的操作。它将输出写入 PostgreSQL 错误日志。允许的值包括 `disabled`、`debug5`、`debug4`、`debug3`、`debug2`、`debug1`、`info`、`notice`、`warning`、`error`、日志、`fatal` 和 `panic`。默认值为 `disabled`。
- 可以设置 `rds.force_autovacuum_logging_level` 参数以捕获 PostgreSQL 错误日志中的各种 autovacuum 操作。有关更多信息，请参阅 记录 autovacuum 和 vacuum 活动。
- 可以安装和配置 PostgreSQL Audit（pgAudit）扩展，以在会话级别或在对象级别捕获活动。有关更多信息，请参阅 使用 pgAudit 记录数据库活动。
- `log_fdw` 扩展使您可以使用 SQL 访问数据库引擎日志。有关更多信息，请参阅 使用 log_fdw 扩展通过 SQL 访问数据库日志。
- `pg_stat_statements` 库被指定为 RDS for PostgreSQL 版本 10 及更高版本中 `shared_preload_libraries` 参数的默认值。您可以使用该库分析正在运行的查询。确保在数据库参数组中设置 `pg_stat_statements`。有关使用此库提供的信息监控 RDS for PostgreSQL 数据库实例的更多信息，请参阅 RDS PostgreSQL 的 SQL 统计数据。
- `log_hostname` 参数将每个客户端连接的主机名捕获到日志中。对于 RDS for PostgreSQL 版本 12 及更高版本，原定设置情况下，此参数设置为 `off`。如果您将其开启，请务必监视会话连接时间。开启后，该服务使用域名系统（DNS）反向查找请求来获取进行连接的客户端的主机名并将其添加到 PostgreSQL 日志中。这在会话连接期间会产生明显的影响。我们建议您仅出于故障排除目的开启此参数。

一般来说，日志记录的目的是使数据库管理员能够监控、优化性能和进行故障排除。许多日志都会自动上载到 Amazon CloudWatch 或 Performance Insights。在这里，对这些日志进行排序和分组，从而为您的数据库实例提供完整指标。要了解有关 Amazon RDS 监控和指标的更多信息，请参阅 监控 Amazon RDS 实例中的指标。

使用 PostgreSQL 管理临时文件

在 PostgreSQL 中，一个复杂的查询可能会同时执行几个排序或哈希操作，每个操作都使用实例内存来存储结果，直至达到在 `work_mem` 参数中指定的值。当实例内存不足时，会创建临时文件来存储结果。这些文件写入磁盘以完成查询执行。稍后，将在查询完成后自动删除这些文件。在 RDS for PostgreSQL 中，这些文件存储在数据卷上的 Amazon EBS 中。有关更多信息，请参阅 Amazon RDS 数据库实例存储。您可以监控 CloudWatch 内发布的 `FreeStorageSpace` 指标，以确保数据库实例具有足够的可用存储空间。有关更多信息，请参阅 `FreeStorageSpace` 。

对于涉及多个并发查询的工作负载（这会增加临时文件的使用量），我们建议使用 Amazon RDS 优化型读取实例。这些实例使用基于本地非易失性存储规范（NVMe）的固态硬盘（SSD）块级存储来存放临时文件。有关更多信息，请参阅 使用 Amazon RDS 优化型读取功能提高 RDS for PostgreSQL 的查询性能。

您可以使用下面的参数和函数来管理实例中的临时文件。

- **`temp_file_limit`** – 此参数取消任何超过 temp_files 大小（以 KB 为单位）的查询。此限制可防止任何查询无休止地运行并使用临时文件消耗磁盘空间。您可以使用来自 `log_temp_files` 参数的结果来估计该值。作为最佳实践，请检查工作负载行为并根据估计值设置限制。以下示例显示了当查询超过限制时如何取消查询。

```
postgres=>select * from pgbench_accounts, pg_class, big_table;
```

```

ERROR: temporary file size exceeds temp_file_limit (64kB)
```

- **`log_temp_files`** – 当删除会话的临时文件时，此参数会向 postgresql.log 发送消息。此参数在查询成功完成后生成日志。因此，它可能无助于对长时间运行的活跃查询进行故障排除。

  以下示例显示，当查询成功完成后，条目将记录在 postgresql.log 文件中，同时清理临时文件。

```

2023-02-06 23:48:35 UTC:205.251.233.182(12456):adminuser@postgres:[31236]:LOG:
 temporary file: path "base/pgsql_tmp/pgsql_tmp31236.5", size 140353536
2023-02-06 23:48:35 UTC:205.251.233.182(12456):adminuser@postgres:[31236]:STATEMENT:
  select a.aid from pgbench_accounts a, pgbench_accounts b where a.bid=b.bid order by
 a.bid limit 10;
2023-02-06 23:48:35 UTC:205.251.233.182(12456):adminuser@postgres:[31236]:LOG:
 temporary file: path "base/pgsql_tmp/pgsql_tmp31236.4", size 180428800
```

```
2023-02-06 23:48:35 UTC:205.251.233.182(12456):adminuser@postgres:[31236]:STATEMENT:
  select a.aid from pgbench_accounts a, pgbench_accounts b where a.bid=b.bid order by
 a.bid limit 10;
```

- **pg_ls_tmpdir** – 此函数在 RDS for PostgreSQL 13 及更高版本中提供，可让用户了解当前临时文件使用情况。完成的查询不会出现在该函数的结果中。在以下示例中，您可以查看此函数的结果。

```
postgres=>select * from pg_ls_tmpdir();
```

```
      name       |    size    |      modification
-----------------+------------+------------------------
 pgsql_tmp8355.1 | 1072250880 | 2023-02-06 22:54:56+00
 pgsql_tmp8351.0 | 1072250880 | 2023-02-06 22:54:43+00
 pgsql_tmp8327.0 | 1072250880 | 2023-02-06 22:54:56+00
 pgsql_tmp8351.1 |  703168512 | 2023-02-06 22:54:56+00
 pgsql_tmp8355.0 | 1072250880 | 2023-02-06 22:54:00+00
 pgsql_tmp8328.1 |  835031040 | 2023-02-06 22:54:56+00
 pgsql_tmp8328.0 | 1072250880 | 2023-02-06 22:54:40+00
(7 rows)
```

```
postgres=>select query from pg_stat_activity where pid = 8355;

query
----------------------------------------------------------------------------------------
select a.aid from pgbench_accounts a, pgbench_accounts b where a.bid=b.bid order by
 a.bid
(1 row)
```

文件名包括生成了临时文件的会话的处理 ID（PID）。更高级的查询（如以下示例所示）对每个 PID 的临时文件执行总和。

```
postgres=>select replace(left(name, strpos(name, '.')-1),'pgsql_tmp','') as pid,
 count(*), sum(size) from pg_ls_tmpdir() group by pid;
```

```
 pid   | count |   sum
```

```
------+-------------------
 8355 |     2 | 2144501760
 8351 |     2 | 2090770432
 8327 |     1 | 1072250880
 8328 |     2 | 2144501760
(4 rows)
```

- **pg_stat_statements** – 如果您激活 pg_stat_sat_statements 参数，则可以查看每个调用的平均临时文件使用量。您可以识别查询的 query_id 并使用它来检查临时文件使用情况，如以下示例所示。

```
postgres=>select queryid from pg_stat_statements where query like 'select a.aid from
 pgbench%';
```

```
       queryid
----------------------
 -7170349228837045701
(1 row)
```

```
postgres=>select queryid, substr(query,1,25), calls, temp_blks_read/calls
 temp_blks_read_per_call, temp_blks_written/calls temp_blks_written_per_call from
 pg_stat_statements where queryid = -7170349228837045701;
```

```
       queryid        |          substr           | calls | temp_blks_read_per_call |
 temp_blks_written_per_call
----------------------+---------------------------+-------+-------------------------
+----------------------------
 -7170349228837045701 | select a.aid from pgbench |    50 |                  239226 |
                    388678
(1 row)
```

- **Performance Insights** – 在性能详情控制面板中，可以通过开启指标 temp_bytes 和 temp_files 来查看临时文件使用情况。然后，您可以看到这两个指标的平均值，并查看它们与查询工作负载的对应关系。性能详情中的视图并未专门显示正在生成临时文件的查询。但是，当您将性能详情与针对 `pg_ls_tmpdir` 显示的查询相结合时，您可以排查、分析并确定查询工作负载的变化。

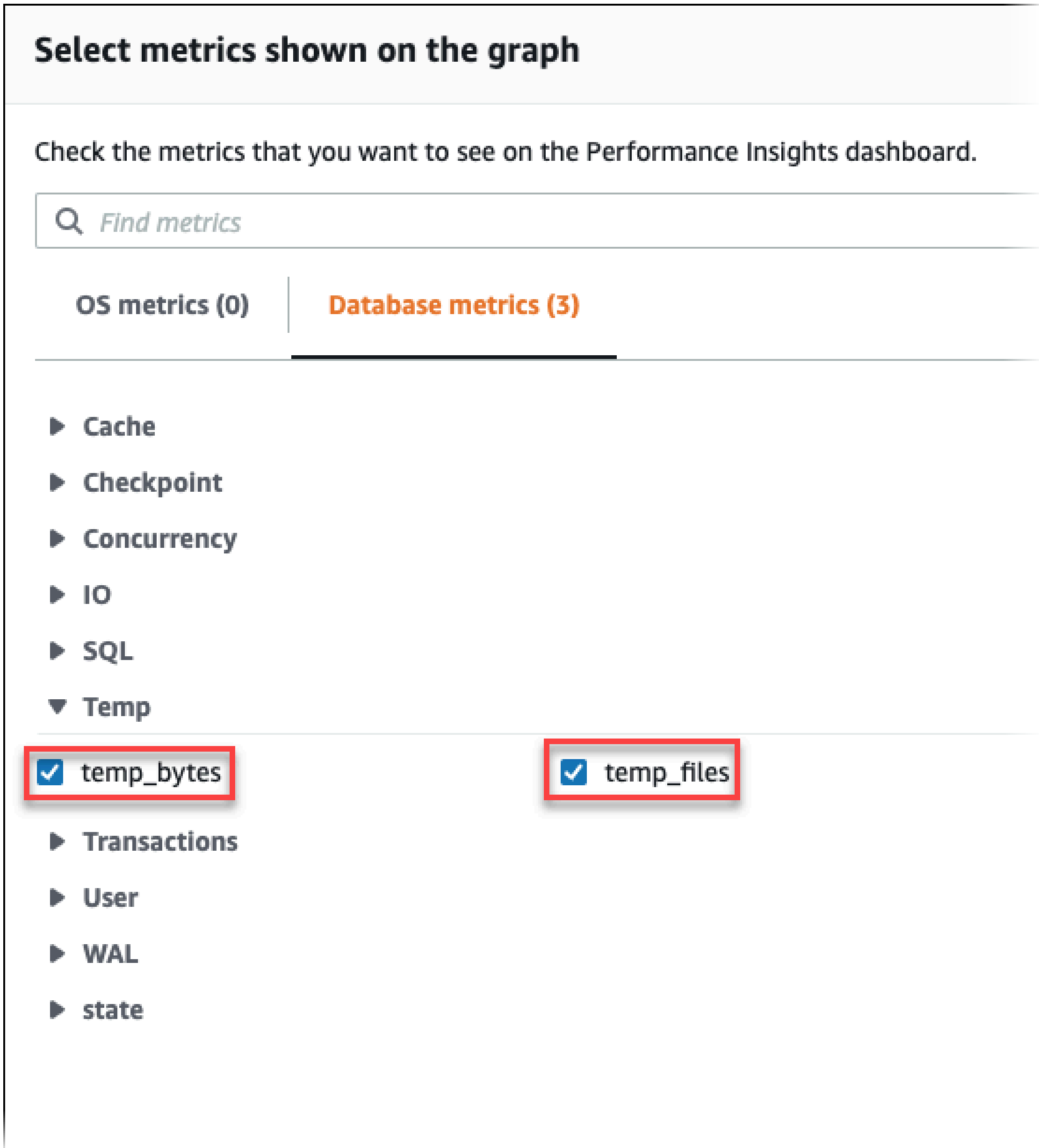
有关如何使用性能详情分析指标和查询的更多信息，请参阅使用 Performance Insights 控制面板分析指标。

有关使用性能详情查看临时文件使用情况的示例，请参阅使用性能详情查看临时文件使用情况

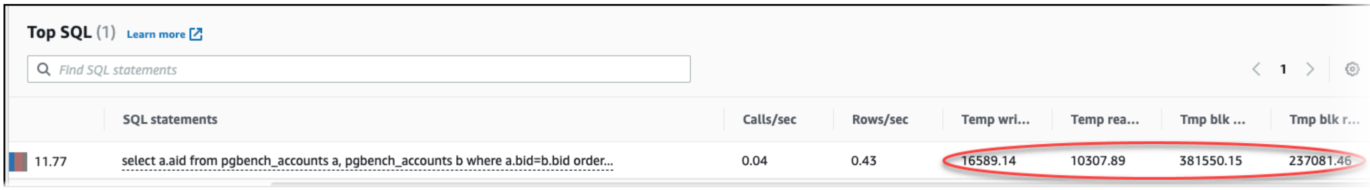
使用性能详情查看临时文件使用情况

可以使用性能详情，通过开启指标 temp_bytes 和 temp_files 来查看临时文件使用情况。性能详情中的视图不显示生成临时文件的特定查询，但是，当您将性能详情与针对 `pg_ls_tmpdir` 显示的查询相结合时，可以排查、分析并确定查询工作负载的变化。

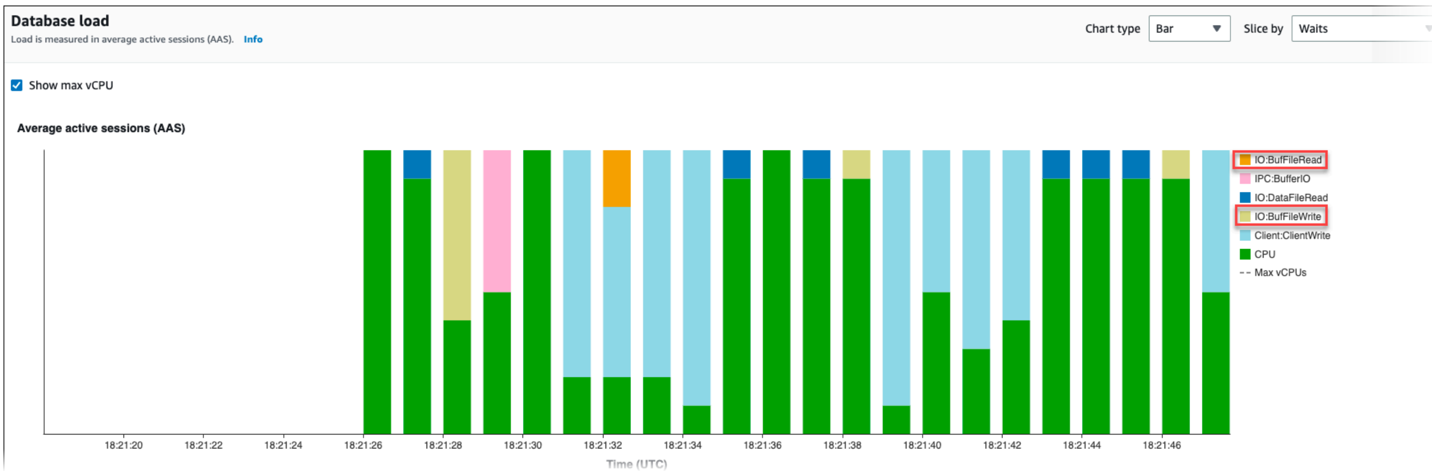
1. 在性能详情控制面板中，选择管理指标。
2. 选择数据库指标，然后选择 temp_bytes 和 temp_files 指标，如下图所示。



3. 在 Top SQL 选项卡中，选择首选项图标。
4. 在首选项窗口中，打开 Top SQL 选项卡中显示的以下统计数据，然后选择继续。
   - 临时写入次数/秒
   - 临时读取次数/秒
   - 临时批量写入/调用
   - 临时批量读取/调用
5. 当临时文件与针对 `pg_ls_tmpdir` 显示的查询相组合时，临时文件将被分解，如以下示例所示。



当您的工作负载中排名靠前的查询经常创建临时文件时，就会发生 `IO:BufFileRead` 和 `IO:BufFileWrite` 事件。通过查看“数据库负载”和“热门 SQL”部分中的平均活动会话（AAS），您可以使用性能详情来确定在 `IO:BufFileRead` 和 `IO:BufFileWrite` 上等待的热门 SQL。



有关如何使用性能详情按等待事件分析热门查询和负载的更多信息，请参阅“Top SQL”（主要 SQL）选项卡概览。您应该识别和调整导致临时文件使用量和相关等待事件增加的查询。有关这些等待事件和补救措施的更多信息，请参阅 IO:BufFileRead 和 IO:BufFileWrite。

> **Note**
>
> `work_mem` 参数控制排序操作何时耗尽内存以及结果何时写入临时文件中。我们建议您不要将此参数的设置更改为高于默认值，因为这将允许每个数据库会话消耗更多内存。此外，执行复杂联接和排序的单个会话可以执行并行操作，其中每个操作都会消耗内存。
>
> 作为最佳实践，当您有一个包含多个联接和排序的大型报告时，请使用 `SET work_mem` 命令在会话级别设置此参数。然后，更改仅应用于当前会话，而不会全局更改该值。

将用于日志分析的 pgBadger 与 PostgreSQL 结合使用

您可以使用日志分析器（如 pgBadger）分析 PostgreSQL 日志。pgBadger 文档声明 %l 模式（会话或进程的日志行）应为前缀的一部分。但如果将当前 RDS `log_line_prefix` 作为参数提供给 pgBadger，它应该仍生成报告。

例如，以下命令使用 pgBadger 正确设置日期为 2014-02-04 的 Amazon RDS for PostgreSQL 日志文件的格式。

```
./pgbadger -f stderr -p '%t:%r:%u@%d:[%p]:' postgresql.log.2014-02-04-00
```

使用 PGSnapper 监控 PostgreSQL

可以使用 PGSnapper 协助定期收集 Amazon RDS for PostgreSQL 性能相关的统计数据和指标。有关更多信息，请参阅使用 PGSnapper 监控 Amazon RDS for PostgreSQL 性能。

在 RDS for PostgreSQL 数据库实例上使用参数

在某些情况下，您可以在不指定自定义参数组的情况下创建 RDS for PostgreSQL 数据库实例。如果是这样，您的数据库实例将使用您选择的 PostgreSQL 版本的默认参数组创建。例如，假设您使用 PostgreSQL 13.3 创建 RDS for PostgreSQL 数据库实例。在这种情况下，数据库实例使用 PostgreSQL 13 版本 `default.postgres13` 的参数组中的值创建。

您还可以创建自己的自定义数据库参数组。如果要从其默认值修改 RDS for PostgreSQL 数据库实例的任何设置，则需要执行此操作。要了解如何操作，请参阅Amazon RDS 的参数组。

您可以通过几种不同的方法跟踪 RDS for PostgreSQL 数据库实例的设置。您可以使用 Amazon Web Services Management Console、Amazon CLI 或 Amazon RDS API。也可以从实例的 PostgreSQL `pg_settings` 表中查询值，如下所示。

```
SELECT name, setting, boot_val, reset_val, unit
 FROM pg_settings
 ORDER BY name;
```

要了解此查询所返回值的更多相关信息，请参阅 PostgreSQL 文档中的 `pg_settings`。

更改 RDS for PostgreSQL 数据库实例上的 `max_connections` 和 `shared_buffers` 的设置时要特别小心。例如，假设您修改 `max_connections` 或 `shared_buffers` 的设置，并且您使用的值对于实际工作负载来说太高。在这种情况下，RDS for PostgreSQL 数据库实例将无法启动。如果发生这种情况，您会在 `postgres.log` 中看到如下错误。

```
2018-09-18 21:13:15 UTC::@:[8097]:FATAL:  could not map anonymous shared memory: Cannot
 allocate memory
2018-09-18 21:13:15 UTC::@:[8097]:HINT:  This error usually means that PostgreSQL's
 request for a shared memory segment
```

```
exceeded available memory or swap space. To reduce the request size (currently
 3514134274048 bytes), reduce
PostgreSQL's shared memory usage, perhaps by reducing shared_buffers or
 max_connections.
```

但是，您无法更改默认 RDS for PostgreSQL 数据库参数组中包含的任何设置值。要更改任何参数的设置，请首先创建自定义数据库参数组。然后，更改该自定义组中的设置，并将自定义参数组应用到 RDS for PostgreSQL 数据库实例。要了解更多信息，请参阅Amazon RDS 的参数组。

RDS for PostgreSQL 中有两种类型的参数。

- 静态参数：静态参数需要在更改后重启 RDS for PostgreSQL 数据库实例，以便新值生效。
- 动态参数：动态参数在更改设置后不需要重启。

> **Note**
>
> 如果 RDS for PostgreSQL 数据库实例使用您自己的自定义数据库参数组，则可以更改正在运行的数据库实例上动态参数的值。您可以通过使用 Amazon Web Services Management Console、Amazon CLI 或 Amazon RDS API 实现此目的。

如果您有这样做的权限，则还可使用 `ALTER DATABASE`、`ALTER ROLE` 和 `SET` 命令更改参数值。

RDS for PostgreSQL 数据库实例参数列表

下表列出了 RDS for PostgreSQL 数据库实例中的一些（但并非全部）可用参数。要查看所有可用参数，请使用 Amazon CLI 命令 describe-db-parameters。例如，要获取 RDS for PostgreSQL 版本 13 的原定设置参数组中所有可用参数的列表，请运行以下命令。

```
aws rds describe-db-parameters --db-parameter-group-name default.postgres13
```

您还可以使用控制台。从 Amazon RDS 菜单中选择 Parameter groups（参数组），然后从您的 Amazon Web Services 区域中的可用参数组中选择参数组。

参数名称	应用类型	描述
application_name	动态	设置要在统计数据和日志中报告的应用程序名称。
archive_command	动态	设置将被调用以存档 WAL 文件的 Shell 命令。
array_nulls	动态	允许在阵列中输入 NULL 元素。
authentication_timeout	动态	设置允许完成客户端身份验证的最长时间。
autovacuum	动态	启动 Autovacuum 子进程。
autovacuum_analyze_scale_factor	动态	analyze 之前插入、更新或删除元组的次数，以对于 reltuple 的占比计。
autovacuum_analyze_threshold	动态	analyze 之前插入、更新或删除元组的最小次数。
autovacuum_freeze_max_age	静态	对表进行 Autovacuum 以防事务 ID 重叠的期限。
autovacuum_naptime	动态	两次 Autovacuum 运行之间的睡眠时间。
autovacuum_max_workers	静态	设置同时运行的 Autovacuum 工作者的最大数量。
autovacuum_vacuum_cost_delay	动态	Autovacuum 的真空开销延迟，以毫秒计。

参数名称	应用类型	描述
autovacuum_vacuum_cost_limit	动态	Autovacuum 在小睡之前可用的真空开销量。
autovacuum_vacuum_scale_factor	动态	vacuum 之前更新或删除元组的次数，以对于 reltuple 的占比计。
autovacuum_vacuum_threshold	动态	vacuum 之前更新或删除元组的最小次数。
backslash_quote	动态	设置字符串字面值中是否允许有反斜杠 (\)。
bgwriter_delay	动态	后台写入实例在两轮之间的睡眠时间。
bgwriter_lru_maxpages	动态	后台写入实例每轮要刷新的最大 LRU 页数。
bgwriter_lru_multiplier	动态	每轮要释放的平均缓冲区用量的倍数。
bytea_output	动态	设置字节的输出格式。
check_function_bodies	动态	在 CREATE FUNCTION 期间检查函数体。
checkpoint_completion_target	动态	在检查点期间刷新脏缓冲区所用的时间，以对于检查点间隔的占比计。
checkpoint_segments	动态	设置日志段中自动预写日志 (WAL) 检查点之间的最大距离。
checkpoint_timeout	动态	设置自动 WAL 检查点之间的最长时间。
checkpoint_warning	动态	如果填充检查点段的频率高于此，则启用警告。
client_connection_check_interval	动态	设置在运行查询时检查断开连接的时间间隔。
client_encoding	动态	设置客户端的字符集编码。

参数名称	应用类型	描述
client_min_messages	动态	设置发送到客户端的消息级别。
commit_delay	动态	设置事务提交与将 WAL 刷新到磁盘之间的延迟，以微秒计。
commit_siblings	动态	设置执行 commit_delay 之前同时打开的最少事务数。
constraint_exclusion	动态	使计划程序可使用约束优化查询。
cpu_index_tuple_cost	动态	设置计划程序对索引扫描期间处理每个索引条目的开销的估算。
cpu_operator_cost	动态	设置计划程序对处理每个运算符或函数调用的开销的估算。
cpu_tuple_cost	动态	设置计划程序对处理每个元组 (行) 的开销的估算。
cursor_tuple_fraction	动态	设置计划程序对光标将检索的行占比的估算。
datestyle	动态	设置日期和时间值的显示格式。
deadlock_timeout	动态	设置在检查死锁之前等待锁定的时间。
debug_pretty_print	动态	缩进分析树和计划树的显示内容。
debug_print_parse	动态	记录每个查询的分析树。
debug_print_plan	动态	记录每个查询的执行计划。
debug_print_rewritten	动态	记录每个查询重写的分析树。
default_statistics_target	动态	设置默认统计数据目标。
default_tablespace	动态	设置要从中创建表和索引的默认表空间。

参数名称	应用类型	描述
default_transaction_deferrable	动态	设置新事务的默认可延迟状态。
default_transaction_isolation	动态	设置每个新事务的事务隔离级别。
default_transaction_read_only	动态	设置新事务的默认只读状态。
default_with_oids	动态	默认情况下，使用对象 ID (OID) 创建新表。
effective_cache_size	动态	设置计划程序对于磁盘缓存大小的假设。
effective_io_concurrency	动态	磁盘子系统可有效处理的并行请求数。
enable_bitmapscan	动态	使计划程序可使用位图扫描计划。
enable_hashagg	动态	使计划程序可使用哈希聚合计划。
enable_hashjoin	动态	使计划程序可使用哈希联接计划。
enable_indexscan	动态	使计划程序可使用索引扫描计划。
enable_material	动态	使计划程序可使用具体化。
enable_mergejoin	动态	使计划程序可使用合并联接计划。
enable_nestloop	动态	使计划程序可使用嵌套循环的联接计划。
enable_seqscan	动态	使计划程序可使用顺序扫描计划。
enable_sort	动态	使计划程序可使用显式排序步骤。
enable_tidscan	动态	使计划程序可使用 TID 扫描计划。
escape_string_warning	动态	警告在普通字符串字面值中有反斜杠 (\) 转义符。

参数名称	应用类型	描述
extra_float_digits	动态	设置所显示的浮点值位数。
from_collapse_limit	动态	设置超出其即不折叠子查询的 FROM 列表大小。
fsync	动态	强制将更新同步到磁盘。
full_page_writes	动态	在检查点后首次修改时向 WAL 写入整页。
geqo	动态	启用基因查询优化。
geqo_effort	动态	GEQO：用于设置其他 GEQO 参数默认值的工作负载。
geqo_generations	动态	GEQO：算法的迭代次数。
geqo_pool_size	动态	GEQO：群体中的个体数。
geqo_seed	动态	GEQO：随机路径选择的种子。
geqo_selection_bias	动态	GEQO：群体中的选择性压力。
geqo_threshold	动态	设置超出其即使用 GEQO 的 FROM 项阈值。
gin_fuzzy_search_limit	动态	通过允许由 GIN 进行的精确搜索得出的最大结果数。
hot_standby_feedback	动态	确定热备用将反馈消息发送到主备用还是上游备用。
intervalstyle	动态	设置间隔值的显示格式。
join_collapse_limit	动态	设置超出其即不平展 JOIN 结构的 FROM 列表大小。
lc_messages	动态	设置显示消息的语言。
lc_monetary	动态	设置用于使货币金额格式化的区域设置。
lc_numeric	动态	设置用于使数字格式化的区域设置。

参数名称	应用类型	描述
lc_time	动态	设置用于使日期和时间值格式化的区域设置。
log_autovacuum_min_duration	动态	设置如超出即记录 Autovacuum 操作的最短运行时间。
log_checkpoints	动态	记录每个检查点。
log_connections	动态	记录每个成功的连接。
log_disconnections	动态	记录会话结束，包括持续时间。
log_duration	动态	记录每个完成的 SQL 语句的持续时间。
log_error_verbosity	动态	设置记录消息的详细程度。
log_executor_stats	动态	向服务器日志写入执行者性能统计数据。
log_filename	动态	设置日志文件的文件名模式。
log_file_mode	动态	设置日志文件的文件权限。默认值为 0644。
log_hostname	动态	在连接日志中记录主机名。从 PostgreSQL 12 及更高版本开始，原定设置情况下，此参数为“off”。开启后，连接使用 DNS 反向查找来获取捕获到连接日志的主机名。如果您启用此参数，则应监控它对建立连接所花费时间的影响。
log_line_prefix	动态	控制每个日志行前缀的信息。
log_lock_waits	动态	记录长锁定等待次数。
log_min_duration_statement	动态	设置如超出即记录语句的最短运行时间。
log_min_error_statement	动态	导致记录所有产生此水平或此水平之上错误的语句。
log_min_messages	动态	设置记录的消息级别。

参数名称	应用类型	描述
log_parser_stats	动态	向服务器日志写入分析器性能统计数据。
log_planner_stats	动态	向服务器日志写入计划程序性能统计数据。
log_rotation_age	动态	将在 N 分钟后进行日志文件自动轮换。
log_rotation_size	动态	将在 N KB 后进行日志文件自动轮换。
log_statement	动态	设置所记录的语句类型。
log_statement_stats	动态	向服务器日志写入累计性能统计数据。
log_temp_files	动态	记录对大于此 KB 数的临时文件的使用情况。
log_timezone	动态	设置要在日志消息中使用的时区。
log_truncate_on_rotation	动态	在日志轮换期间截断同名的现有日志文件。
logging_collector	静态	启动子进程将 stderr 输出和/或 csvlog 捕获到日志文件中。
maintenance_work_mem	动态	设置要用于维护操作的最大内存。
max_connections	静态	设置最大并行连接数。
max_files_per_process	静态	设置同时为每个服务器进程打开的最大文件数。
max_locks_per_transaction	静态	设置每个事务的最大锁定数。
max_pred_locks_per_transaction	静态	设置每个事务的最大谓词锁定数。
max_prepared_transactions	静态	设置同时准备的最大事务数。
max_stack_depth	动态	设置最大堆栈长度，以 KB 计。

参数名称	应用类型	描述
max_standby_archive_delay	动态	设置在有热备用服务器处理已存档的 WAL 数据时取消查询之前的最大延迟。
max_standby_streaming_delay	动态	设置在有热备用服务器处理流式 WAL 数据时取消查询之前的最大延迟。

max_wal_size	动态	设置触发检查点的 WAL 大小（MB）。
		• 对于 RDS for PostgreSQL 15 及更早版本，它为 2 GB。
		• 对于 RDS for PostgreSQL 16 及更高版本，RDS for PostgreSQL 会根据您的实例分配的存储大小自动进行配置：
		  • 对于分配的存储空间为 100 GB 及以上的实例，为 6 GB。
		  • 对于分配的存储空间小于 100 GB 的实例，为 2 GB。
		对 Amazon RDS for PostgreSQL 数据库实例使用以下命令可查看其当前值：
		`SHOW max_wal_size;`

min_wal_size	动态	设置将 WAL 缩小到的最小大小。对于 PostgreSQL 9.6 版及更早版本，`min_wal_size` 以 16 MB 为单位。对于 PostgreSQL 10 版及更高版本，`min_wal_size` 以 1 MB 为单位。
quote_all_identifiers	动态	在生成 SQL 片段时向所有标识符添加引号 (")。
random_page_cost	动态	设置计划程序对非连续提取磁盘页面的开销的估算。除非开启查询计划管理（QPM），否则此参数没有值。开启 QPM 时，此参数的默认值为 4。

参数名称	应用类型	描述
rds.adaptive_autovacuum	动态	在超过事务 ID 阈值时自动优化 Autovacuum 参数。
rds.force_ssl	动态	要求使用 SSL 连接。对于 RDS for PostgreSQL 版本 15，默认值设为 1（开启）。所有其他 RDS for PostgreSQL 主要版本 14 及更早版本的默认值都设为 0（关闭）。
rds.local_volume_spill_enabled	静态	允许将逻辑溢出文件写入本地卷。
rds.log_retention_period	动态	设置日志保留，以便 Amazon RDS 删除超过 n 分钟的 PostgreSQL 日志。
rds.rds_superuser_reserved_connections	静态	设置为 rds_superuser 保留的连接插槽数。此参数仅在版本 15 及更高版本中提供。有关更多信息，请参阅 PostgreSQL 文档 reserved_connections。
rds.replica_identity_full	动态	当您将此参数设置为 on 时，它会将所有数据库表的副本标识设置覆盖为 `FULL`。这意味着，无论您的 `REPLICA IDENTITY FULL` 设置如何，所有列值都将写入预写日志（WAL）。 **Note** 由于额外的 WAL 日志记录，开启此参数可能会增加数据库实例 IOPS。
rds.restrict_password_commands	静态	将可以管理密码的用户限制为具有 `rds_password` 角色的用户。将此参数设置为 1 可启用密码限制。默认值为 0。
search_path	动态	设置针对非架构限定名称的架构搜索顺序。
seq_page_cost	动态	设置计划程序对连续提取磁盘页面的开销的估算。

参数名称	应用类型	描述
session_replication_role	动态	设置触发器和重写规则的会话行为。
shared_buffers	静态	设置服务器使用的共享内存缓冲区数。
shared_preload_libraries	静态	列出要预加载到 RDS for PostgreSQL 数据库实例中的共享库。支持的值包括 auto_explain、orafce、pgaudit、pglogical、pg_bigm、pg_cron、pg_hint_plan、pg_prewarm、pg_similarity、pg_stat_statements、pg_tle、pg_transport、plprofiler 和 plrust。
ssl	动态	启用 SSL 连接。
sql_inheritance	动态	导致在各种命令中默认加入子表。
ssl_renegotiation_limit	动态	设置在重新协商加密密钥之前发送和接收的流量。
standard_conforming_strings	动态	导致 ... 字符串按字面处理反斜杠。
statement_timeout	动态	设置允许任何语句的最长持续时间。
synchronize_seqscans	动态	启用同步顺序扫描。
synchronous_commit	动态	设置当前事务同步级别。
tcp_keepalives_count	动态	重新传输 TCP 保持连接信号的最大次数。
tcp_keepalives_idle	动态	发出两次 TCP 保持连接信号之间的时间。
tcp_keepalives_interval	动态	两次 TCP 保持连接信号重新传输之间的时间。
temp_buffers	动态	设置每个会话使用的临时缓冲区的最大数量。

参数名称	应用类型	描述
temp_file_limit	动态	设置临时文件可以达到的最大大小，以 KB 为单位。
temp_tablespaces	动态	选择用于临时表和排序文件的表空间。
timezone	动态	设置用于显示和解译时间戳的时区。 互联网编号分配机构（IANA）每年多次在 https://www.iana.org/time-zones 上发布新的时区。每当 RDS 发布 PostgreSQL 的新次要维护版本时，RDS 都会附带发布时的最新时区数据。当您使用最新的 RDS for PostgreSQL 版本时，您会获得来自 RDS 的最新时区数据。为确保您的数据库实例具有最新的时区数据，我们建议升级到更高的数据库引擎版本。您无法手动修改 PostgreSQL 数据库实例中的时区表。RDS 不会修改或重置正在运行的数据库实例的时区数据。只有在执行数据库引擎版本升级时，才会安装新的时区数据。
track_activities	动态	收集有关运行命令的信息。
track_activity_query_size	静态	设置为 pg_stat_activity.current_query 保留的大小，以字节计。
track_counts	动态	收集有关数据库活动的统计数据。
track_functions	动态	收集有关数据库活动的函数级别统计数据。
track_io_timing	动态	收集有关数据库活动的时序统计数据。
transaction_deferrable	动态	指示是否将某个只读可序列化事务推迟到启动它不会发生序列化失败时。
transaction_isolation	动态	设置当前事务隔离级别。

参数名称	应用类型	描述
transaction_read_only	动态	设置当前事务只读状态。
transform_null_equals	动态	将 expr=NULL 视为 expr IS NULL。
update_process_title	动态	更新进程标题以显示活动的 SQL 命令。
vacuum_cost_delay	动态	真空开销延迟，以毫秒计。
vacuum_cost_limit	动态	小睡之前可用的真空开销量。
vacuum_cost_page_dirty	动态	由真空弄脏的页面的真空开销。
vacuum_cost_page_hit	动态	在缓冲区缓存中找到的页面的真空开销。
vacuum_cost_page_miss	动态	在缓冲区缓存中未找到的页面的真空开销。
vacuum_defer_cleanup_age	动态	真空和热清理应推迟的事务数 (如果有)。
vacuum_freeze_min_age	动态	真空应冻结表格行的最短期限。
vacuum_freeze_table_age	动态	真空应扫描整个表以冻结元组的期限。
wal_buffers	静态	设置 WAL 的共享内存中的磁盘页面缓冲区数。
wal_writer_delay	动态	两次 WAL 刷新之间的 WAL 写入实例睡眠时间。
work_mem	动态	设置要用于查询工作区的最大内存。
xmlbinary	动态	设置如何将二进制值编码到 XML 中。

参数名称	应用类型	描述
xmloption	动态	设置要将隐式分析和序列化操作中的 XML 数据视为文档还是内容片段。

Amazon RDS 对所有参数均使用默认的 PostgreSQL 单位。下表显示每个参数的 PostgreSQL 原定设置单位。

参数名称	单位
archive_timeout	s
authentication_timeout	s
autovacuum_naptime	s
autovacuum_vacuum_cost_delay	ms
bgwriter_delay	ms
checkpoint_timeout	s
checkpoint_warning	s
deadlock_timeout	ms
effective_cache_size	8 KB
lock_timeout	ms
log_autovacuum_min_duration	ms
log_min_duration_statement	ms
log_rotation_age	分钟
log_rotation_size	KB
log_temp_files	KB

参数名称	单位
maintenance_work_mem	KB
max_stack_depth	KB
max_standby_archive_delay	ms
max_standby_streaming_delay	ms
post_auth_delay	s
pre_auth_delay	s
segment_size	8 KB
shared_buffers	8 KB
statement_timeout	ms
ssl_renegotiation_limit	KB
tcp_keepalives_idle	s
tcp_keepalives_interval	s
temp_file_limit	KB
work_mem	KB
temp_buffers	8 KB
vacuum_cost_delay	ms
wal_buffers	8 KB
wal_receiver_timeout	ms
wal_segment_size	B
wal_sender_timeout	ms

参数名称	单位
wal_writer_delay	ms
wal_receiver_status_interval	s

使用 RDS for PostgreSQL 的等待事件进行优化

等待事件是 RDS for PostgreSQL 的重要优化工具。当您能查明会话为什么在等待资源以及会话在做什么时，您就能更好地减少瓶颈。您可以使用本节中的信息来查找可能的原因和纠正措施。本节还讨论基本的 PostgreSQL 优化概念。

本节中的等待事件特定于 RDS for PostgreSQL。

主题

RDS for PostgreSQL 优化的基本概念

在优化 RDS for PostgreSQL 数据库之前，请务必了解什么是等待事件以及它们发生的原因。还可以查看 RDS for PostgreSQL 的基本内存和磁盘架构。有关有用的架构图，请参阅 PostgreSQL wikibook。

主题

RDS for PostgreSQL 等待事件

等待事件表示会话正在等待资源。例如，当 RDS for PostgreSQL 等待从客户端接收数据时，会发生等待事件 `Client:ClientRead`。会话通常会等待如下资源。

- 例如，当会话试图修改缓冲区时，对缓冲区的单线程访问
- 当前被另一个会话锁定的行
- 已读取一个数据文件
- 已写入一个日志文件

例如，为了满足查询，会话可能会执行完整的表扫描。如果数据尚未在内存中，会话将等待磁盘输入/输出完成。当缓冲区读取到内存时，会话可能需要等待，因为其他会话正在访问相同的缓冲区。数据库使用预定义的等待事件记录等待。这些事件按类别进行分组。

等待事件本身并不表示存在性能问题。例如，如果请求的数据不在内存中，则必须从磁盘读取数据。如果一个会话锁定行以进行更新，则另一个会话将等待解锁该行，以便它可以更新该行。提交需要等待对日志文件的写入完成。等待是数据库正常运行不可或缺的组成部分。

另一方面，大量的等待事件通常表示存在性能问题。在这种情况下，您可以使用等待事件数据来确定会话将时间花费在哪里。例如，如果通常在几分钟内运行的报告现在运行需要几个小时，则可以确定对总等待时间贡献最大的等待事件。如果您能确定顶级等待事件的原因，您有时就可以进行更改来提高性能。例如，如果您的会话正在等待已被另一个会话锁定的行，则可以结束锁定会话。

RDS for PostgreSQL 内存

RDS for PostgreSQL 内存分为共享内存和本地内存。

主题

- [RDS for PostgreSQL 中的共享内存]
- [RDS for PostgreSQL 中的本地内存]

RDS for PostgreSQL 中的共享内存

RDS for PostgreSQL 会在实例启动时分配共享内存。共享内存分为多个子区域。以下各节提供了最重要子区域的说明。

主题

- [共享缓冲区]
- [预写日志 (WAL) 缓冲区]

共享缓冲区

共享缓冲池是一个 RDS for PostgreSQL 内存区域，它包含应用程序连接现在正在使用或过去使用的所有页面。分页是磁盘数据块的内存版本。共享缓冲池缓存从磁盘读取的数据块。该缓冲池减少了从磁盘重新读取数据的需求，从而提高了数据库的运行效率。

每个表和索引都存储为固定大小的页面数组。每个数据块包含多个元组，它们与行相对应。元组可以存储在任何页面中。

共享缓冲池的内存有限。如果新请求需要一个不在内存中的页面，并且没有更多的内存，RDS for PostgreSQL 会移除一个较少使用的页面来容纳请求。移出策略通过时钟扫描算法来实现。

`shared_buffers` 参数确定服务器用于缓存数据的内存量。根据数据库实例的可用内存，默认值设置为 `{DBInstanceClassMemory/32768}` 字节。

预写日志 (WAL) 缓冲区

预写日志（WAL）缓冲区保存 RDS for PostgreSQL 稍后写入持久存储的事务数据。使用 WAL 机制，RDS for PostgreSQL 可以执行以下操作：

- 发生故障后恢复数据
- 通过避免频繁写入磁盘来减少磁盘输入/输出

当客户端更改数据时，RDS for PostgreSQL 会将更改写入 WAL 缓冲区。当客户发出 `COMMIT`，WAL 写入器进程将事务数据写入 WAL 文件。

`wal_level` 参数决定向 WAL 写入的信息量，可能的值包括 `minimal`、`replica` 和 `logical` 等。

RDS for PostgreSQL 中的本地内存

每个后端进程都会为查询处理分配本地内存。

主题

- 工作内存区域
- 维护工作内存区域
- 临时缓冲区

工作内存区域

工作内存区域保存执行排序和哈希的查询的临时数据。例如，包含 `ORDER BY` 子句的查询执行排序。查询在哈希联接和聚合中使用哈希表。

`work_mem` 参数表示在写入临时磁盘文件之前，内部排序操作和哈希表要使用的内存量，以 MB 为单位。原定设置值为 4MB。可以同时运行多个会话，且每个会话可以并行运行维护操作。出于这个原因，使用的总工作内存可以是 `work_mem` 设置的倍数。

维护工作内存区域

维护工作内存区域缓存数据以进行维护操作。这些操作包括 vacuum 操作、创建索引和添加外键。

`maintenance_work_mem` 参数指定维护操作要使用的最大内存量，以 MB 为单位。原定设置值为 64MB。一个数据库会话一次只能运行一个维护操作。

临时缓冲区

临时缓冲区缓存每个数据库会话的临时表。

每个会话都根据需要分配临时缓冲区，但不超过您指定的限制。当会话结束时，服务器将清除缓冲区。

`temp_buffers` 参数设置每个会话使用的临时缓冲区的最大数量，以 MB 为单位。默认值为 8 MB。在会话中首次使用临时表之前，您可以更改 `temp_buffers` 值。

RDS for PostgreSQL 进程

RDS for PostgreSQL 使用多个进程。

主题

- 邮件管理员过程
- 后端进程

- 后台进程

邮件管理员过程

邮件管理员进程是启动 RDS for PostgreSQL 时开始的第一个进程。邮件管理员过程负有以下主要责任：

- 分流并监控后台进程
- 接收来自客户端进程的身份验证请求，并在允许数据库为请求提供服务之前对这些请求进行身份验证

后端进程

如果邮件管理员对客户户请求进行身份验证，邮件管理员会分流一个新的后端进程，也称为 postgres 进程。一个客户端进程只连接到一个后端进程。客户端进程和后端进程直接通信，而无需邮件管理员过程的干预。

后台进程

邮件管理员过程会分流执行不同后端任务的几个进程。其中一些更重要的事项包括：

- WAL 写入器

  RDS for PostgreSQL 会将 WAL（预写日志记录）缓冲区中的数据写入日志文件。预写日志记录的原则是，在数据库将描述这些更改的日志记录写入磁盘之后，数据库才能将更改写入数据文件。WAL 机制减少了磁盘 I/O，并允许 RDS for PostgreSQL 在出现故障后使用日志恢复数据库。
- 后台写入器

  此进程会定期将内存缓冲区中的脏（已修改）分页写入数据文件。当后端进程在内存中修改分页时，分页会变脏。
- Autovacuum 守护进程

  守护进程由以下各项组成：

  - Autovacuum 启动程序
  - Autovacuum 工件进程

  当 Autovacuum 开启时，它会检查包含大量插入的、更新的或删除的元组的表。守护进程要承担以下责任：

  - 恢复或重复使用更新或删除的行占用的磁盘空间

- 更新计划人员使用的统计数据
- 防止因事务 ID 重叠而导致旧数据丢失

Autovacuum 功能自动执行 `VACUUM` 和 `ANALYZE` 命令。`VACUUM` 具有以下变体：标准和完整版。标准 vacuum 与其他数据库操作并行运行。`VACUUM FULL` 需要对您工作所在的表具有专有锁定。因此，它不能与访问同一表的操作并行运行。`VACUUM` 创建了大量的输入/输出流量，这可能会导致其他活动会话的性能不佳。

RDS for PostgreSQL 等待事件

下表列出了 RDS for PostgreSQL 最常指示性能问题的等待事件，并总结了最常见的原因和纠正措施。

等待事件	定义
Client:ClientRead	当 RDS for PostgreSQL 等待从客户端接收数据时，会发生此事件。
Client:ClientWrite	当 RDS for PostgreSQL 等待将数据写入客户端时，会发生此事件。
CPU	当线程在 CPU 中处于活动状态或正在等待 CPU 时，会发生此事件。
IO:BufFileRead 和 IO:BufFileWrite	这些事件发生在 RDS for PostgreSQL 创建临时文件时。
IO:DataFileRead	当由于分页在共享内存中不可用，连接等待后端进程从存储中读取所需分页时，会发生此事件。
IO:WALWrite	当 RDS for PostgreSQL 正在等待将预写日志（WAL）缓冲区写入 WAL 文件时，会发生此事件。
Lock:advisory	当 PostgreSQL 应用程序使用锁定来协调多个会话之间的活动时，会发生此事件。

等待事件	定义
Lock:extend	当后端进程正在等待锁定关系以对其进行扩展，而另一个进程出于同样目的锁定该关系时，会发生此事件。
Lock:Relation	当查询等待获取当前被另一个事务锁定的表或视图上的锁定时，会发生此事件。
Lock:transactionid	当事务正在等待行级锁定时，会发生此事件。
Lock:tuple	在后端进程等待获取元组锁定时，会发生此事件。
LWLock:BufferMapping (LWLock:buffer_mapping)	当会话正在等待将数据块与共享缓冲池中的缓冲区关联起来时，会发生此事件。
LWLock:BufferIO (IPC:BufferIO)	当 RDS for PostgreSQL 正在等待其他进程在同时尝试访问页面时完成输入/输出（I/O）操作时，会发生此事件。
LWLock:buffer_content (BufferContent)	当某个会话等待读取或写入内存中的某个数据页面，而另一个会话正锁定该页面以进行写入时，会发生此事件。
LWLock:lock_manager (LWLock:lockmanager)	当 RDS for PostgreSQL 引擎维护共享锁的内存区域以便在无法使用快速路径锁时分配、检查和取消分配锁时，会发生此事件。
Timeout:PgSleep	当服务器进程调用 `pg_sleep` 函数并且正在等待睡眠超时过期时，会发生此事件。
Timeout:VacuumDelay	此事件表明 vacuum 进程正处于休眠状态，因为已达到估计的成本上限。

Client:ClientRead

当 RDS for PostgreSQL 等待从客户端接收数据时，会发生 `Client:ClientRead` 事件。

主题

支持的引擎版本

RDS for PostgreSQL 版本 10 及更高版本支持此等待事件信息。

上下文

RDS for PostgreSQL 数据库实例正在等待从客户端接收数据。RDS for PostgreSQL 数据库实例必须先从客户端接收数据，然后才能向客户端发送更多数据。实例在从客户端接收数据之前等待的时间为 `Client:ClientRead` 事件。

等待次数增加的可能原因

`Client:ClientRead` 显示在主要等待中的常见原因包括以下各项：

网络延迟增加

RDS for PostgreSQL 数据库实例和客户端之间的网络延迟可能会增加。较高的网络延迟会增加数据库实例从客户端接收数据所需的时间。

客户端负载增加

客户端上可能存在 CPU 压力或网络饱和。客户端负载的增加可能会延迟从客户端向 RDS for PostgreSQL 数据库实例传输数据的时间。

过多的网络往返次数

RDS for PostgreSQL 数据库实例和客户端之间的大量网络往返可能会延迟将数据从客户端传输到 RDS for PostgreSQL 数据库实例的过程。

大型复制操作

在复制操作期间，数据将从客户端的文件系统传输到 RDS for PostgreSQL 数据库实例。向数据库实例发送大量数据可能会延迟从客户端向数据库实例传输数据的时间。

空闲客户端连接

当客户端以 `idle in transaction` 状态连接到 RDS for PostgreSQL 数据库实例时，数据库实例可能会等待客户端发送更多数据或发出命令。在这种状态下的连接可能会导致 `Client:ClientRead` 事件增加。

用于连接池的 PgBouncer

PgBouncer 有一个名为 `pkt_buf` 的低级网络配置设置，预设情况下设置为 4096。如果工作负载通过 PgBouncer 发送大于 4096 字节的查询数据包，我们建议增加 `pkt_buf` 设置为 8192。如果新设置没有减少 `Client:ClientRead` 事件的数量，我们建议增加 `pkt_buf` 设置为较大的值，例如 16384 或 32768。如果查询文本很大，则较大的设置可能会特别有用。

操作

根据等待事件的原因，我们建议采取不同的操作。

主题

将客户端放置在与实例相同的可用区和 VPC 子网中

为了减少网络延迟并提高网络吞吐量，请将客户端放在与 RDS for PostgreSQL 数据库实例相同的可用区和虚拟私有云（VPC）子网中。确保客户端在地理位置上尽可能靠近数据库实例。

扩展客户端

使用 Amazon CloudWatch 或其他主机指标，确定您的客户端当前是受 CPU 或网络带宽的限制，还是受此两者的限制。如果客户端受到限制，请相应地扩展您的客户端。

使用当前一代实例

在某些情况下，您可能没有使用支持巨型帧的数据库实例类。如果您在 Amazon EC2 上运行应用程序，请考虑为客户端使用当前一代实例。另外，在客户端操作系统上配置最大传输单位 (MTU)。这

种技术可能会减少网络往返次数并提高网络吞吐量。有关更多信息，请参阅《Amazon EC2 用户指南》中的巨型帧（9001 MTU）。

有关数据库实例类的信息，请参阅 数据库实例类。要确定等同于 Amazon EC2 实例类型的数据库实例类，请将 db. 放置在 Amazon EC2 实例类型名称之前。例如，r5.8xlarge Amazon EC2 实例等同于 db.r5.8xlarge 数据库实例类。

提高网络带宽

使用 NetworkReceiveThroughput 和 NetworkTransmitThroughput Amazon CloudWatch 指标监控数据库实例上的传入和传出网络流量。这些指标可以帮助您确定网络带宽是否足以满足您的工作负载。

如果您的网络带宽不够，请增加它。如果 Amazon 客户端或您的数据库实例已达到网络带宽限制，增加带宽的唯一方法是增加数据库实例大小。有关更多信息，请参阅 数据库实例类类型。

有关 CloudWatch 指标的更多信息，请参阅Amazon RDS 的 Amazon CloudWatch 指标。

监控网络性能的最大值

如果您使用的是 Amazon EC2 客户端，Amazon EC2 会提供网络性能指标的最大值，包括聚合入站和出站网络带宽。它还提供连接跟踪功能，以确保按预期返回数据包以及域名系统 (DNS) 等服务的链接本地服务访问。要监控这些最大值，请使用当前的增强型联网驱动程序并监控客户端的网络性能。

有关更多信息，请参阅《Amazon EC2 用户指南》中的监控您的 Amazon EC2 实例的网络性能，以及《Amazon EC2 用户指南》中的监控您的 Amazon EC2 实例的网络性能。

监控处于“空闲事务”状态的事务

检查您是否有越来越多的 idle in transaction 连接。要做到这一点，请监控 pg_stat_activity 表中的 state 列。您可能能够通过运行类似于以下内容的查询来识别连接源。

```
select client_addr, state, count(1) from pg_stat_activity
where state like 'idle in transaction%'
group by 1,2
order by 3 desc
```

Client:ClientWrite

当 RDS for PostgreSQL 等待将数据写入客户端时，会发生 Client:ClientWrite 事件。

主题

支持的引擎版本

RDS for PostgreSQL 版本 10 及更高版本支持此等待事件信息。

上下文

客户端进程必须先读取从 RDS for PostgreSQL 数据库集群接收的所有数据，然后集群才能发送更多数据。集群在将更多数据发送给客户端之前等待的时间为 `Client:ClientWrite` 事件。

RDS for PostgreSQL 数据库实例与客户端之间的网络吞吐量降低可能会导致此事件。客户端的 CPU 压力和网络饱和也可能导致此事件。CPU 压力是 CPU 被充分利用并且有任务等待 CPU 时间的时间。网络饱和是当数据库和客户端之间的网络传输的数据超出其处理能力之时。

等待次数增加的可能原因

`Client:ClientWrite` 显示在主要等待中的常见原因包括以下各项：

网络延迟增加

RDS for PostgreSQL 数据库实例和客户端之间的网络延迟可能会增加。较高的网络延迟会增加客户端接收数据所需的时间。

客户端负载增加

客户端上可能存在 CPU 压力或网络饱和。客户端负载的增加会延迟从 RDS for PostgreSQL 数据库实例接收数据的过程。

发送到客户端的大量数据

RDS for PostgreSQL 数据库实例可能会向客户端发送大量数据。客户端可能无法像集群发送数据那样地快速接收数据。诸如大型表的副本之类的活动可能会导致 `Client:ClientWrite` 事件增加。

操作

根据等待事件的原因，我们建议采取不同的操作。

主题

- 将客户端放置在与集群相同的可用区和 VPC 子网中。
- 使用当前一代实例
- 减少发送到客户端的数据量
- 扩展客户端

将客户端放置在与集群相同的可用区和 VPC 子网中。

为了减少网络延迟并提高网络吞吐量，请将客户端放在与 RDS for PostgreSQL 数据库实例相同的可用区和虚拟私有云（VPC）子网中。

使用当前一代实例

在某些情况下，您可能没有使用支持巨型帧的数据库实例类。如果您在 Amazon EC2 上运行应用程序，请考虑为客户端使用当前一代实例。另外，在客户端操作系统上配置最大传输单位 (MTU)。这种技术可能会减少网络往返次数并提高网络吞吐量。有关更多信息，请参阅《Amazon EC2 用户指南》中的巨型帧（9001 MTU）。

有关数据库实例类的信息，请参阅 数据库实例类。要确定等同于 Amazon EC2 实例类型的数据库实例类，请将 `db.` 放置在 Amazon EC2 实例类型名称之前。例如，`r5.8xlarge` Amazon EC2 实例等同于 `db.r5.8xlarge` 数据库实例类。

减少发送到客户端的数据量

如果可能，请调整应用程序以减少 RDS for PostgreSQL 数据库实例发送给客户端的数据量。进行这样的调整可以减轻客户端上的 CPU 和网络争用。

扩展客户端

使用 Amazon CloudWatch 或其他主机指标，确定您的客户端当前是受 CPU 或网络带宽的限制，还是受此两者的限制。如果客户端受到限制，请相应地扩展您的客户端。

CPU

当线程在 CPU 中处于活动状态或正在等待 CPU 时，会发生此事件。

主题

- 支持的引擎版本

支持的引擎版本

此等待事件信息与所有的 RDS for PostgreSQL 版本相关。

上下文

中央处理单元 (CPU) 是运行指令的计算机的组件。例如，CPU 指令执行算术运算并在内存中交换数据。如果查询增加了通过数据库引擎执行的指令的数量，则运行查询所花费的时间将增加。CPU 调度正在为进程提供 CPU 时间。调度由操作系统的内核编排。

主题

如何判断此等待何时发生

该 CPU 等待事件表示后端进程在 CPU 中处于活动状态或正在等待 CPU。您知道，当查询显示以下信息时会发生这种情况：

- The `pg_stat_activity.state` column has the value `active`。
- `pg_stat_activity` 中的 `wait_event_type` 和 `wait_event` 列都是 `null`。

要查看正在使用或等待 CPU 的后端进程，请运行以下查询。

```
SELECT *
FROM   pg_stat_activity
WHERE  state = 'active'
AND    wait_event_type IS NULL
AND    wait_event IS NULL;
```

DBLoadCPU 指标

CPU 的性能详情指标为 `DBLoadCPU`。`DBLoadCPU` 的值可能与 Amazon CloudWatch 指标 `CPUUtilization` 的值不同。后一个指标是从 Hypervisor 中收集的，用于数据库实例。

os.cpuUtilization 指标

性能详情操作系统指标提供有关 CPU 利用率的详细信息。例如，您可以显示以下指标：

- `os.cpuUtilization.nice.avg`
- `os.cpuUtilization.total.avg`
- `os.cpuUtilization.wait.avg`
- `os.cpuUtilization.idle.avg`

性能详情将数据库引擎的 CPU 使用情况报告为 `os.cpuUtilization.nice.avg`。

CPU 调度的可能原因

操作系统（OS）内核处理 CPU 的调度。当 CPU 处于活动状态时，进程可能需要等待才能获得调度。CPU 在执行计算时处于活动状态。当它有一个未运行的空闲线程（也即，一个等待内存输入/输出的空闲线程）时也处于活动状态。这种类型的 I/O 主导着典型的数据库工作负载。

满足以下条件时，进程可能会等待获得 CPU 调度：

- CloudWatch `CPUUtilization` 指标接近 100%。
- 平均负载大于 vCPU 的数量，表示负载过重。您可以在性能详情中的操作系统指标部分找到 `loadAverageMinute` 指标。

等待次数增加的可能原因

当此事件的发生率超过正常（可能表示性能问题）时，典型的原因包括以下几点。

主题

- [突然猛增的可能原因]
- [长期高频的可能原因]
- [极端状况]

突然猛增的可能原因

突然猛增的最可能原因如下：

- 您的应用程序打开了太多与数据库同时连接。这种情况被称为“连接风暴”。
- 您的应用程序工作负载在以下任一方面发生变化：
  - 新查询
  - 数据集的大小增加
  - 索引维护或创建
  - 新函数
  - 新的运营商
  - 并行查询执行增加
- 您的查询执行计划已更改。在某些情况下，更改可能会导致缓冲区增加。例如，查询之前使用索引，而现在正在使用顺序扫描。在这种情况下，查询需要更多的 CPU 才能实现同样的目标。

长期高频的可能原因

长期反复出现的事件的最可能原因：

- CPU 上同时运行的后端进程太多。这些进程可以是并行工件。
- 查询执行不佳，因为它们需要大量缓冲区。

极端状况

如果所有可能的原因都不是实际原因，则可能会发生以下情况：

- CPU 正在换入和换出进程。
- 如果关闭了大页功能，CPU 可能正在管理页表条目。原定设置情况下，微型、小型和中型数据库实例类以外的所有数据库实例类都会开启内存管理功能。有关更多信息，请参阅 适用于 RDS for PostgreSQL 的大页 。

操作

如果 CPU 等待事件主导着数据库活动，它不一定表示性能问题。只在性能下降时应对此事件。

主题

- 调查数据库是否导致 CPU 增加
- 确定连接数量是否增加
- 响应工作负载变化

调查数据库是否导致 CPU 增加

检查性能详情中的 os.cpuUtilization.nice.avg 指标。如果此值远低于 CPU 使用率，则非数据库进程是 CPU 的主要贡献者。

确定连接数量是否增加

检查 Amazon CloudWatch 中的 DatabaseConnections 指标。您的操作取决于 CPU 等待事件增加期间该数量是增加还是减少。

连接增加

如果连接数量增加，请将消耗 CPU 的后端进程数与 vCPU 的数量进行比较。以下是可能的情况：

- 消耗 CPU 的后端进程数量少于 vCPU 的数量。

  在这种情况下，连接数量不是问题。但是，您仍然可以尝试降低 CPU 利用率。
- 消耗 CPU 的后端进程数量大于 vCPU 的数量。

  在这种情况下，需考虑以下选项：
  - 减少连接到数据库的后端进程的数量。例如，实施连接池解决方案，例如 RDS 代理。要了解更多信息，请参阅 的 Amazon RDS 代理。
  - 升级实例大小以获得更多 vCPU 数量。
  - 如果适用，将一些只读工作负载重新导向到读取器节点。

连接未增加

检查性能详情中的 blks_hit 指标。寻找 blks_hit 增加与 CPU 使用率之间的相关性。以下是可能的情况：

- CPU 使用率和 blks_hit 具有相关性。

  在这种情况下，找到与 CPU 使用率相关联的主要 SQL 语句，然后查找计划更改。您可以使用下面的方法之一：
  - 手动解释计划并将其与预期的执行计划进行比较。

  - 寻找每秒数据块命中量和每秒局部数据块命中量的增加。在性能详情控制面板的主要 SQL 部分中，选择 Preferences（首选项）。
- CPU 使用率和 `blks_hit` 无关。

  在这种情况下，请确定是否出现以下任一情况：

  - 应用程序正在快速连接到数据库并断开与数据库的连接。

    通过打开 `log_connections` 和 `log_disconnections`，然后分析 PostgreSQL 日志来诊断此行为。考虑使用 `pgbadger` 日志分析器。有关更多信息，请参阅 https://github.com/darold/pgbadger。
  - 操作系统已超载。

    在这种情况下，性能详情显示，后端进程使用 CPU 的时间比平时更长。在性能详情 `os.cpuUtilization` 指标或 CloudWatch `CPUUtilization` 指标中寻找证据。如果操作系统过载，请查看增强监控指标以进一步诊断。具体来说，请查看进程列表以及每个进程占用的 CPU 百分比。
  - 主要 SQL 语句消耗的 CPU 太多。

    检查与 CPU 使用率相关联的语句，看看它们是否可以使用更少的 CPU。运行 `EXPLAIN` 命令，并将重点放在影响最大的计划节点上。考虑使用 PostgreSQL 执行计划可视化工具。要试用此工具，请参阅 http://explain.dalibo.com/。

响应工作负载变化

如果您的工作负载发生了变化，请查找以下类型的更改：

新查询

  检查是否预计会出现新的查询。如果是，请确保他们的执行计划和每秒执行次数符合预期。

数据集的大小增加

  确定分区（如果尚未实施）是否有帮助。此策略可能会减少查询需要检索的页数。

索引维护或创建

  检查维护时间表是否符合预期。最佳实践是在高峰活动之外安排维护活动。

新函数

  检查这些功能在测试期间是否按预期运行。具体来说，检查每秒执行次数是否符合预期。

新的运营商

检查它们在测试期间是否按预期运行。

运行并行查询的增加

确定是否出现以下任一情况：

- 所涉及的关系或索引的规模突然增长，因此它们与 min_parallel_table_scan_size 或 min_parallel_index_scan_size 显著不同。
- 最近对 parallel_setup_cost 或 parallel_tuple_cost 进行了更改。
- 最近对 max_parallel_workers 或 max_parallel_workers_per_gather 进行了更改。

IO:BufFileRead 和 IO:BufFileWrite

IO:BufFileRead 和 IO:BufFileWrite 事件发生在 RDS for PostgreSQL 创建临时文件时。当操作需要的内存超过当前定义的工作内存参数时，它们会将临时数据写入持久性存储。此操作有时被称为"溢出到磁盘"。

主题

- 支持的引擎版本
- 上下文
- 等待次数增加的可能原因
- 操作

支持的引擎版本

RDS for PostgreSQL 的所有版本均支持此等待事件信息。

上下文

IO:BufFileRead 和 IO:BufFileWrite 与工作内存区域和维护工作内存区域有关。有关这些本地内存区域的更多信息，请参阅 PostgreSQL 文档中的资源消耗量。

work_mem 的原定设置值为 4MB。如果一个会话并行执行操作，则处理并行性的每个工件将使用 4MB 的内存。出于此原因，请仔细设置 work_mem。如果您将值增加的太大，则运行很多会话的数据库可能会占用太多内存。如果您将值设置得太低，RDS for PostgreSQL 会在本地存储中创建临时文件。这些临时文件的磁盘输入/输出可能会降低性能。

如果观察到以下事件顺序，则数据库可能正在生成临时文件：

1. 可用性突然急剧下降
2. 可用空间的快速恢复

您可能还会看到“chainsaw”模式。此模式可能表明您的数据库在不断创建小文件。

等待次数增加的可能原因

一般来说，这些等待事件是占用内存比 `work_mem` 或 `maintenance_work_mem` 参数分配的内存更多的操作造成。为了进行补偿，操作会写入临时文件。`IO:BufFileRead` 和 `IO:BufFileWrite` 事件的常见原因包括以下内容：

需要比工作内存区域中存在的内存更多的查询

具有以下特征的查询使用工作内存区域：

- 哈希联接
- `ORDER BY` 子句
- `GROUP BY` 子句
- `DISTINCT`
- 窗口函数
- `CREATE TABLE AS SELECT`
- 具体化视图刷新

需要比维护工作内存区域中存在的内存更多的语句

以下语句使用维护工作内存区域：

- `CREATE INDEX`
- `CLUSTER`

操作

根据等待事件的原因，我们建议采取不同的操作。

主题

- [识别问题]
- [请检查您的联接查询]

- 检查您的 ORDER BY 和 GROUP BY 查询
- 避免使用 DISTINCT 操作
- 考虑使用窗口函数而不是 GROUP BY 函数
- 调查具体化视图和 CTAS 语句
- 在重新构建索引时使用 pg_repack
- 聚集表时，增加 maintenance_work_mem
- 优化内存以防止 IO:BufFileRead 和 IO:BufFileWrite

识别问题

假设存在一种情况，性能详情尚未开启的一种情况，而您怀疑 IO:BufFileRead 和 IO:BufFileWrite 的发生频率比正常情况高。要确定问题的根源，可以将 log_temp_files 参数设置为记录所生成的临时文件超过指定阈值 KB 的所有查询。原定设置情况下，log_temp_files 设置为 -1，这将关闭此日志记录功能。如果您将此参数设置为 0，RDS for PostgreSQL 会记录所有临时文件。如果值为 1024，RDS for PostgreSQL 会记录所生成的临时文件大于 1MB 的所有查询。有关 log_temp_files 更多信息，请参阅 PostgreSQL 文档中的错误报告和日志记录。

请检查您的联接查询

您的查询很可能使用联接。例如，以下查询将四个表联接到一起。

```
SELECT *
       FROM "order"
 INNER JOIN order_item
       ON (order.id = order_item.order_id)
 INNER JOIN customer
       ON (customer.id = order.customer_id)
 INNER JOIN customer_address
       ON (customer_address.customer_id = customer.id AND
           order.customer_address_id = customer_address.id)
 WHERE customer.id = 1234567890;
```

临时文件使用率激增的可能原因是查询本身存在问题。例如，中断的子句可能无法正确筛选联接。考虑以下示例中的第二个内联接。

```
SELECT *
       FROM "order"
 INNER JOIN order_item
       ON (order.id = order_item.order_id)
```

```
INNER JOIN customer
      ON (customer.id = customer.id)
INNER JOIN customer_address
      ON (customer_address.customer_id = customer.id AND
          order.customer_address_id = customer_address.id)
WHERE customer.id = 1234567890;
```

前面的查询错误地将 customer.id 与 customer.id 进行了联接，在每个客户和每个订单之间生成了笛卡尔积。这种类型的意外联接会生成大型临时文件。根据表的大小，笛卡尔查询甚至可以填满存储空间。满足以下条件时，您的应用程序可能会有笛卡尔联接：

- 您可以看到存储可用性大幅下降，然后是快速恢复。
- 现在没有创建任何索引。
- 现在没有发布任何 CREATE TABLE FROM SELECT 语句。
- 没有进行任何具体化视图的刷新。

要查看是否使用正确的键联接表，请检查查询和对象关系映射指令。请记住，应用程序的某些查询不会总是被调用，而且有些查询是动态生成的。

检查您的 ORDER BY 和 GROUP BY 查询

在某些情况下，ORDER BY 子句可能会导致过多的临时文件。请考虑以下准则：

- 当需要对它们进行排序时，只包括 ORDER BY 子句中的列。本指南对于返回数千行并在 ORDER BY 子句中指定很多列的查询尤其重要。
- 考虑创建索引以在 ORDER BY 子句与具有相同升序或降序的列匹配时对它们进行加速。部分索引更可取，因为它们较小。较小的索引可以更快地读取和遍历。
- 如果为可以接受 null 值的列创建索引，请考虑是希望将 null 值存储在索引的末尾还是在索引的开头存储。

  如果可能，通过筛选结果集来减少需要排序的行数。如果您使用 WITH 子句语句或子查询，请记住，内部查询会生成一个结果集并会将其传递给外部查询。查询可以筛选出的行越多，查询需要进行的排序就越少。
- 如果您不需要获取完整的结果集，请使用 LIMIT 子句。例如，如果您只想要前五行，则使用 LIMIT 子句的查询不会继续生成结果。这样，查询需要更少的内存和临时文件。

使用 GROUP BY 子句的查询也可能需要临时文件。GROUP BY 查询通过使用以下函数汇总值：

- COUNT
- AVG
- MIN
- MAX
- SUM
- STDDEV

要优化 GROUP BY 查询，请按照 ORDER BY 查询的建议。

避免使用 DISTINCT 操作

如果可能的话，避免使用 DISTINCT 操作来删除重复的行。查询返回的不必要和重复的行越多，DISTINCT 操作就会越昂贵。如果可能，请在 WHERE 子句中添加筛选条件，即使您对不同的表使用相同的筛选条件。筛选查询并正确联接可以提高性能并减少资源使用。它还可以防止错误的报告和结果。

如果您需要将 DISTINCT 用于同一个表的多行，请考虑创建复合索引。将索引中的多个列进行分组可以缩短评估不同行的时间。此外，如果您使用 RDS for PostgreSQL 版本 10 或更高版本，则可以使用 CREATE STATISTICS 命令在多个列之间关联统计数据。

考虑使用窗口函数而不是 GROUP BY 函数

使用 GROUP BY，您可以更改结果集，然后检索聚合的结果。使用窗口函数，可以在不更改结果集的情况下聚合数据。窗口函数使用 OVER 子句来跨查询定义的集执行计算，从而将一行与另一行关联。您可以使用窗口函数中的所有 GROUP BY 函数，但也可以使用以下函数：

- RANK
- ARRAY_AGG
- ROW_NUMBER
- LAG
- LEAD

为了尽量减少窗口函数生成的临时文件的数量，请在需要两个不同的聚合时删除同一结果集的重复项。请考虑以下查询。

```
SELECT sum(salary) OVER (PARTITION BY dept ORDER BY salary DESC) as sum_salary
     , avg(salary) OVER (PARTITION BY dept ORDER BY salary ASC) as avg_salary
```

```
  FROM empsalary;
```

您可以使用如下 WINDOW 子句重新写入查询。

```
SELECT sum(salary) OVER w as sum_salary
         , avg(salary) OVER w as_avg_salary
    FROM empsalary
  WINDOW w AS (PARTITION BY dept ORDER BY salary DESC);
```

原定设置情况下，RDS for PostgreSQL 执行计划器会整合类似的节点，这样它就不会重复操作。但是，通过对窗口数据块使用显式声明，您可以更轻松地维护查询。您还可以通过防止重复来提高性能。

调查具体化视图和 CTAS 语句

当具体化视图刷新时，它会运行查询。此查询可以包含 GROUP BY、ORDER BY 或 DISTINCT 之类的操作。刷新期间，您可能会观察到大量临时文件以及等待事件 IO:BufFileWrite 和 IO:BufFileRead。同样地，当您根据 SELECT 语句创建表时，CREATE TABLE 语句会运行查询。要减少所需的临时文件，请优化查询。

在重新构建索引时使用 pg_repack

创建索引时，引擎会对结果集进行排序。随着表的大小增加以及索引列中的值变得更加多样化，临时文件需要更多的空间。在大多数情况下，如果不修改维护工作内存区域，就无法阻止为大型表创建临时文件。有关 maintenance_work_mem 的更多信息，请参阅 PostgreSQL 文档中的https://www.postgresql.org/docs/current/runtime-config-resource.html。

重新创建大型索引时可能的解决方法是使用 pg_repack 扩展。有关更多信息，请参阅 pg_repack 文档中的用最少的锁定重新组织 PostgreSQL 数据库中的表。有关在 RDS for PostgreSQL 数据库实例中设置此扩展的信息，请参阅使用 pg_repack 扩展减少表和索引的膨胀。

聚集表时，增加 maintenance_work_mem

CLUSTER 命令基于 index_name 指定的现有索引聚集 table_name 指定的表。RDS for PostgreSQL 以物理方式重新创建表以匹配给定索引的顺序。

当磁性存储普遍存在时，集群很常见，因为存储吞吐量有限。由于基于 SSD 的存储已经很常见，因此集群不太受欢迎。但是，如果对表进行聚集，您仍然可以根据表大小、索引、查询等稍微提高性能。

如果您运行 CLUSTER 命令并观察到等待事件 IO:BufFileWrite 和 IO:BufFileRead，请优化 maintenance_work_mem。将内存大小增加到相当大的量。较高的值意味着引擎可以使用更多内存进行集群操作。

优化内存以防止 IO:BufFileRead 和 IO:BufFileWrite

在某些情况下，您需要优化内存。您的目标是使用相应的参数平衡以下消耗区域间的内存，如下所示。

- `work_mem` 值
- 折扣 `shared_buffers` 值后剩余的内存
- 已打开和使用中的最大连接数，受限于 `max_connections`

有关优化内存的更多信息，请参阅 PostgreSQL 文档中的资源消耗量。

增加工作内存区域的大小

在某些情况下，唯一的选项是增加会话使用的内存。如果您的查询编写正确并且正在使用正确的键进行连接，请考虑增加 `work_mem` 值。

要了解查询生成了多少个临时文件，请将 `log_temp_files` 设置为 `0`。如果您将 `work_mem` 值增加为日志中标识的最大值，则可以防止查询生成临时文件。但是，`work_mem` 为每个连接或并行工件设置每个计划节点的最大值。如果数据库有 5000 个连接，并且每个连接使用 256MiB 内存，则引擎需要 1.2TiB 的 RAM。因此，您的实例可能会耗尽内存。

为共享缓冲池预留足够的内存

您的数据库使用很多内存区域，例如共享缓冲池，而不仅仅是工作内存区域。在增加 `work_mem` 之前考虑这些额外的内存区域的要求。

例如，假设您的 RDS for PostgreSQL 实例类为 db.r5.2xlarge。此实例类拥有 64GiB 的内存。原定设置情况下，将 25% 的内存预留为共享缓冲池。减去分配给共享内存区域的量后，仍然有 16384 MB。不要将剩余内存专门分配给工作内存区域，因为操作系统和引擎还需要内存。

您可以分配给 `work_mem` 的内存取决于实例类。如果您使用较大的实例类，则可用的内存更多。但是，在前面的示例中，您不能使用超过 16GiB 的内存。否则，当内存耗尽时，您的实例将变得不可用。要从不可用状态恢复实例，RDS for PostgreSQL 自动化服务会自动重新启动。

管理连接数

假设您的数据库实例具有 5000 个同时连接。每个连接至少使用 4MiB 的 `work_mem` 连接的内存消耗过高可能会降低性能。作为响应，您可进行以下选择：

- 升级到更大的实例类。
- 使用连接代理或池程序减少同时数据库连接的数量。

对于代理，请考虑 Amazon RDS 代理、pgBouncer 或基于您的应用程序的连接池程序。此解决方案减轻了 CPU 负载。它还可以降低所有连接都需要工作内存区域时的风险。当数据库连接较少时，您可以增加 `work_mem` 的值。通过这种方式，您可以减少 `IO:BufFileRead` 和 `IO:BufFileWrite` 等待事件的发生率。此外，等待工作内存区域的查询显著加速。

IO:DataFileRead

当由于分页在共享内存中不可用，连接等待后端进程从存储中读取所需分页时，会发生 `IO:DataFileRead` 事件。

主题

- 支持的引擎版本
- 上下文
- 等待次数增加的可能原因
- 操作

支持的引擎版本

RDS for PostgreSQL 的所有版本均支持此等待事件信息。

上下文

所有查询和数据处理 (DML) 操作都会访问缓冲池中的页面。可以诱导读取的语句包括 `SELECT`、`UPDATE` 和 `DELETE`。例如，`UPDATE` 可以从表或索引中读取页面。如果请求或更新的页面不在共享缓冲池中，则此读取可能会导致 `IO:DataFileRead` 事件。

由于共享缓冲池是有限的，所以它可以填满。在这种情况下，对不在内存中的页面的请求会强制数据库从磁盘中读取数据块。如果 `IO:DataFileRead` 事件频繁发生，您的共享缓冲池可能太小，从而无法容纳您的工作负载。对于读取大量不适合缓冲池的行的 `SELECT` 查询，此问题很严重。有关缓冲区池的更多信息，请参阅 PostgreSQL 文档中的资源消耗量。

等待次数增加的可能原因

`IO:DataFileRead` 事件的常见原因包括以下各项：

连接激增

> 您可能会发现多个连接生成相同数量的 IO:DataFileRead 等待事件。在这种情况下，`IO:DataFileRead` 事件可能会发生激增（突然大幅度增加）。

执行顺序扫描的 SELECT 和 DML 语句

您的应用程序可能正在执行新的操作。或者，现有的操作可能会因为新的执行计划而发生变化。在这种情况下，请查找具有更大的 `seq_scan` 值的表格（特别是大型表格）。通过查询 `pg_stat_user_tables` 查找它们。要跟踪生成更多读取操作的查询，请使用扩展 `pg_stat_statements`。

适用于大型数据集的 CTAS 和 CREATE INDEX

CTAS 是一个 `CREATE TABLE AS SELECT` 语句。如果您使用大型数据集作为源来运行 CTAS，或者在大型表上创建索引，则可能会发生 `IO:DataFileRead` 事件。创建索引时，数据库可能需要使用顺序扫描读取整个对象。当页面不在内存中时，CTAS 会生成 `IO:DataFile` 读取。

多个 vacuum 工件同时运行

vacuum 工件可以手动或自动触发。我们建议采取积极的 vacuum 策略。但是，当表中有许多更新或删除的行时，`IO:DataFileRead` 等待增加。回收空间后，花在 `IO:DataFileRead` 上的 vacuum 时间减少。

摄取大量数据

当您的应用程序提取大量数据时，`ANALYZE` 操作可能会更频繁地发生。`ANALYZE` 进程可以由 Autovacuum 启动程序触发，也可以手动调用。

`ANALYZE` 操作可以读取表的子集。必须扫描的页数通过将 30 乘以 `default_statistics_target` 值进行计算。有关更多信息，请参阅 PostgreSQL 文档。`default_statistics_target` 参数接受 1 到 10000 之间的值，其中原定设置值为 100。

资源匮乏

如果消耗了实例网络带宽或 CPU，`IO:DataFileRead` 事件可能会更频繁地发生。

操作

根据等待事件的原因，我们建议采取不同的操作。

主题

- 检查谓词筛选条件是否存在生成等待的查询
- 尽量减少维护操作的影响
- 响应大量连接

检查谓词筛选条件是否存在生成等待的查询

假设您确定了正在生成 IO:DataFileRead 等待事件的特定查询。您可以使用以下方法识别它们：

- 性能详情
- 目录视图，例如扩展程序 pg_stat_statements 提供的视图
- 目录视图 pg_stat_all_tables，如果它定期显示物理读取数量增加
- pg_statio_all_tables 视图，如果它显示 _read 计数器正在增加

我们建议您确定这些查询的谓词（WHERE 子句）中使用了哪些筛选条件。请遵循以下准则：

- 运行 EXPLAIN 命令。在输出中，确定使用的扫描类型。顺序扫描不一定表示存在问题。与使用筛选条件的查询相比，使用顺序扫描的查询自然会产生更多的 IO:DataFileRead 事件。

  了解 WHERE 子句中列出的列是否已编入索引。如果没有，请考虑为此列创建索引。这种方法避免了顺序扫描并减少了 IO:DataFileRead 事件。如果某个查询具有限制性筛选条件并且仍然生成顺序扫描，请评估是否使用了正确的索引。
- 了解查询是否正在访问非常大的表。在某些情况下，对表进行分区可以提高性能，从而允许查询只读取必要的分区。
- 检查联接操作的基数（总行数）。请注意您在筛选条件中为您的 WHERE 子句传递的值的限制性。如果可能，请优化查询以减少在计划的每个步骤中传递的行数。

尽量减少维护操作的影响

维护操作（例如 VACUUM 和 ANALYZE）非常重要。我们建议您不要将其关闭，因为您会找到与这些维护操作相关的 IO:DataFileRead 等待事件。以下方法可以最大限度地减少这些操作的影响：

- 在非高峰时段手动运行维护操作。此方法可防止数据库达到自动操作的阈值。
- 对于非常大的表，请考虑对表进行分区。这种方法减少了维护操作的开销。数据库只访问需要维护的分区。
- 当您摄取大量数据时，请考虑禁用自动分析功能。

当以下公式为真时，系统会自动为表触发 Autovacuum 功能。

```
pg_stat_user_tables.n_dead_tup > (pg_class.reltuples x autovacuum_vacuum_scale_factor)
 + autovacuum_vacuum_threshold
```

视图 pg_stat_user_tables 和目录 pg_class 有多个行。一行可以对应于表中的一行。这个公式假设 reltuples 适用于特定的表。参数 autovacuum_vacuum_scale_factor（原定设置为 0.20）和 autovacuum_vacuum_threshold（原定设置为 50 个元组）通常在全局范围内为整个实例设置。但是，您可以为特定表设置不同的值。

主题

查找不必要地占用空间的表

要查找不必要地占用空间的表，可以使用 PostgreSQL pgstattuple 扩展中的函数。原定设置情况下，此扩展（模块）在所有 RDS for PostgreSQL 数据库实例上均可用，并且可以使用以下命令在实例上进行实例化。

```
CREATE EXTENSION pgstattuple;
```

有关此扩展的更多信息，请参阅 PostgreSQL 文档中的 pgstattuple。

您可以在应用程序中检查表和索引膨胀。有关更多信息，请参诊断表和索引膨胀。

查找不必要地占用空间的索引

要查找臃肿的索引并估计在您具有读取权限的表上不必要地消耗的空间量，可以运行以下查询。

```
-- WARNING: rows with is_na = 't' are known to have bad statistics ("name" type is not
 supported).
-- This query is compatible with PostgreSQL 8.2 and later.

SELECT current_database(), nspname AS schemaname, tblname, idxname,
 bs*(relpages)::bigint AS real_size,
  bs*(relpages-est_pages)::bigint AS extra_size,
  100 * (relpages-est_pages)::float / relpages AS extra_ratio,
  fillfactor, bs*(relpages-est_pages_ff) AS bloat_size,
  100 * (relpages-est_pages_ff)::float / relpages AS bloat_ratio,
  is_na
  -- , 100-(sub.pst).avg_leaf_density, est_pages, index_tuple_hdr_bm,
  -- maxalign, pagehdr, nulldatawidth, nulldatahdrwidth, sub.reltuples, sub.relpages
  -- (DEBUG INFO)
FROM (
```

```
  SELECT coalesce(1 +
       ceil(reltuples/floor((bs-pageopqdata-pagehdr)/(4+nulldatahdrwidth)::float)), 0
       -- ItemIdData size + computed avg size of a tuple (nulldatahdrwidth)
    ) AS est_pages,
    coalesce(1 +
       ceil(reltuples/floor((bs-pageopqdata-pagehdr)*fillfactor/
(100*(4+nulldatahdrwidth)::float))), 0
    ) AS est_pages_ff,
    bs, nspname, table_oid, tblname, idxname, relpages, fillfactor, is_na
    -- , stattuple.pgstatindex(quote_ident(nspname)||'.'||quote_ident(idxname)) AS
 pst,
    -- index_tuple_hdr_bm, maxalign, pagehdr, nulldatawidth, nulldatahdrwidth,
 reltuples
    -- (DEBUG INFO)
  FROM (
    SELECT maxalign, bs, nspname, tblname, idxname, reltuples, relpages, relam,
 table_oid, fillfactor,
      ( index_tuple_hdr_bm +
          maxalign - CASE -- Add padding to the index tuple header to align on MAXALIGN
            WHEN index_tuple_hdr_bm%maxalign = 0 THEN maxalign
            ELSE index_tuple_hdr_bm%maxalign
          END
        + nulldatawidth + maxalign - CASE -- Add padding to the data to align on
 MAXALIGN
            WHEN nulldatawidth = 0 THEN 0
            WHEN nulldatawidth::integer%maxalign = 0 THEN maxalign
            ELSE nulldatawidth::integer%maxalign
          END
      )::numeric AS nulldatahdrwidth, pagehdr, pageopqdata, is_na
      -- , index_tuple_hdr_bm, nulldatawidth -- (DEBUG INFO)
    FROM (
      SELECT
        i.nspname, i.tblname, i.idxname, i.reltuples, i.relpages, i.relam, a.attrelid
 AS table_oid,
        current_setting('block_size')::numeric AS bs, fillfactor,
        CASE -- MAXALIGN: 4 on 32bits, 8 on 64bits (and mingw32 ?)
          WHEN version() ~ 'mingw32' OR version() ~ '64-bit|x86_64|ppc64|ia64|amd64'
 THEN 8
          ELSE 4
        END AS maxalign,
        /* per page header, fixed size: 20 for 7.X, 24 for others */
        24 AS pagehdr,
        /* per page btree opaque data */
        16 AS pageopqdata,
```

```
        /* per tuple header: add IndexAttributeBitMapData if some cols are null-able */
        CASE WHEN max(coalesce(s.null_frac,0)) = 0
          THEN 2 -- IndexTupleData size
          ELSE 2 + (( 32 + 8 - 1 ) / 8)
          -- IndexTupleData size + IndexAttributeBitMapData size ( max num filed per
 index + 8 - 1 /8)
        END AS index_tuple_hdr_bm,
        /* data len: we remove null values save space using it fractionnal part from
 stats */
        sum( (1-coalesce(s.null_frac, 0)) * coalesce(s.avg_width, 1024)) AS
 nulldatawidth,
        max( CASE WHEN a.atttypid = 'pg_catalog.name'::regtype THEN 1 ELSE 0 END ) > 0
 AS is_na
      FROM pg_attribute AS a
        JOIN (
          SELECT nspname, tbl.relname AS tblname, idx.relname AS idxname,
            idx.reltuples, idx.relpages, idx.relam,
            indrelid, indexrelid, indkey::smallint[] AS attnum,
            coalesce(substring(
              array_to_string(idx.reloptions, ' ')
               from 'fillfactor=([0-9]+)')::smallint, 90) AS fillfactor
          FROM pg_index
            JOIN pg_class idx ON idx.oid=pg_index.indexrelid
            JOIN pg_class tbl ON tbl.oid=pg_index.indrelid
            JOIN pg_namespace ON pg_namespace.oid = idx.relnamespace
          WHERE pg_index.indisvalid AND tbl.relkind = 'r' AND idx.relpages > 0
        ) AS i ON a.attrelid = i.indexrelid
        JOIN pg_stats AS s ON s.schemaname = i.nspname
          AND ((s.tablename = i.tblname AND s.attname =
 pg_catalog.pg_get_indexdef(a.attrelid, a.attnum, TRUE))
          -- stats from tbl
          OR  (s.tablename = i.idxname AND s.attname = a.attname))
          -- stats from functional cols
        JOIN pg_type AS t ON a.atttypid = t.oid
      WHERE a.attnum > 0
      GROUP BY 1, 2, 3, 4, 5, 6, 7, 8, 9
    ) AS s1
  ) AS s2
    JOIN pg_am am ON s2.relam = am.oid WHERE am.amname = 'btree'
) AS sub
-- WHERE NOT is_na
ORDER BY 2,3,4;
```

查找符合 Autovacuum 操作条件的表

要查找符合 Autovacuum 操作条件的表，请运行以下查询。

```
--This query shows tables that need vacuuming and are eligible candidates.
--The following query lists all tables that are due to be processed by autovacuum.
-- During normal operation, this query should return very little.
WITH  vbt AS (SELECT setting AS autovacuum_vacuum_threshold
              FROM pg_settings WHERE name = 'autovacuum_vacuum_threshold')
    , vsf AS (SELECT setting AS autovacuum_vacuum_scale_factor
              FROM pg_settings WHERE name = 'autovacuum_vacuum_scale_factor')
    , fma AS (SELECT setting AS autovacuum_freeze_max_age
              FROM pg_settings WHERE name = 'autovacuum_freeze_max_age')
    , sto AS (SELECT opt_oid, split_part(setting, '=', 1) as param,
                split_part(setting, '=', 2) as value
              FROM (SELECT oid opt_oid, unnest(reloptions) setting FROM pg_class) opt)
SELECT
    '"'||ns.nspname||'"."'||c.relname||'"' as relation
    , pg_size_pretty(pg_table_size(c.oid)) as table_size
    , age(relfrozenxid) as xid_age
    , coalesce(cfma.value::float, autovacuum_freeze_max_age::float)
 autovacuum_freeze_max_age
    , (coalesce(cvbt.value::float, autovacuum_vacuum_threshold::float) +
         coalesce(cvsf.value::float,autovacuum_vacuum_scale_factor::float) *
 c.reltuples)
         as autovacuum_vacuum_tuples
    , n_dead_tup as dead_tuples
FROM pg_class c
JOIN pg_namespace ns ON ns.oid = c.relnamespace
JOIN pg_stat_all_tables stat ON stat.relid = c.oid
JOIN vbt on (1=1)
JOIN vsf ON (1=1)
JOIN fma on (1=1)
LEFT JOIN sto cvbt ON cvbt.param = 'autovacuum_vacuum_threshold' AND c.oid =
 cvbt.opt_oid
LEFT JOIN sto cvsf ON cvsf.param = 'autovacuum_vacuum_scale_factor' AND c.oid =
 cvsf.opt_oid
LEFT JOIN sto cfma ON cfma.param = 'autovacuum_freeze_max_age' AND c.oid = cfma.opt_oid
WHERE c.relkind = 'r'
AND nspname <> 'pg_catalog'
AND (
    age(relfrozenxid) >= coalesce(cfma.value::float, autovacuum_freeze_max_age::float)
    or
    coalesce(cvbt.value::float, autovacuum_vacuum_threshold::float) +
```

```
      coalesce(cvsf.value::float,autovacuum_vacuum_scale_factor::float) * c.reltuples
 <= n_dead_tup
    -- or 1 = 1
)
ORDER BY age(relfrozenxid) DESC;
```

响应大量连接

当您监控 Amazon CloudWatch 时，您可能会发现 `DatabaseConnections` 指标激增。这种增加表示与数据库的连接数量有所增加。我们建议采取以下方法：

- 限制应用程序可与每个实例一起打开的连接数。如果您的应用程序具有嵌入式连接池功能，请设置合理数量的连接。根据实例中的 vCPU 可以有效并行处理的数量来确定数量。

  如果您的应用程序没有使用连接池功能，请考虑使用 Amazon RDS 代理或替代方案。这种方法允许您的应用程序打开与负载均衡器的多个连接。然后，均衡器可以打开与数据库的数量有限的连接。由于并行运行的连接减少，您的数据库实例在内核中执行的上下文切换会减少。查询的进度应该更快，从而导致等待事件减少。有关更多信息，请参阅 [的 Amazon RDS 代理](#)。
- 尽可能利用 RDS for PostgreSQL 的只读副本。当您的应用程序运行只读操作时，将这些请求发送到只读副本。此方法可减少主（写入器）节点上的输入/输出压力。
- 请考虑纵向扩展数据库实例。更高容量的实例类可提供更多内存，这为 RDS for PostgreSQL 提供了一个更大的共享缓冲池来容纳页面。较大的大小还为数据库实例提供了更多的 vCPU 来处理连接。当生成 `IO:DataFileRead` 等待事件的操作为写入时，更多的 vCPU 会特别有用。

IO:WALWrite

主题

- [支持的引擎版本](#)
- [上下文](#)
- [等待次数增加的可能原因](#)
- [操作](#)

支持的引擎版本

RDS for PostgreSQL 10 及更高版本的所有版本均支持此等待事件信息。

上下文

数据库中生成预写日志数据的活动首先填满 WAL 缓冲区，然后异步写入磁盘。当 SQL 会话等待 WAL 数据完成向磁盘写入以便它可以释放事务的 COMMIT 调用时，会生成等待事件 `IO:WALWrite`。

等待次数增加的可能原因

如果这种等待事件经常发生，则应查看您的工作负载以及工作负载执行的更新类型及其频率。特别要查找以下类型的活动。

繁重的 DML 活动

更改数据库表中的数据不会立即发生。对一个表的插入可能需要等待从另一客户端对同一个表进行插入或更新。用于更改数据值的数据操作语言（DML）语句（INSERT、UPDATE、DELETE、COMMIT、ROLLBACK TRANSATION）可能会导致争用，从而导致预写日志文件等待刷新缓冲区。以下 Amazon RDS 性能详情指标反映了这种情况，这些指标表明 DML 活动非常多。

- `tup_inserted`
- `tup_updated`
- `tup_deleted`
- `xact_rollback`
- `xact_commit`

有关这些指标的更多信息，请参阅 适用于 Amazon RDS for PostgreSQL 的性能详情计数器。

频繁的检查点活动

过多的检查点会导致更多数量的 WAL 文件。在 RDS for PostgreSQL 中，整页写入始终处于“开启”状态。整页写入有助于防止数据丢失。但是，当检查点活动过于频繁时，系统可能会遇到整体性能问题。在 DML 活动非常多的系统上尤其如此。在某些情况下，您可能会在 `postgresql.log` 中发现错误消息，指出“检查点出现频率过高”。

我们建议您在优化检查点时，谨慎平衡性能与出现异常关闭时进行恢复所需的预期时间。

操作

我们建议通过执行以下操作来减少此等待事件的数量。

主题

减少提交的数量

为了减少提交的数量，您可以将语句合并到事务数据块中。使用 Amazon RDS 性能详情来检查正在运行的查询类型。您也可以将大型维护操作移至非高峰时段。例如，在非生产时间创建索引或使用 `pg_repack` 操作。

监控您的检查点

您可以监控两个参数，以查看 RDS for PostgreSQL 数据库实例对于检查点写入 WAL 文件的频率。

- `log_checkpoints` –原定设置情况下，该参数为“on”。它会导致对于每个检查点向 PostgreSQL 日志发送一条消息。这些日志消息包括写入的缓冲区数量、写入缓冲区所花的时间以及针对给定检查点添加、删除或回收的 WAL 文件数。

  有关此参数的更多信息，请参阅 PostgreSQL 文档中的错误报告和日志记录。
- `checkpoint_warning` – 此参数为检查点频率设置阈值（以秒为单位），超过该阈值将生成警告。原定设置情况下，不在 RDS for PostgreSQL 中设置此参数。您可以设置此参数的值，以便在 RDS for PostgreSQL 数据库实例中的数据库更改以 WAL 文件大小无法处理的速率写入时收到警告。例如，假设您将参数设置为 30。如果您的 RDS for PostgreSQL 实例需要写入更改的频率超过每 30 秒，则会向 PostgreSQL 日志发送“检查点出现频率过高”的警告。这可能表明应增加您的 `max_wal_size` 值。

  有关更多信息，请参阅 PostgreSQL 文档中的预写日志。

纵向扩展 IO

这种类型的输入/输出（IO）等待事件可以通过扩展每秒进行读写操作的次数（IOPS）以提供更快的 IO 来进行修复。扩展 IO 比扩展 CPU 更可取，因为扩展 CPU 会导致更多的 IO 争用，原因在于增加的 CPU 可以处理更多的工作，从而使 IO 瓶颈进一步恶化。一般情况下，建议在执行扩展操作之前优化您的工作负载。

专用日志卷（DLV）

您可以利用 Amazon RDS 控制台、Amazon CLI 或 Amazon RDS API，将专用日志卷（DLV）用于使用预调配 IOPS（PIOPS）存储的数据库实例。DLV 将 PostgreSQL 数据库事务日志移动到与包含数据库表的卷不同的存储卷中。有关更多信息，请参阅 专用日志卷（DLV）。

IPC:并行等待事件

以下 `IPC:parallel wait events` 表明会话正在等待与并行查询执行操作相关的进程间通信。

- `IPC:BgWorkerStartup`：进程正在等待并行工作进程完成其启动序列。在初始化工作线程以执行并行查询时会发生这种情况。
- `IPC:BgWorkerShutdown`：进程正在等待并行工作进程完成其关闭序列。这发生在并行查询执行的清理阶段。
- `IPC:ExecuteGather`：在查询执行期间，进程正在等待从并行工作进程接收数据。当领导进程需要从其工作线程中收集结果时，会发生这种情况。
- `IPC:ParallelFinish`：进程正在等待并行工作线程完成其执行并报告其最终结果。这种情况发生在并行查询执行的完成阶段。

主题

支持的引擎版本

Aurora PostgreSQL 的所有版本均支持此等待事件信息。

上下文

PostgreSQL 中的并行查询执行涉及多个进程协同工作以处理单个查询。当确定查询适合并行化时，领导进程会根据 `max_parallel_workers_per_gather` 参数设置与一个或多个并行工作进程进行协调。领导进程在各工作线程之间划分工作，每个工作线程独立处理自己的那部分数据，然后将结果收集回领导进程。

> **Note**
>
> 每个并行工作线程均作为一个单独的进程运行，其资源需求类似于完整的用户会话。这意味着，与非并行查询相比，具有 4 个工作线程的并行查询消耗的资源（CPU、内存、I/O 带宽）最多可达 5 倍，因为领导进程和每个工作进程都保持自己的资源分配。例如，诸如 `work_mem` 的设置单独应用于每个工作线程，可能会使所有进程的总内存使用量成倍增加。

并行查询架构由三个主要部分组成：

- 领导进程：启动并行操作、划分工作负载并与工作进程进行协调的主进程。
- 工作进程：并行执行查询各部分的后台进程。
- 收集/收集合并：将来自多个工作进程的结果合并回领导进程的操作

在并行执行期间，进程需要通过进程间通信（IPC）机制相互通信。这些 IPC 等待事件发生在不同的阶段：

- 工作线程启动：初始化并行工作线程时
- 数据交换：当工作线程处理数据并将结果发送给领导进程时
- 工作线程关闭：当并行执行完成且终止工作线程时
- 同步点：当进程需要协调或等待其它进程完成其任务时

了解这些等待事件对于诊断与并行查询执行相关的性能问题至关重要，尤其是在可能同时执行多个并行查询的高并发环境中。

等待次数增加的可能原因

有几个因素可能导致与并行相关的 IPC 等待事件增加：

并行查询的高并发性

> 当多个并行查询同时运行时，可能会导致资源争用和 IPC 操作的等待时间增加。这在事务量或分析工作负载较高的系统中尤其常见。

并行查询计划不够理想

如果查询计划器选择效率低下的并行计划，则可能会导致不必要的并行化或工作线程间的工作分配不佳。这可能会导致 IPC 等待增加，尤其是对于 `IPC:ExecuteGather` 和 `IPC:ParallelFinish` 事件。这些规划问题通常源于过时的统计数据和表/索引膨胀。

并行工作线程频繁启动和关闭

频繁启动和终止并行工作线程的短期查询可能会导致 `IPC:BgWorkerStartup` 和 `IPC:BgWorkerShutdown` 事件增加。这种情况经常出现在具有许多小型可并行查询的 OLTP 工作负载中。

资源约束

有限的 CPU、内存或 I/O 容量可能会导致并行执行出现瓶颈，从而导致所有 IPC 事件的等待时间增加。例如，如果 CPU 饱和，则工作进程可能需要更长的时间才能启动或处理自己的那部分工作。

复杂的查询结构

具有多个并行性级别的查询（例如，并行联接后跟并行聚合）可能会导致更复杂的 IPC 模式并可能增加等待时间，尤其是对于 `IPC:ExecuteGather` 事件。

大型结果集

生成大型结果集的查询可能会导致 `IPC:ExecuteGather` 等待时间增加，因为领导进程将花费更多的时间来收集和处理来自工作进程的结果。

了解这些因素有助于诊断和解决与 Aurora PostgreSQL 中的并行查询执行相关的性能问题。

操作

当您看到与并行查询相关的等待时，这通常表示后端进程正在协调或等待并行工作进程。在执行并行计划期间，这些等待很常见。您可以通过监控并行工作线程使用情况、查看参数设置以及调整查询执行和资源分配，来调查并缓解这些等待的影响。

主题

- 分析查询计划是否存在并行性效率低下问题
- 监控并行查询使用情况
- 查看和调整并行查询设置
- 优化资源分配
- 调查连接管理
- 查看和优化维护操作

分析查询计划是否存在并行性效率低下问题

并行查询执行可能经常导致系统不稳定、CPU 峰值和不可预测的查询性能变化。彻底分析并行性是否确实可以改善特定工作负载至关重要。使用 EXPLAIN ANALYZE 查看并行查询执行计划。

在会话级别暂时禁用并行性以比较计划效率：

```
SET max_parallel_workers_per_gather = 0;
EXPLAIN ANALYZE <your_query>;
```

重新启用并行性并进行比较：

```
RESET max_parallel_workers_per_gather;
EXPLAIN ANALYZE <your_query>;
```

如果禁用并行性会产生更好或更一致的结果，则考虑使用 SET 命令在会话级别为特定查询禁用并行性。为了获得更广泛的影响，您可能需要通过调整数据库参数组中的相关参数，来在实例级别禁用并行性。有关更多信息，请参阅在 Amazon RDS 中修改数据库参数组中的参数。

监控并行查询使用情况

使用以下查询来深入了解并行查询活动和容量：

检查活动的并行工作进程：

```
SELECT
    COUNT(*)
FROM
    pg_stat_activity
WHERE
    backend_type = 'parallel worker';
```

此查询显示活动的并行工作进程的数量。较高的值可能表示为“max_parallel_workers”配置了较高的值，您可能需要考虑将其降低。

检查并发并行查询：

```
SELECT
    COUNT(DISTINCT leader_pid)
FROM
    pg_stat_activity
WHERE
    leader_pid IS NOT NULL;
```

此查询返回已启动并行查询的不同领导进程的数量。此处的数字较高表示多个会话正在并发运行并行查询，这可能会增加对 CPU 和内存的需求。

查看和调整并行查询设置

查看以下参数以确保它们与您的工作负载一致：

- `max_parallel_workers`：所有会话中并行工作线程的总数。
- `max_parallel_workers_per_gather`：每个查询的最大工作线程数。

对于 OLAP 工作负载，增加这些值可以提高性能。对于 OLTP 工作负载，通常首选较低的值。

```
SHOW max_parallel_workers;
SHOW max_parallel_workers_per_gather;
```

优化资源分配

监控 CPU 利用率，如果 vCPU 的数量一直很高，并且应用程序从并行查询中受益，请考虑调整 vCPU 的数量。确保有足够的内存可用于并行操作。

- 使用性能详情指标来确定系统是否受到 CPU 限制。
- 每个并行工作线程都使用自己的 `work_mem`。确保总内存使用量在实例限制范围内。

并行查询可能比非并行查询消耗的资源要多得多，因为每个工作进程都是一个完全独立的进程，它对系统的影响与额外的用户会话大致相同。在为此设置选择值时，以及在配置其它用于控制资源利用率的设置（例如 `work_mem`）时，应考虑到这一点。有关更多信息，请参阅 [PostgreSQL 文档](#)。资源限制（例如 `work_mem`）单独应用于每个工作线程，这意味着跨所有进程的总利用率可能比针对任何单个进程的正常利用率高得多。

如果工作负载高度并行化，可以考虑增加 vCPU 数量或调整内存参数。

调查连接管理

如果遇到连接耗尽的情况，请查看应用程序连接池策略。考虑在应用程序级别实现连接池（如果尚未使用）。

查看和优化维护操作

协调索引创建和其它维护任务，以防止资源争用。考虑将这些操作安排在非高峰时段。避免在用户查询负载较高的时期安排繁重的维护工作（例如并行索引构建）。这些操作可能会消耗并行工作线程并影响常规查询的性能。

Lock:advisory

当 PostgreSQL 应用程序使用锁定来协调多个会话之间的活动时，会发生 `Lock:advisory` 事件。

主题

相关引擎版本

此等待事件信息与 RDS for PostgreSQL 版本 9.6 及更高版本相关。

上下文

PostgreSQL 咨询锁是由用户的应用程序代码显式锁定和解锁的应用程序级别的合作锁。应用程序可以使用 PostgreSQL 咨询锁来协调多个会话之间的活动时。与常规锁、对象级锁或行级锁不同的是，应用程序对锁的生命周期拥有完全控制权。有关更多信息，请参阅 PostgreSQL 文档中的咨询锁。

咨询锁可以在事务结束之前释放，也可以在事务中由会话持有。对于隐式的、系统强制的锁，例如 `CREATE INDEX` 语句获取的对表的互斥访问锁，情况并非如此。

有关用于获取（锁定）和释放（解锁）咨询锁的函数的说明，请参阅 PostgreSQL 文档中的咨询锁函数。

咨询锁是在常规的 PostgreSQL 锁定系统之上实施的，并且在 pg_locks 系统视图中可见。

原因

这种锁类型由显式使用它的应用程序完全控制。作为查询的一部分为每个行获取的咨询锁可能会导致锁激增或长期积累。

当以获得比查询返回的行更多的锁的方式运行查询时，会发生这些影响。应用程序最终必须释放每个锁，但是如果在未返回的行上获取锁，则应用程序无法找到所有锁。

以下示例来自 PostgreSQL 文档中的咨询锁。

```
SELECT pg_advisory_lock(id) FROM foo WHERE id > 12345 LIMIT 100;
```

在此示例中，LIMIT 子句只能在内部选择行并锁定其 ID 值后停止查询的输出。当数据量不断增长导致计划人员选择在开发过程中未测试的其他执行计划时，可能会突然发生这种情况。在这种情况下，由于应用程序会为锁定的每个 ID 值调用 pg_advisory_unlock，会发生累积。但是，在这种情况下，它找不到在未返回的行上获取的锁集合。由于锁是在会话级别获取的，因此它们不会在事务结束时自动释放。

阻止锁定尝试激增的另一个可能原因是意外的冲突。在这些冲突中，应用程序的不相关部分错误地共享了相同的锁 ID 空间。

操作

查看应用程序对咨询锁的使用情况，并详细说明在应用程序流中获取和释放每种类型的咨询锁的位置和时间。

确定会话是获取太多锁定还是长时间运行的会话没有尽早释放锁，从而导致锁缓慢累积。您可以通过使用 pg_terminate_backend(pid) 结束会话来纠正会话级别锁定的缓慢累积。

正在等待咨询锁的客户端显示在带有 wait_event_type=Lock 和 wait_event=advisory 的 pg_stat_activity 中。您可以通过查询相同 pid 的 pg_locks 系统视图来获取特定的锁定值，以寻找 locktype=advisory 和 granted=f。

然后您可以通过查询具有 granted=t 的相同咨询锁的 pg_locks 来识别阻止的会话，如以下示例所示。

```
SELECT blocked_locks.pid AS blocked_pid,
         blocking_locks.pid AS blocking_pid,
         blocked_activity.usename AS blocked_user,
```

```
        blocking_activity.usename AS blocking_user,
        now() - blocked_activity.xact_start AS blocked_transaction_duration,
        now() - blocking_activity.xact_start AS blocking_transaction_duration,
        concat(blocked_activity.wait_event_type,':',blocked_activity.wait_event) AS
blocked_wait_event,
        concat(blocking_activity.wait_event_type,':',blocking_activity.wait_event) AS
blocking_wait_event,
        blocked_activity.state AS blocked_state,
        blocking_activity.state AS blocking_state,
        blocked_locks.locktype AS blocked_locktype,
        blocking_locks.locktype AS blocking_locktype,
        blocked_activity.query AS blocked_statement,
        blocking_activity.query AS blocking_statement
   FROM pg_catalog.pg_locks blocked_locks
   JOIN pg_catalog.pg_stat_activity blocked_activity ON blocked_activity.pid =
blocked_locks.pid
   JOIN pg_catalog.pg_locks blocking_locks
       ON blocking_locks.locktype = blocked_locks.locktype
       AND blocking_locks.DATABASE IS NOT DISTINCT FROM blocked_locks.DATABASE
       AND blocking_locks.relation IS NOT DISTINCT FROM blocked_locks.relation
       AND blocking_locks.page IS NOT DISTINCT FROM blocked_locks.page
       AND blocking_locks.tuple IS NOT DISTINCT FROM blocked_locks.tuple
       AND blocking_locks.virtualxid IS NOT DISTINCT FROM blocked_locks.virtualxid
       AND blocking_locks.transactionid IS NOT DISTINCT FROM
blocked_locks.transactionid
       AND blocking_locks.classid IS NOT DISTINCT FROM blocked_locks.classid
       AND blocking_locks.objid IS NOT DISTINCT FROM blocked_locks.objid
       AND blocking_locks.objsubid IS NOT DISTINCT FROM blocked_locks.objsubid
       AND blocking_locks.pid != blocked_locks.pid
   JOIN pg_catalog.pg_stat_activity blocking_activity ON blocking_activity.pid =
blocking_locks.pid
   WHERE NOT blocked_locks.GRANTED;
```

所有的咨询锁 API 函数都有两组参数，一个 `bigint` 参数或两个 `integer` 参数：

- 对于具有一个 `bigint` 参数的 API 函数，上面的 32 位在 `pg_locks.classid` 中，下面的 32 位在 `pg_locks.objid` 中。
- 对于具有两个 `integer` 参数的 API 函数，第一个参数是 `pg_locks.classid`，第二个参数是 `pg_locks.objid`。

`pg_locks.objsubid` 值表示使用了哪个 API 表单：1 表示一个 `bigint` 参数；2 表示两个 `integer` 参数。

Lock:extend

当后端进程正在等待锁定关系以对其进行扩展，而另一个进程出于同样目的锁定该关系时，会发生 Lock:extend 事件。

主题

- 支持的引擎版本
- 上下文
- 等待次数增加的可能原因
- 操作

支持的引擎版本

RDS for PostgreSQL 的所有版本均支持此等待事件信息。

上下文

事件 Lock:extend 表示后端进程正在等待扩展另一个后端进程在扩展该关系时保持锁定的关系。由于每次只有一个进程可以扩展关系，因此系统会生成 Lock:extend 等待事件。INSERT、COPY 和 UPDATE 操作可以生成此事件。

等待次数增加的可能原因

当 Lock:extend 事件的发生率超过正常（可能表示性能问题）时，典型原因包括以下几点：

对同一表的并发插入或更新激增

插入或更新同一表的查询的并发会话数可能会增加。

网络带宽不足

数据库实例上的网络带宽可能不足以满足当前工作负载的存储通信需求。这可能会导致存储延迟，从而导致 Lock:extend 事件增加。

操作

根据等待事件的原因，我们建议采取不同的操作。

主题

- 减少同一关系的并发插入和更新

- [提高网络带宽]

减少同一关系的并发插入和更新

首先，确定 tup_inserted 和 tup_updated 指标是否有增加，以及此等待事件是否伴随增加。如果是这样，请检查哪些关系在插入和更新操作中处于高争用状态。要确定这一点，请查询 pg_stat_all_tables 视图，以了解 n_tup_ins 和 n_tup_upd 字段中的值。有关 pg_stat_all_tables 视图的信息，请参阅 PostgreSQL 文档中的 [pg_stat_all_tables]。

要获取有关正在阻止和已阻止的查询的更多信息，请如以下示例所示查询 pg_stat_activity：

```
SELECT
    blocked.pid,
    blocked.usename,
    blocked.query,
    blocking.pid AS blocking_id,
    blocking.query AS blocking_query,
    blocking.wait_event AS blocking_wait_event,
    blocking.wait_event_type AS blocking_wait_event_type
FROM pg_stat_activity AS blocked
JOIN pg_stat_activity AS blocking ON blocking.pid = ANY(pg_blocking_pids(blocked.pid))
where
blocked.wait_event = 'extend'
and blocked.wait_event_type = 'Lock';

   pid  | usename   |             query              | blocking_id |
             blocking_query                           | blocking_wait_event |
 blocking_wait_event_type
  ------+----------+------------------------------+-------------
+------------------------------------------------------------------
+---------------------+--------------------------
   7143 |  myuser   | insert into tab1 values (1); |        4600 | INSERT INTO tab1 (a)
 SELECT s FROM generate_series(1,1000000) s; | DataFileExtend      | IO
```

在您确定有助于增加 Lock:extend 事件的关系后，请使用以下方法来减少争用：

- 查明是否可以使用分区来减少同一个表的争用。将插入或更新的元组分成不同的分区可以减少争用。有关分区的信息，请参阅 [使用 pg_partman 扩展管理 PostgreSQL 分区]。
- 如果等待事件主要是由于更新活动造成的，请考虑减少关系的 fillfactor 值。这可以减少更新期间对新数据块的请求。fillfactor 是表的存储参数，用于确定打包表页面的最大空间量。它表示为页面总空间的百分比。有关 fillfactor 参数的更多信息，请参阅 PostgreSQL 文档中的 [CREATE TABLE]。

> **Important**
>
> 我们强烈建议您在更改 fillfactor 时测试系统，因为更改此值可能会对性能产生负面影响，这具体取决于您的工作负载。

提高网络带宽

要查看写入延迟是否增加，请检查 CloudWatch 中的 `WriteLatency` 指标。如果有，请使用 `WriteThroughput` 和 `ReadThroughput` Amazon CloudWatch 指标监控数据库实例上与存储相关的流量。这些指标可以帮助您确定网络带宽是否足以满足您的工作负载的存储活动。

如果您的网络带宽不够，请增加它。如果您的数据库实例已达到网络带宽限制，增加带宽的唯一方法是增加数据库实例大小。

有关 CloudWatch 指标的更多信息，请参阅Amazon RDS 的 Amazon CloudWatch 实例级指标。有关每个数据库实例类的网络性能的信息，请参阅 Amazon RDS 的 Amazon CloudWatch 实例级指标。

Lock:Relation

当查询等待获取当前被另一个事务锁定的表或视图（关系）上的锁定时，会发生 `Lock:Relation` 事件。

主题

支持的引擎版本

RDS for PostgreSQL 的所有版本均支持此等待事件信息。

上下文

大多数 PostgreSQL 命令隐式使用锁来控制对表中数据的并发访问。您还可以通过 `LOCK` 命令在应用程序代码中显式使用这些锁。许多锁定模式彼此不兼容，它们可以在尝试访问同一对象时阻止事务。发生这种情况时，RDS for PostgreSQL 会生成一个 `Lock:Relation` 事件。以下是一些常见的示例：

- ACCESS EXCLUSIVE 之类的独占锁可以阻止所有并发访问。数据定义语言 (DDL) 操作（例如 DROP TABLE、TRUNCATE、VACUUM FULL 和 CLUSTER）隐式获取 ACCESS EXCLUSIVE 锁定。ACCESS EXCLUSIVE 也是用于未显式指定模式的 LOCK TABLE 语句的原定设置锁定模式。
- 在表上使用 CREATE INDEX (without CONCURRENT) 与数据操作语言 (DML) 语句 UPDATE、DELETE 和 INSERT 有冲突，这些语句可获取 ROW EXCLUSIVE 锁定。

有关表级锁和冲突锁模式的更多信息，请参阅 PostgreSQL 文档中的显式锁定。

阻止查询和事务通常通过以下方式之一解锁阻止：

- 阻止查询 – 应用程序可以取消查询或者用户可以结束该过程。引擎还可以因会话的语句超时或死锁检测机制而强制结束查询。
- 阻止事务 – 事务在运行 ROLLBACK 或 COMMIT 语句时停止阻止。当会话被客户端或网络问题断开连接或结束时，也会自动发生回滚。当数据库引擎关闭、系统内存不足等时，可以结束会话。

等待次数增加的可能原因

当 Lock:Relation 事件的发生频率高于正常值时，可能表明存在性能问题。典型的原因包括：

增加与表锁冲突的并发会话

> 用冲突锁定模式锁定相同表格的查询的并发会话数可能会增加。

维护操作

> VACUUM 和 ANALYZE 之类的运行状况维护操作可以显著增加冲突锁的数量。VACUUM FULL 获取 ACCESS EXCLUSIVE 锁，ANALYSE 获取 SHARE UPDATE EXCLUSIVE锁。这两种类型的锁都可能导致 Lock:Relation 等待事件。应用程序数据维护操作（例如刷新具体化视图）也可以增加阻止的查询和事务。

读取器实例的锁定

> 写入器和读取器持有的关系锁之间可能存在冲突。目前，仅 ACCESS EXCLUSIVE 关系锁复制到读取器实例。但是，ACCESS EXCLUSIVE 关系锁将与读取器持有的任何 ACCESS SHARE 关系锁冲突。这可能会导致读取器上的锁定关系等待事件增加。

操作

根据等待事件的原因，我们建议采取不同的操作。

主题

- [减少阻止 SQL 语句的影响]
- [尽量减少维护操作的影响]

减少阻止 SQL 语句的影响

为了减少阻止 SQL 语句的影响，请尽可能修改应用程序代码。以下是减少数据块的两种常见方法：

- 使用 `NOWAIT` 选项 – 一些 SQL 命令，例如 `SELECT` 和 `LOCK` 语句，支持此选项。如果无法立即获取锁定，则 `NOWAIT` 指令将取消请求锁定的查询。这种方法可以帮助防止阻止会话导致其后面堆积阻止的会话。

  例如：假设事务 A 正在等待事务 B 所持有的锁定。现在，如果 B 请求对被事务 C 锁定的表进行锁，那么事务 A 可能会被阻止，直到事务 C 完成。但是，如果事务 B 在请求对 C 进行锁定时使用 `NOWAIT`，它可能会很快失败，并确保事务 A 不必无限期等待。
- 使用 `SET lock_timeout` – 设置 `lock_timeout` 值，以限制 SQL 语句等待获取关系锁的时间。如果锁未在指定的超时内获取，则请求锁定的事务将被取消。在会话级别设置此值。

尽量减少维护操作的影响

维护操作（例如 `VACUUM` 和 `ANALYZE`）非常重要。我们建议您不要将其关闭，因为您会找到与这些维护操作相关的 `Lock:Relation` 等待事件。以下方法可以最大限度地减少这些操作的影响：

- 在非高峰时段手动运行维护操作。
- 要减少由 Autovacuum 任务导致的 `Lock:Relation` 等待，执行任何需要的 Autovacuum 优化。有关优化 Autovacuum 的信息，请参阅《Amazon RDS 用户指南》中的[在 Amazon RDS 上使用 PostgreSQL Autovacuum]。

Lock:transactionid

当事务正在等待行级锁定时，会发生 `Lock:transactionid` 事件。

主题

- [支持的引擎版本]
- [上下文]
- [等待次数增加的可能原因]

- 操作

支持的引擎版本

RDS for PostgreSQL 的所有版本均支持此等待事件信息。

上下文

当事务试图获取已被授予同时运行的事务的行级锁时，会发生事件 `Lock:transactionid`。显示 `Lock:transactionid` 等待事件的会话因此锁定被阻止。当阻止事务在 `COMMIT` 或 `ROLLBACK` 语句中结束后，被阻止的事务可以继续进行。

RDS for PostgreSQL 的多版本并发控制语义保证读取器不会阻止写入器，写入器也不会阻止读取器。为了发生行级冲突，正在阻止和已阻止的事务必须发出以下类型的冲突语句：

- `UPDATE`
- `SELECT … FOR UPDATE`
- `SELECT … FOR KEY SHARE`

语句 `SELECT … FOR KEY SHARE` 是一种特殊情况。数据库使用子句 `FOR KEY SHARE` 以优化参照完整性的性能。一行上的行级锁定可以组织引用该行的其他表上的 `INSERT`、`UPDATE` 和 `DELETE` 命令。

等待次数增加的可能原因

当此事件出现频率超过正常时，原因通常是 `UPDATE`、`SELECT … FOR UPDATE` 或 `SELECT … FOR KEY SHARE` 语句结合以下条件。

主题

- 高并发性
- 空闲事务
- 长时间运行的事务

高并发性

RDS for PostgreSQL 可以使用细粒度的行级锁定语义。满足以下条件时，行级冲突的可能性会增加：

- 高度并发的工作负载争用相同的行。
- 并发性增加。

空闲事务

有时，pg_stat_activity.state 列会显示值 idle in transaction。对于已开始事务但尚未发布 COMMIT 或 ROLLBACK 的会话，会显示此值。如果 pg_stat_activity.state 值不为 active，pg_stat_activity 中显示的查询是完成运行的最新版本。阻止会话没有主动处理查询，因为未完成的事务持有锁定。

如果空闲事务获得了行级锁，则可能会阻止其他会话获取它。这种情况导致等待事件 Lock:transactionid 频繁发生。要诊断问题，请检查来自的 pg_stat_activity 和 pg_locks 的输出。

长时间运行的事务

长时间运行的事务会获得很长一段时间的锁定。这些长期保留的锁定可以阻止其他事务运行。

操作

行锁定是 UPDATE、SELECT … FOR UPDATE 或 SELECT … FOR KEY SHARE 语句之间发生的冲突。在尝试解决方案之前，请先了解这些语句何时在同一行上运行。使用此信息选择以下各部分所述的策略。

主题

- 响应高并发
- 回应空闲事务
- 响应长期运行的事务

响应高并发

如果并发性是问题，请尝试以下方法之一：

- 降低应用程序中的并发率。例如，减少活动会话的数量。
- 实施连接池。要了解如何使用 RDS 代理进行池连接，请参阅 的 Amazon RDS 代理。
- 设计应用程序或数据模型以避免争用 UPDATE 和 SELECT … FOR UPDATE 语句。您还可以减少 SELECT … FOR KEY SHARE 语句访问的外键的数量。

回应空闲事务

如果 `pg_stat_activity.state` 显示 `idle in transaction`，请使用以下策略：

- 尽可能开启自动提交。这种方法可防止事务在等待 `COMMIT` 或 `ROLLBACK` 时阻止其他事务。
- 搜索缺失 `COMMIT`、`ROLLBACK` 或 `END` 的代码路径。
- 确保应用程序中的异常处理逻辑始终具有通向有效 `end of transaction` 的路径。
- 确保您的应用程序在结束与`COMMIT` 或 `ROLLBACK` 的事务后处理查询结果。

响应长期运行的事务

如果长时间运行的事务导致频繁发生 `Lock:transactionid`，请尝试以下策略：

- 在长时间运行的事务中保持行锁定。
- 尽可能通过实现自动提交来限制查询的长度。

Lock:tuple

在后端进程等待获取元组锁定时，会发生 `Lock:tuple` 事件。

主题

- [支持的引擎版本]
- [上下文]
- [等待次数增加的可能原因]
- [操作]

支持的引擎版本

RDS for PostgreSQL 的所有版本均支持此等待事件信息。

上下文

事件 `Lock:tuple` 表示一个后端正在等待获取元组上的锁定，而另一个后端在同一个元组上保持冲突锁定。下表说明了会话生成 `Lock:tuple` 事件的场景。

时间	会话 1	会话 2	会话 3
t1	开始事务。		
t2	更新第 1 行。		
t3		更新第 1 行。会话获取元组上的独占锁定，然后等待会话 1 通过提交或回滚来释放锁。	
t4			更新第 1 行。会话等待会话 2 才能释放元组上的独占锁定。

或者您可以使用基准测试工具 pgbench 来模拟此等待事件。配置大量并发会话以使用自定义 SQL 文件更新表中的同一行。

要了解冲突锁模式的更多信息，请参阅 PostgreSQL 文档中的显式锁定。要了解有关 pgbench 的更多信息，请参阅 PostgreSQL 文档中的 pgbench。

等待次数增加的可能原因

当此事件的发生率超过正常（可能表示性能问题）时，典型原因包括以下几点：

- 大量并发会话试图通过运行 `UPDATE` 或 `DELETE` 语句获取相同元组的冲突锁定。
- 高度并发的会话正在使用 `FOR UPDATE` 或 `FOR NO KEY UPDATE` 锁定模式运行 `SELECT` 语句。
- 各种因素促使应用程序或连接池打开更多会话以执行相同的操作。由于新会话正在尝试修改相同的行，数据库负载可能会激增，`Lock:tuple` 可以出现。

有关更多信息，请参阅 PostgreSQL 文档中的行级锁定。

操作

根据等待事件的原因，我们建议采取不同的操作。

主题

- 调查应用程序逻辑

- 查找阻止器会话
- 在并发性高时降低并发性
- 排查瓶颈

调查应用程序逻辑

了解阻止器会话是否已经处于 idle in transaction 状态很长一段时间。如果是这样，请考虑结束阻止器会话，作为短期解决方案。您可以使用 pg_terminate_backend 函数。有关此函数的更多信息，请参阅 PostgreSQL 文档中的服务器信号函数。

要获得长期解决方案，请执行以下操作：

- 调整应用程序逻辑。
- 使用 idle_in_transaction_session_timeout 参数。此参数可结束空闲时间超过指定时间的已打开事务的任何会话。有关更多信息，请参阅 PostgreSQL 文档中的客户端连接原定设置。
- 尽可能多地使用自动提交。有关更多信息，请参阅 PostgreSQL 文档中的 SET AUTOCOMMIT。

查找阻止器会话

在 Lock:tuple 等待事件发生时，通过找出哪些锁相互依赖来识别阻止器和已阻止的会话。有关更多信息，请参阅 PostgreSQL wiki 中的锁定依赖项信息。

以下示例查询所有会话，并对 tuple 进行筛选，通过 wait_time 进行排序。

```
SELECT blocked_locks.pid AS blocked_pid,
         blocking_locks.pid AS blocking_pid,
         blocked_activity.usename AS blocked_user,
         blocking_activity.usename AS blocking_user,
         now() - blocked_activity.xact_start AS blocked_transaction_duration,
         now() - blocking_activity.xact_start AS blocking_transaction_duration,
         concat(blocked_activity.wait_event_type,':',blocked_activity.wait_event) AS
 blocked_wait_event,
         concat(blocking_activity.wait_event_type,':',blocking_activity.wait_event) AS
 blocking_wait_event,
         blocked_activity.state AS blocked_state,
         blocking_activity.state AS blocking_state,
         blocked_locks.locktype AS blocked_locktype,
         blocking_locks.locktype AS blocking_locktype,
         blocked_activity.query AS blocked_statement,
```

```
        blocking_activity.query AS blocking_statement
    FROM pg_catalog.pg_locks blocked_locks
    JOIN pg_catalog.pg_stat_activity blocked_activity ON blocked_activity.pid =
blocked_locks.pid
    JOIN pg_catalog.pg_locks blocking_locks
        ON blocking_locks.locktype = blocked_locks.locktype
        AND blocking_locks.DATABASE IS NOT DISTINCT FROM blocked_locks.DATABASE
        AND blocking_locks.relation IS NOT DISTINCT FROM blocked_locks.relation
        AND blocking_locks.page IS NOT DISTINCT FROM blocked_locks.page
        AND blocking_locks.tuple IS NOT DISTINCT FROM blocked_locks.tuple
        AND blocking_locks.virtualxid IS NOT DISTINCT FROM blocked_locks.virtualxid
        AND blocking_locks.transactionid IS NOT DISTINCT FROM
blocked_locks.transactionid
        AND blocking_locks.classid IS NOT DISTINCT FROM blocked_locks.classid
        AND blocking_locks.objid IS NOT DISTINCT FROM blocked_locks.objid
        AND blocking_locks.objsubid IS NOT DISTINCT FROM blocked_locks.objsubid
        AND blocking_locks.pid != blocked_locks.pid
    JOIN pg_catalog.pg_stat_activity blocking_activity ON blocking_activity.pid =
blocking_locks.pid
    WHERE NOT blocked_locks.GRANTED;
```

在并发性高时降低并发性

`Lock:tuple` 事件可能会不断发生，特别是在繁忙的工作负载时间。在这种情况下，考虑降低非常繁忙的行的高并发率。通常，只有几个行控制队列或布尔逻辑，这使得这些行非常繁忙。

您可以根据业务需求、应用程序逻辑和工作负载类型使用不同的方法来降低并发性。例如，您可以执行以下操作：

- 重新设计表和数据逻辑以降低高并发性。
- 更改应用程序逻辑以降低行级别的高并发性。
- 使用行级锁定利用和重新设计查询。
- 使用具有重试操作的 `NOWAIT` 子句。
- 考虑使用乐观和混合锁定逻辑并发控制。
- 考虑更改数据库隔离级别。

排查瓶颈

当出现诸如 CPU 匮乏或 Amazon EBS 带宽的最大使用率的瓶颈时，可能会发生 `Lock:tuple`。要减少瓶颈，请考虑以下方法：

- 纵向扩展您的实例类类型。
- 优化资源密集型查询。
- 更改应用程序逻辑。
- 存档很少访问的数据。

LWLock:BufferMapping (LWLock:buffer_mapping)

当会话正在等待将数据块与共享缓冲池中的缓冲区关联起来时，会发生此事件。

> **Note**
>
> 对于 RDS for PostgreSQL 版本 13 及更高版本，此事件命名为 `LWLock:BufferMapping`。对于 RDS for PostgreSQL 版本 12 及更旧版本，此事件命名为 `LWLock:buffer_mapping`。

主题

支持的引擎版本

此等待事件信息与 RDS for PostgreSQL 版本 9.6 及更高版本相关。

上下文

共享缓冲池是一个 PostgreSQL 内存区域，它包含进程现在正在使用或过去使用的所有页面。当进程需要页面时，它会将页面读入共享缓冲池中。`shared_buffers` 参数会设置共享缓冲区大小并保留一个内存区域来存储表和索引页。如果更改此参数，请确保重新启动数据库。

以下情况下回发生 `LWLock:buffer_mapping` 等待事件：

- 进程在缓冲区表中搜索页面并获取共享缓冲区映射锁。
- 进程将页面加载到缓冲池中并获取独占缓冲区映射锁。
- 进程从缓冲池中删除页面并获取独占缓冲区映射锁。

原因

当此事件发生超过正常时（可能表示性能问题），数据库正在共享缓冲池中移入和移出分页。典型的原因包括：

- 大型查询
- 臃肿的索引和表
- 完整的表扫描
- 小于工作集的共享池大小

操作

根据等待事件的原因，我们建议采取不同的操作。

主题

- 监控缓冲区相关的指标
- 评估您的索引策略
- 减少必须快速分配的缓冲区数量

监控缓冲区相关的指标

当 LWLock:buffer_mapping 等待激增时，调查缓冲区命中率。您可以使用这些指标更好地了解缓冲区缓存中发生的情况。检查以下指标：

blks_hit

此性能详情计数器指标表示从共享缓冲池中检索的数据块的数量。当 LWLock:buffer_mapping 等待事件出现后，您可能会观察到 blks_hit 激增。

blks_read

此性能详情计数器指标表示需要将输入/输出读入共享缓冲池的数据块的数量。您可能会在 LWLock:buffer_mapping 等待事件的前面观察到 blks_read 激增。

评估您的索引策略

要确认您的索引策略不会降低性能，请检查以下各项：

索引膨胀

确保索引和表膨胀不会导致不必要的分页被读入共享缓冲区。如果表中包含未使用的行，请考虑存档数据并从表中删除这些行。然后，您可以为调整大小的表重建索引。

常用查询的索引

要确定您是否拥有最佳索引，请在性能详情中监控数据库引擎指标。`tup_returned` 指标显示读取的行数。`tup_fetched` 指标显示返回到客户端的行数量。如果 `tup_returned` 明显大于 `tup_fetched`，可能无法正确编制数据索引。此外，您的表统计数据可能不是最新的。

减少必须快速分配的缓冲区数量

要减少 `LWLock:buffer_mapping` 等待事件，请尝试减少必须快速分配的缓冲区数量。一种策略是执行较小的批处理操作。通过对表进行分区，也许能够实现较小的批处理。

LWLock:BufferIO (IPC:BufferIO)

当 RDS for PostgreSQL 正在等待其他进程在同时尝试访问页面的情况下完成输入/输出（I/O）操作时，会发生 `LWLock:BufferIO` 事件。它的目的是将同一个分页读入共享缓冲区中。

主题

- [相关引擎版本]
- [上下文]
- [原因]
- [操作]

相关引擎版本

此等待事件信息与所有 RDS for PostgreSQL 版本相关。对于 RDS for PostgreSQL 12 及更早版本，此等待事件命名为 lwlock:buffer_io，而在 RDS for PostgreSQL 13 版本中，它命名为 lwlock:bufferio。从 RDS for PostgreSQL 14 版本开始，BufferIO 等待事件从 `LWLock` 移到 `IPC` 等待事件类型（IPC:BufferIO）。

上下文

每个共享缓冲区都有一个与 `LWLock:BufferIO` 等待事件相关的输入/输出锁，每次必须在共享缓冲池外检索数据块（或分页）。

此锁定用于处理多个会话，而这些会话都需要访问同一个数据块。必须从共享缓冲池外部读取此数据库块，该缓冲池由 `shared_buffers` 参数定义。

一旦在共享缓冲池内读取分页，`LWLock:BufferIO` 锁即被释放。

> **Note**
> `LWLock:BufferIO` 等待事件发生在 IO:DataFileRead 等待事件之前。`IO:DataFileRead` 事件在从存储中读取数据时发生。

有关轻量级锁定的更多信息，请参阅锁定概览。

原因

`LWLock:BufferIO` 显示在主要等待中的常见原因包括以下各项：

- 多个后端或连接试图访问同样在等待输入/输出操作的同一页面
- 共享缓冲池大小之间的比率（由 `shared_buffers` 参数定义）以及当前工作负载所需的缓冲区数量
- 共享缓冲池的大小与其他操作移出的分页数量没有很好地平衡
- 需要引擎在共享缓冲池中读取更多页面的臃肿的大索引
- 缺乏强制数据库引擎从表中读取更多页面的索引
- 检查点发生太频繁或需要刷新太多修改过的页面
- 试图在同一页面上执行操作的数据库连接突增

操作

根据等待事件的原因，我们建议采取不同的操作：

- 观察 Amazon CloudWatch 指标，了解 `BufferCacheHitRatio` 突然减少和 `LWLock:BufferIO` 等待事件之间的关系。此影响可能意味着您有一个较小的共享缓冲区设置。您可能需要增加数据库实例类或对其进行纵向扩展。您可以将工作负载拆分为更多的读取器节点。
- 如果您发现 `LWLock:BufferIO` 与 `BufferCacheHitRatio` 指标降低相一致，请根据您的工作负载峰值时间优化 `max_wal_size` 和 `checkpoint_timeout`。然后确定哪个查询可能会导致发生此情况。
- 验证是否有未使用的索引，然后将其删除。
- 使用分区表（也具有分区索引）。这样做有助于保持较低的指数重新排序并降低其影响。

- 避免对列进行不必要的索引编制。
- 使用连接池防止数据库连接突增。
- 作为最佳实践，限制与数据库的最大连接数。

LWLock:buffer_content (BufferContent)

当某个会话等待读取或写入内存中的某个数据页面，而另一个会话正锁定该页面以进行写入时，会发生 `LWLock:buffer_content` 事件。在 RDS for PostgreSQL 13 及更高版本中，此等待事件称为 `BufferContent`。

主题

- 支持的引擎版本
- 上下文
- 等待次数增加的可能原因
- 操作

支持的引擎版本

RDS for PostgreSQL 的所有版本均支持此等待事件信息。

上下文

要读取或操作数据，PostgreSQL 会通过共享内存缓冲区访问数据。要从缓冲区读取，进程会在共享模式下获取缓冲区内容的轻量级锁 (LWLock)。要写入缓冲区，它会在独占模式下获得该锁。共享锁允许其他进程同时获取对该内容的共享锁。独占锁可防止其他进程获取对该内容的任何类型的锁定。

`LWLock:buffer_content` (`BufferContent`) 事件表示多个进程试图获取对特定缓冲区的内容的锁定。

等待次数增加的可能原因

当 `LWLock:buffer_content` (`BufferContent`) 事件的发生率超过正常（可能表示性能问题）时，典型原因包括以下几点：

增加了对同一数据的并发更新

> 更新相同缓冲区内容的查询的并发会话数可能会增加。在具有大量索引的表中，这种争用可能更加明显。

工作负载数据不在内存中

当活动工作负载正在处理的数据不在内存中时，这些等待事件可能会增加。这种影响是因为持有锁的进程可以在执行磁盘输入/输出操作时保持更长时间。

过度使用外键约束

外键约束可能会增加进程在缓冲区内容锁上保留的时间。这种影响是因为读取操作需要在更新引用的键时对该键进行共享缓冲区内容锁定。

操作

根据等待事件的原因，我们建议采取不同的操作。您可以通过使用 Amazon RDS 性能详情或查询视图 `pg_stat_activity` 来识别 `LWLock:buffer_content` (BufferContent) 事件。

主题

- [提高内存中的效率]
- [减少对外键约束的使用]
- [删除未使用的索引]
- [使用序列时增加缓存大小]

提高内存中的效率

为了增加活动工作负载数据在内存中的可能性，请对表进行分区或纵向扩展您的实例类。有关数据库实例类的信息，请参阅 [数据库实例类]。

减少对外键约束的使用

调查在使用外键约束时遇到大量`LWLock:buffer_content` (BufferContent) 等待事件的工作负载。删除不必要的外键约束。

删除未使用的索引

对于遇到大量 `LWLock:buffer_content`(BufferContent) 等待事件的工作负载，识别未使用的索引并删除它们。

使用序列时增加缓存大小

如果您的表使用序列，请增加缓存大小以消除序列页和索引页上的争用。每个序列都是共享内存中的单个页面。预定义缓存是对应于每个连接的。当许多并发会话获取序列值时，这可能不足以处理工作负载。

LWLock:lock_manager (LWLock:lockmanager)

当 RDS for PostgreSQL 引擎维护共享锁的内存区域以便在无法使用快速路径锁时分配、检查和取消分配锁时，会发生此事件。

主题

支持的引擎版本

此等待事件信息与 RDS for PostgreSQL 版本 9.6 及更高版本相关。对于早于版本 13 的 RDS for PostgreSQL 发行版，此等待事件的名称为 `LWLock:lock_manager`。对于 RDS for PostgreSQL 版本 13 及更高版本，此等待事件的名称为 `LWLock:lockmanager`。

上下文

发布 SQL 语句时，RDS for PostgreSQL 会记录锁，以在并发操作期间保护数据库的结构、数据和完整性。引擎可以使用快速路径锁或不快的路径锁来实现这个目标。不快的路径锁定更昂贵，并且比快速路径锁定造成的开销更大。

快速路径锁定

为了减少频繁获取和释放但很少发生冲突的锁的开销，后端进程可以使用快速路径锁定。数据库对满足以下条件的锁定使用此机制：

- 他们使用 DEFAULT 锁定方法。
- 它们代表数据库关系的锁，而不是共享关系。
- 它们是不太可能发生冲突的弱锁。
- 引擎可以快速确认不可能存在发生冲突的锁。

在以下任一条件为真时，引擎无法使用快速路径锁定：

- 锁不满足上述标准。
- 没有更多的插槽可用于后端进程。

要优化查询以实现快速路径锁，可以使用以下查询。

```
SELECT count(*), pid, mode, fastpath
  FROM pg_locks
 WHERE fastpath IS NOT NULL
 GROUP BY 4,3,2
 ORDER BY pid, mode;
 count | pid  |      mode       | fastpath
-------+------+-----------------+----------
16 | 9185 | AccessShareLock | t
336 | 9185 | AccessShareLock | f
1 | 9185 | ExclusiveLock   | t
```

以下查询仅显示数据库中的总数。

```
SELECT count(*), mode, fastpath
  FROM pg_locks
 WHERE fastpath IS NOT NULL
 GROUP BY 3,2
 ORDER BY mode,1;
count |      mode       | fastpath
-------+-----------------+----------
16 | AccessShareLock | t
337 | AccessShareLock | f
1 | ExclusiveLock   | t
(3 rows)
```

有关快速路径锁定的更多信息，请参阅 PostgreSQL 锁管理器 README 中的快速路径和 PostgreSQL 文档中的 pg-locks。

锁管理器的扩缩问题示例

在此示例中，名为 purchases 的表存储五年的数据，按天进行分区。每个分区有两个索引。将发生以下一系列事件：

1. 您查询了许多天的数据，这需要数据库读取许多分区。
2. 数据库为每个分区创建一个锁条目。如果分区索引是优化程序访问路径的一部分，则数据库也会为它们创建一个锁条目。
3. 当同一后端进程请求的锁条目数大于 16 时（这是 FP_LOCK_SLOTS_PER_BACKEND 的值），锁管理器会使用非快速路径锁定方法。

现代应用程序可能有数百个会话。如果并发会话在没有适当的分区修剪的情况下查询父级数据库，则数据库可能会创建数百甚至数千个非快速路径锁。通常，当此并发性高于 vCPU 的数量时，会出现 `LWLock:lock_manager` 等待事件。

> **Note**
>
> `LWLock:lock_manager` 等待事件与数据库架构中的分区或索引数量无关。相反，它与数据库必须控制的非快速路径锁的数量有关。

等待次数增加的可能原因

当 `LWLock:lock_manager` 等待事件发生率超出正常（可能表明性能问题）时，突增的最可能原因如下：

- 并发活动会话正在运行不使用快速路径锁的查询。这些会话还超出了最大 vCPU。
- 大量并发活动会话正在访问严重分区的表。每个分区都有多个索引。
- 数据库正在经历连接风暴。预设情况下，当数据库缓慢时，某些应用程序和连接池软件会创建更多连接。这种做法使问题变得更糟。优化连接池软件，以免发生连接风暴。
- 大量会话在不修剪分区的情况下查询父级表。
- 数据定义语言 (DDL)、数据操作语言 (DML) 或维护命令专门锁定经常访问或修改的繁忙关系或元组。

操作

如果 CPU 等待事件发生，它不一定表示性能问题。仅当性能下降并且此等待事件主导了数据库负载时才响应此事件。

主题

使用分区修剪

分区修剪是一种适用于声明性分区表的查询优化策略，它从表扫描中排除不需要的分区，从而提高性能。预设情况下，分区修剪处于开启状态。如果它已关闭，请按如下方式将其打开。

```
SET enable_partition_pruning = on;
```

查询可以在其 `WHERE` 子句包含用于分区的列时利用分区修建。有关更多信息，请参阅 PostgreSQL 文档中的分区修建。

删除不必要的索引

数据库可能包含未使用或很少使用的索引。如果是，请考虑删除它们。请执行以下任一操作：

- 通过阅读 PostgreSQL wiki 中的未使用索引了解如何查找不必要的索引。
- 运行 PG 收集器。此 SQL 脚本会收集数据库信息并将其显示在整合的 HTML 报告中。检查“未使用的索引”部分。有关更多信息，请参阅 Amazon Labs GitHub 存储库中的 pg-collector。

优化查询以实现快速路径锁定

要了解您的查询是否使用快速路径锁定，请查询 `pg_locks` 表中的 `fastpath` 列。如果您的查询没有使用快速路径锁定，请尝试将每个查询的关系数减少到 16 个以下。

优化其他等待事件

如果 `LWLock:lock_manager` 排在主要等待列表中的第一个或第二个，请检查列表中是否也显示了以下等待事件：

- `Lock:Relation`
- `Lock:transactionid`
- `Lock:tuple`

如果前面的事件显示在列表的前面，请考虑首先优化这些等待事件。这些事件可以是 `LWLock:lock_manager` 的驱动因素。

减少硬件瓶颈

您可能存在硬件瓶颈，例如 CPU 匮乏或 Amazon EBS 带宽的最大使用率。在这样的情况下，需考虑减少硬件瓶颈。请考虑以下操作：

- 纵向扩展您的实例类。
- 优化占用大量 CPU 和内存的查询。
- 更改应用程序逻辑。
- 将您的数据存档。

有关 CPU、内存和 EBS 网络带宽的更多信息，请参阅 Amazon RDS 实例类型。

使用连接池程序。

如果您的活动连接总数超过最大 vCPU，则需要 CPU 的操作系统进程超过实例类型所能支持的数量。在这种情况下，需考虑使用或优化连接池。有关您的实例类型 vCPU 数量的更多信息，请参阅 Amazon RDS 实例类型。

有关为连接池的更多信息，请参阅以下资源：

- 的 Amazon RDS 代理
- pgbouncer
- PostgreSQL 文档中的连接池和数据源

升级您的 RDS for PostgreSQL 版本

如果您当前的 RDS for PostgreSQL 版本低于 12，请升级到版本 12 或更高版本。PostgreSQL 版本 12 及更高版本具有改进的分区机制。有关版本 12 的更多信息，请参阅 PostgreSQL 12.0 发布说明。有关升级 RDS for PostgreSQL 的更多信息，请参阅 升级 RDS for PostgreSQL 数据库引擎。

Timeout:PgSleep

当服务器进程调用 `pg_sleep` 函数并且等待睡眠超时过期时，会发生 `Timeout:PgSleep` 事件。

主题

- 支持的引擎版本
- 等待次数增加的可能原因
- 操作

支持的引擎版本

RDS for PostgreSQL 的所有版本均支持此等待事件信息。

等待次数增加的可能原因

当应用程序、存储函数或用户发出调用以下函数之一的 SQL 语句时，会发生此等待事件：

- `pg_sleep`
- `pg_sleep_for`
- `pg_sleep_until`

前面的函数会延迟执行，直到经过指定的秒数为止。例如，`SELECT pg_sleep(1)` 暂停 1 秒。有关更多信息，请参阅 PostgreSQL 文档中的延迟执行。

操作

确定正在运行 `pg_sleep` 函数的语句。确定使用该功能是否合适。

Timeout:VacuumDelay

`Timeout:VacuumDelay` 事件表明已超过 vacuum 输入输出的成本限制，vacuum 进程已进入休眠状态。vacuum 操作在相应的成本延迟参数中指定的持续时间内停止，然后恢复工作。对于手动 vacuum 命令，延迟在 `vacuum_cost_delay` 参数中指定。对于 autovacuum 进程守护程序，延迟在 `autovacuum_vacuum_cost_delay parameter.` 中指定。

主题

- 支持的引擎版本
- 上下文
- 等待次数增加的可能原因
- 操作

支持的引擎版本

RDS for PostgreSQL 的所有版本均支持此等待事件信息。

上下文

PostgreSQL 既有 autovacuum 进程守护程序，也有手动 vacuum 命令。RDS for PostgreSQL 数据库实例的 autovacuum 进程原定设置为“on”。手动 vacuum 命令可根据需要使用，例如，用于清除不活动元组表或生成新的统计数据。

在执行 vacuum 操作时，PostgreSQL 在系统执行各种输入输出操作时使用内部计数器来跟踪估计成本。当计数器达到由成本限制参数指定的值时，执行操作的进程将在成本延迟参数中指定的短暂持续时间内休眠。然后，它重置计数器并继续操作。

vacuum 进程具有可用于调节资源消耗的参数。autovacuum 和手动 vacuum 命令有自己的参数来设置成本限制值。它们也有自己的参数来指定成本延迟，即达到极限时让 vacuum 进入休眠状态的时间量。通过这种方式，成本延迟参数可以作为资源消耗的限制机制。在下面的列表中，您可以找到这些参数的说明。

影响 autovacuum 进程守护程序限制的参数

- `autovacuum_vacuum_cost_limit` – 指定要在自动 vacuum 操作中使用的成本限制值。增加此参数的设置可使 Vacuum 进程使用更多资源并减少 `Timeout:VacuumDelay` 等待事件。
- `autovacuum_vacuum_cost_delay` – 指定要在自动 vacuum 操作中使用的成本延迟值。原定设置值为 2 毫秒。将延迟参数设置为 0 会关闭限制机制，因此，不会出现 `Timeout:VacuumDelay` 等待事件。

有关更多信息，请参阅 PostgreSQL 文档中的 Automatic Vacuuming。

影响限制手动 vacuum 进程的参数

- `vacuum_cost_limit` – vacuum 进程进入休眠状态的阈值。原定设置情况下，此限制为 200。此数字表示各种资源所需的额外输入输出的累计成本估计值。增加此值会减少 `Timeout:VacuumDelay` 等待事件的数量。
- `vacuum_cost_delay` – 达到 vacuum 成本限制时 vacuum 进程休眠的时间量。原定设置为 0，这意味着此功能关闭。您可以将其设置为整数值以指定开启此功能的毫秒数，但我们建议您将其保留为原定设置。

有关 `vacuum_cost_delay` 参数的更多信息，请参阅 PostgreSQL 文档中的资源消耗量。

要了解有关如何在 RDS for PostgreSQL 中配置和使用 autovacuum 的更多信息，请参阅 在 Amazon RDS for PostgreSQL 上使用 PostgreSQL autovacuum。

等待次数增加的可能原因

`Timeout:VacuumDelay` 受成本限制参数设置（`vacuum_cost_limit`、`autovacuum_vacuum_cost_limit`）与控制 vacuum 的休眠持续时间的成本延迟参数（`vacuum_cost_delay`、`autovacuum_vacuum_cost_delay`）之间平

衡的影响。提高成本限制参数值允许 vacuum 在进入休眠状态之前使用更多资源。这样可以减少 `Timeout:VacuumDelay` 等待事件。增加任一延迟参数都会使 `Timeout:VacuumDelay` 等待事件发生的频率更高且持续的时间更长。

`autovacuum_max_workers` 参数设置还可以增加 `Timeout:VacuumDelay` 的数量。每个额外的 autovacuum 工作进程都有助于内部计数器机制，因此，与单个 autovacuum 工作进程相比，可以更快地达到限制。随着更快地达到成本限制，成本延迟会更频繁地生效，从而导致更多的 `Timeout:VacuumDelay` 等待事件。有关更多信息，请参阅 PostgreSQL 文档中的 autovacuum_max_workers。

大型对象（如 500GB 或更大）也会引发此等待事件，因为 vacuum 可能需要一些时间才能完成处理大型对象。

操作

如果 vacuum 操作按预期完成，则无需补救。换句话说，此等待事件不一定表示有问题。它表示 vacuum 在延迟参数中指定的时间段内进入休眠状态，以便可以将资源应用于需要完成的其他进程。

如果您希望更快地完成 vacuum 操作，可以降低延迟参数。这可缩短 vacuum 休眠的时间。

使用 Amazon DevOps Guru 主动见解优化 RDS for PostgreSQL

DevOps Guru 主动见解可检测 RDS for PostgreSQL 数据库实例上可能导致问题的情况，并在问题发生之前告知您。主动的洞察可以提醒您具有长时间运行的事务空闲连接。有关排查长时间运行的事务空闲连接的更多信息，请参阅数据库具有长时间运行的事务空闲连接。

DevOps Guru 可以执行以下操作：

- 通过对照常见的建议设置交叉检查数据库配置，可以防止许多常见的数据库问题。
- 提醒您注意实例集中的关键问题，如果不加以检查，以后可能会导致更大的问题。
- 提醒您注意新发现的问题。

每项主动见解都包含对问题原因的分析和纠正措施建议。

有关适用于 Amazon RDS 的 Amazon DevOps Guru 的更多信息，请参阅使用适用于 Amazon RDS 的 Amazon DevOps Guru 分析性能异常。

数据库具有长时间运行的事务空闲连接

与数据库的连接处于 `idle in transaction` 状态的时间已超过 1800 秒。

主题

- 支持的引擎版本
- 上下文
- 这个问题的可能原因
- 操作
- 相关指标

支持的引擎版本

RDS for PostgreSQL 的所有版本都支持这些见解信息。

上下文

处于 `idle in transaction` 状态的事务可能持有旨在阻止其他查询的锁。它还可以防止 `VACUUM`（包括 autovaum）清理死行，从而导致索引或表膨胀或事务 ID 重叠。

这个问题的可能原因

在使用 BEGIN 或 START TRANSATION 的交互式会话中启动的事务尚未通过使用 COMMIT、ROLLBACK 或 END 命令结束。这会导致事务移至 `idle in transaction` 状态。

操作

您可以通过查询 `pg_stat_activity` 来查找空闲的事务。

在 SQL 客户端中，运行以下查询以列出所有处于 `idle in transaction` 状态的连接，并按持续时间对它们进行排序：

```
SELECT now() - state_change as idle_in_transaction_duration, now() - xact_start as
 xact_duration,*
FROM  pg_stat_activity
WHERE state  = 'idle in transaction'
AND   xact_start is not null
ORDER BY 1 DESC;
```

根据见解的原因，我们建议采取不同的操作。

主题

- 结束事务
- 终止连接
- 配置 idle_in_transaction_session_timeout 参数
- 检查 AUTOCOMMIT 状态
- 检查应用程序代码中的事务逻辑

结束事务

当您在交互式会话中使用 BEGIN 或 START TRANSACTION 启动事务时，它会转入 `idle in transaction` 状态。它会一直保持此状态，直到您通过发出 COMMIT、ROLLBACK、END 命令结束事务，或完全断开连接以回滚事务。

终止连接

使用以下查询终止与空闲事务的连接：

```
SELECT pg_terminate_backend(pid);
```

pid 是连接的进程 ID。

配置 idle_in_transaction_session_timeout 参数

在参数组中配置 `idle_in_transaction_session_timeout` 参数。配置此参数的优势在于，它不需要手动干预即可终止长时间空闲的事务。有关此参数的更多信息，请参阅 PostgreSQL 文档。

当事务处于 idle_in_transaction 状态超过指定时间时，连接终止后，PostgreSQL 日志文件中将报告以下消息。

```
FATAL: terminating connection due to idle in transaction timeout
```

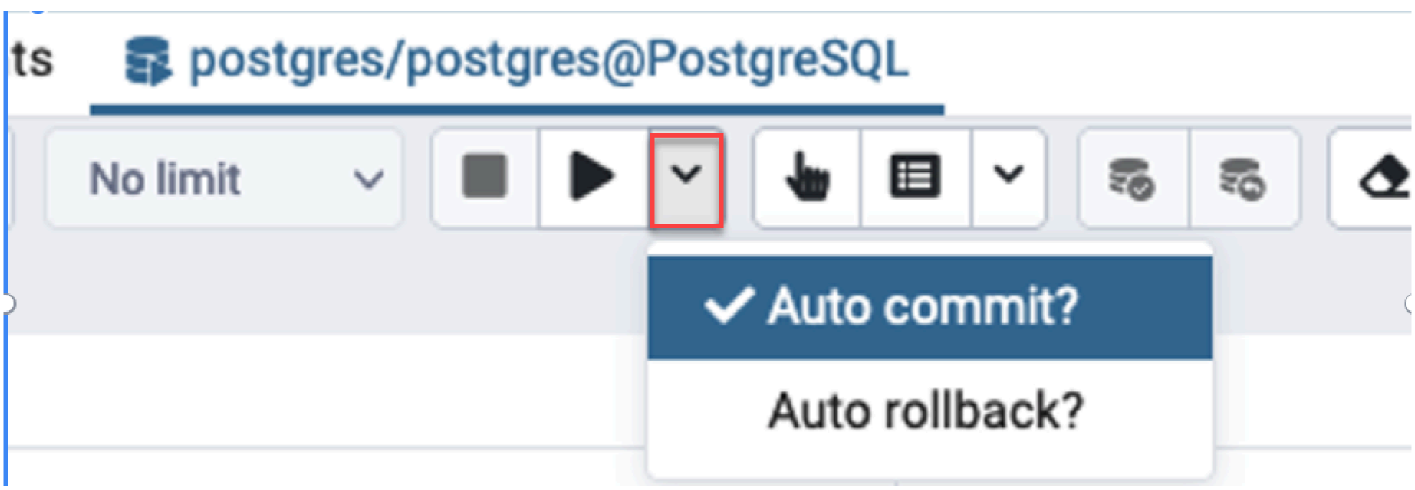
检查 AUTOCOMMIT 状态

原定设置情况下，AUTOCOMMIT 处于开启状态。但是，如果在客户端中意外将其关闭，请确保将其重新开启。

- 在 psql 客户端中，运行以下命令：

```
postgres=> \set AUTOCOMMIT on
```

- 在 pgadmin 中，通过从向下箭头中选择 AUTOCOMMIT 选项将其开启。



检查应用程序代码中的事务逻辑

调查应用程序逻辑中可能存在的问题。请考虑以下操作：

- 检查应用程序中 JDBC 自动提交是否设置为 true。另外，可以考虑在代码中使用显式 `COMMIT` 命令。
- 检查错误处理逻辑，看看它是否会在错误后关闭事务。
- 在事务打开时，检查您的应用程序处理查询所返回的行是否花费了很长时间。如果是，请考虑在处理行之前对应用程序进行编码以关闭事务。
- 检查事务是否包含许多长时间运行的操作。如果是，请将单个事务分成多个事务。

相关指标

以下 PI 指标与此见解相关：

- idle_in_transaction_count - 处于 `idle in transaction` 状态的会话数。
- idle_in_transaction_max_time - 处于 `idle in transaction` 状态的运行时间最长的事务的持续时间。

将 PostgreSQL 扩展与 Amazon RDS for PostgreSQL 结合使用

您可以通过安装各种扩展和模块来扩展 PostgreSQL 的功能。例如，要处理空间数据，您可以安装和使用 PostGIS 扩展。有关更多信息，请参阅 使用 PostGIS 扩展管理空间数据。作为另一个例子，如果要改进非常大的表的数据输入，您可以考虑使用 `pg_partman` 扩展对数据进行分区。要了解更多信息，请参阅使用 pg_partman 扩展管理 PostgreSQL 分区。

> **Note**
>
> RDS for PostgreSQL 通过 `pg_tle` 扩展支持 PostgreSQL 的可信语言扩展，您可以将该扩展添加到数据库实例中。通过使用此扩展，开发人员可以在安全的环境中创建自己的 PostgreSQL 扩展，从而简化设置和配置要求。要了解支持 `pg_tle` 扩展的 RDS for PostgreSQL 版本以及更多信息，请参阅使用适用于 PostgreSQL 的可信语言扩展。

在某些情况下，您可以将特定模块添加到 RDS for PostgreSQL 数据库实例的自定义数据库参数组中的 `shared_preload_libraries` 列表中，而不是安装扩展。通常，默认的数据库集群参数组仅加载 `pg_stat_statements`，但还有其他几个模块可供添加到此列表中。例如，您可以通过添加 `pg_cron` 模块来添加调度功能，详情请见使用 PostgreSQL pg_cron 扩展计划维护。再举一个例子，您可以通过加载 `auto_explain` 模块来记录查询执行计划。要了解更多信息，请参阅 Amazon 知识中心中的记录查询执行计划。

根据您的 RDS for PostgreSQL 版本，安装扩展可能需要 `rds_superuser` 权限，如下所示：

- 对于 RDS for PostgreSQL 版本 12 和更早版本，安装扩展需要 `rds_superuser` 权限。
- 对于 RDS for PostgreSQL 版本 13 及更高版本，对给定数据库实例具有创建权限的用户（角色）可以安装和使用任何可信扩展。有关可信扩展清单，请参阅PostgreSQL 可信扩展。

您还可以通过在 `rds.allowed_extensions` 参数中列出扩展，来精确指定可以在 RDS for PostgreSQL 数据库实例上安装的扩展。有关更多信息，请参阅 限制 PostgreSQL 扩展的安装。

要了解有关 `rds_superuser` 角色的更多信息，请参阅了解 PostgreSQL 角色和权限。

主题

- 使用 pgAudit 记录数据库活动
- 使用 PostgreSQL pg_cron 扩展计划维护
- 使用 pglogical 跨实例同步数据
- 使用 pgactive 支持主动-主动复制
- 使用 pg_repack 扩展减少表和索引的膨胀
- 升级和使用 PLV8 扩展
- 使用 PL/Rust 通过 Rust 语言编写 PostgreSQL 函数
- 使用 PostGIS 扩展管理空间数据

使用 orafce 扩展中的函数

orafce 扩展提供了模拟 Oracle 数据库中的函数和软件包子集的函数和运算符。orafce 扩展使您能够更轻松地将 Oracle 应用程序移植到 PostgreSQL。RDS for PostgreSQL 版本 9.6.6 及更高版本支持该扩展。有关 orafce 的更多信息，请参阅 GitHub 上的 orafce。

> **Note**
>
> RDS for PostgreSQL 不支持属于 orafce 扩展一部分的 `utl_file` 软件包。这是因为 `utl_file` 架构函数可提供对操作系统文本文件的读取和写入操作，这需要对底层主机拥有超级用户访问权限。作为一项托管服务，RDS for PostgreSQL 不提供主机访问权限。

使用 orafce 扩展

1. 使用用于创建数据库实例的主用户名连接到数据库实例。

   如果要为同一数据库实例中的不同数据库启用 orafce，请使用 `/c dbname` psql 命令。使用此命令，您可以在启动连接后从主数据库进行更改。
2. 使用 `CREATE EXTENSION` 语句启用 orafce 扩展。

   ```
   CREATE EXTENSION orafce;
   ```
3. 使用 `ALTER SCHEMA` 语句将 oracle 架构的所有权移交给 rds_superuser 角色。

   ```
   ALTER SCHEMA oracle OWNER TO rds_superuser;
   ```

如果要查看 oracle 架构的所有者列表，请使用 \dn psql 命令。

使用适用于 PostgreSQL 的 Amazon RDS 委派扩展支持

使用适用于 PostgreSQL 的 Amazon RDS 委派扩展支持，您可以将扩展管理委派给其它用户，而且不必让该用户成为 `rds_superuser`。通过这种委派扩展支持，将创建一个名为 `rds_extension` 的新角色，您必须将其分配给用户才能管理其他扩展。此角色可以创建、更新和删除扩展。

可以通过在 `rds.allowed_extensions` 参数中列出扩展，指定可以在 RDS 数据库实例上安装哪些扩展。有关更多信息，请参阅将 PostgreSQL 扩展与 Amazon RDS for PostgreSQL 结合使用。

您可以通过 `rds.allowed_delegated_extensions` 参数来限制具有 `rds_extension` 角色的用户可以管理的可用扩展列表。

委派扩展支持在以下版本中可用：

- 所有更高版本
- 16.4 及更高的 16 版本
- 15.8 及更高的 15 版本
- 14.13 及更高的 14 版本
- 13.16 及更高的 13 版本
- 12.20 及更高的 12 版本

主题

- 为用户开启委派扩展支持
- 在适用于 PostgreSQL 的 RDS 委派扩展支持中使用的配置
- 关闭对委派扩展的支持
- 使用 Amazon RDS 委派扩展支持的优势
- 适用于 PostgreSQL 的 Amazon RDS 委派扩展支持的限制
- 某些扩展所需的权限
- 安全考虑因素
- 已禁用删除扩展级联
- 可以使用委派扩展支持添加的扩展示例

为用户开启委派扩展支持

必须执行以下操作才能为用户启用委派扩展支持：

1. 向用户授予 **rds_extension** 角色：以 rds_superuser 身份连接到数据库并执行以下命令：

```
Postgres => grant rds_extension to user_name;
```

2. 设置可供委派用户管理的扩展列表：rds.allowed_delegated_extensions 允许您使用 rds.allowed_extensions 在数据库集群参数中指定可用扩展的子集。您可以在以下级别之一执行此操作：

   - 在集群或实例参数组中，通过 Amazon Web Services Management Console或 API。有关更多信息，请参阅 Amazon RDS 的参数组。
   - 在数据库级别使用以下命令：

```
alter database database_name set rds.allowed_delegated_extensions =
 'extension_name_1,
                    extension_name_2,...extension_name_n';
```

   - 在用户级别使用以下命令：

```
alter user user_name set rds.allowed_delegated_extensions = 'extension_name_1,
                    extension_name_2,...extension_name_n';
```

> **Note**
>
> 更改 rds.allowed_delegated_extensions 动态参数后，无需重启数据库。

3. 允许委派用户访问在扩展创建过程中创建的对象：某些扩展创建的对象要求先授予其它权限，然后具有 rds_extension 角色的用户才能访问它们。rds_superuser 必须向委派用户授予对这些对象的访问权限。其中一个选项是使用事件触发器自动向委派用户授予权限。有关更多信息，请参阅关闭对委派扩展的支持中的事件触发器示例。

在适用于 PostgreSQL 的 RDS 委派扩展支持中使用的配置

配置名称	描述	默认值	注意	谁可以修改或授予权限
rds.a ed_de	此参数限制 rds_extension 角色	空字符串	• 默认情况下，此参数为空字符串，	rds_superuser

配置名称	描述	默认值	注意	谁可以修改或授予权限
ted_extensions	可以在数据库中管理的扩展。它必须是 rds.allowed_extensions 的子集。		这意味着没有向具有 `rds_extension` 的用户委派任何扩展。 • 如果用户有添加受支持的扩展的权限，则可以添加此扩展。为此，请将 `rds.allowed_delegated_extensions` 参数设置为一串以逗号分隔的扩展名称。通过向该参数添加扩展列表，可以显式确定具有 `rds_extension` 角色的用户可以安装的扩展。 • 如果设置为 `*`，则表示 `rds_allowed_extensions` 中列出的所有扩展都将委派给具有 `rds_extension` 角色的用户。 要了解有关设置此参数的更多信息，请参	

配置名称	描述	默认值	注意	谁可以修改或授予权限
			阅为用户开启委派扩展支持。	
`rds.a ed_ex ions`	此参数可让客户限制可以在 RDS 数据库实例中安装的扩展。有关更多信息，请参阅限制 PostgreSQL 扩展的安装。	"*"	默认情况下，此参数设置为“*”，这意味着具有必要权限的用户能够创建 RDS for PostgreSQL 和 Aurora PostgreSQL 支持的所有扩展。 空表示无法在 RDS 数据库实例中安装任何扩展。	管理员
`rds- deleg ated_ nsion ow_dr ascac`	此参数控制拥有 `rds_extension` 的用户使用级联选项删除扩展的能力。	off	默认情况下，将 `rds-delegated_extension_allow_drop_cascade` 设置为 `off`。这意味着不允许拥有 `rds_extension` 的用户使用级联选项删除扩展。 要授予该能力，`rds.delegated_extension_allow_drop_cascade` 参数应设置为 `on`。	rds_superuser

关闭对委派扩展的支持

部分关闭

委派用户无法创建新的扩展，但仍然可以更新现有的扩展。

- 在数据库集群参数组中将 `rds.allowed_delegated_extensions` 重置为默认值。
- 在数据库级别使用以下命令：

```
alter database database_name reset rds.allowed_delegated_extensions;
```

- 在用户级别使用以下命令：

```
alter user user_name reset rds.allowed_delegated_extensions;
```

完全关闭

撤销用户的 `rds_extension` 角色会将该用户恢复为标准权限。用户无法再创建、更新或删除扩展。

```
postgres => revoke rds_extension from user_name;
```

事件触发器示例

如果您希望允许具有 `rds_extension` 的委派用户使用扩展，而此类扩展要求对通过扩展创建过程所创建的对象设置权限，则可以自定义以下事件触发器示例，并仅添加您希望委派用户有权访问其全部功能的扩展。此事件触发器可以在 template1（默认模板）上创建，因此所有从 template1 创建的数据库都将具有该事件触发器。当委派用户安装扩展时，此触发器将自动授予对扩展创建的对象的所有权。

```

CREATE OR REPLACE FUNCTION create_ext()

  RETURNS event_trigger AS $$

DECLARE

  schemaname TEXT;
  databaseowner TEXT;

  r RECORD;

BEGIN
```

```
 IF tg_tag = 'CREATE EXTENSION' and current_user != 'rds_superuser' THEN
   RAISE NOTICE 'SECURITY INVOKER';
   RAISE NOTICE 'user: %', current_user;
   FOR r IN SELECT * FROM pg_event_trigger_ddl_commands()
   LOOP
       CONTINUE WHEN r.command_tag != 'CREATE EXTENSION' OR r.object_type !=
'extension';

       schemaname = (
           SELECT n.nspname
           FROM pg_catalog.pg_extension AS e
           INNER JOIN pg_catalog.pg_namespace AS n
           ON e.extnamespace = n.oid
           WHERE e.oid = r.objid
       );

       databaseowner = (
           SELECT pg_catalog.pg_get_userbyid(d.datdba)
           FROM pg_catalog.pg_database d
           WHERE d.datname = current_database()
       );
       RAISE NOTICE 'Record for event trigger %, objid: %,tag: %, current_user: %,
schema: %, database_owenr: %', r.object_identity, r.objid, tg_tag, current_user,
schemaname, databaseowner;
       IF r.object_identity = 'address_standardizer_data_us' THEN
           EXECUTE format('GRANT SELECT, UPDATE, INSERT, DELETE ON TABLE %I.us_gaz TO
%I WITH GRANT OPTION;', schemaname, databaseowner);
           EXECUTE format('GRANT SELECT, UPDATE, INSERT, DELETE ON TABLE %I.us_lex TO
%I WITH GRANT OPTION;', schemaname, databaseowner);
           EXECUTE format('GRANT SELECT, UPDATE, INSERT, DELETE ON TABLE %I.us_rules
TO %I WITH GRANT OPTION;', schemaname, databaseowner);
       ELSIF r.object_identity = 'dict_int' THEN
           EXECUTE format('ALTER TEXT SEARCH DICTIONARY %I.intdict OWNER TO %I;',
schemaname, databaseowner);
       ELSIF r.object_identity = 'pg_partman' THEN
           EXECUTE format('GRANT SELECT, UPDATE, INSERT, DELETE ON TABLE
%I.part_config TO %I WITH GRANT OPTION;', schemaname, databaseowner);
           EXECUTE format('GRANT SELECT, UPDATE, INSERT, DELETE ON TABLE
%I.part_config_sub TO %I WITH GRANT OPTION;', schemaname, databaseowner);
           EXECUTE format('GRANT SELECT, UPDATE, INSERT, DELETE ON TABLE
%I.custom_time_partitions TO %I WITH GRANT OPTION;', schemaname, databaseowner);
       ELSIF r.object_identity = 'postgis_topology' THEN
```

```
            EXECUTE format('GRANT SELECT, UPDATE, INSERT, DELETE ON ALL TABLES IN
 SCHEMA topology TO %I WITH GRANT OPTION;', databaseowner);
            EXECUTE format('GRANT USAGE, SELECT ON ALL SEQUENCES IN SCHEMA topology TO
 %I WITH GRANT OPTION;', databaseowner);
            EXECUTE format('GRANT EXECUTE ON ALL FUNCTIONS IN SCHEMA topology TO %I
 WITH GRANT OPTION;', databaseowner);
            EXECUTE format('GRANT USAGE ON SCHEMA topology TO %I WITH GRANT OPTION;',
 databaseowner);
        END IF;
    END LOOP;
  END IF;
END;
$$ LANGUAGE plpgsql SECURITY DEFINER;

CREATE EVENT TRIGGER log_create_ext ON ddl_command_end EXECUTE PROCEDURE create_ext();
```

使用 Amazon RDS 委派扩展支持的优势

通过使用适用于 PostgreSQL 的 Amazon RDS 委派扩展支持，您可以安全地将扩展管理委派给没有 `rds_superuser` 角色的用户。该功能具有以下优势：

- 您可以轻松地将扩展管理委派给您选择的用户。
- 这不需要 `rds_superuser` 角色。
- 支持为同一个数据库集群中的不同数据库提供不同的扩展集。

适用于 PostgreSQL 的 Amazon RDS 委派扩展支持的限制

- 在扩展创建过程中创建的对象可能需要额外的权限才能使扩展正常运行。
- 默认情况下，委派扩展用户无法管理某些扩展，包括以下各项：`log_fdw`、`pg_cron`、`pg_tle`、`pgactive`、`pglogical`、`postgis_raster`、`postgis_tiger_`

某些扩展所需的权限

要创建、使用或更新以下扩展，委派用户应具有对以下函数、表和架构的必要权限。

需要所有权或权限的扩展	函数	表	架构	文本搜索词典	注释
addre tanda er_da s	none	us_gaz、us _lex、us_l ex、I.us_rules	none	none	none
amch	bt_index_ check、bt_ index_par ent_check	none	none	none	none
dict_i	none	none	none	intdict	none
pg_p n	none	custom_ti me_partit ions、part _config、p art_config_sub	none	none	none
pg_s taten	none	none	none	none	none
Post(	st_tileenvelope	spatial_ref_sys	none	none	none
postç aster	none	none	none	none	none

需要所有权或权限的扩展	函数	表	架构	文本搜索词典	注释
postg opolc	none	topology、layer	topology	none	委派用户必须是数据库所有者
log_f	create_fo reign_tab le_for_log_file	none	none	none	none
rds_t	role_pass word_encr yption_type	none	none	none	none
postg iger_ oder	none	geocode_s ettings_d efault、ge ocode_settings	tiger	none	none
pg_fr acem	pg_freespace	none	none	none	none
pg_v lity	pg_visibility	none	none	none	none

安全考虑因素

请记住，具有 rds_extension 角色的用户将能够管理他们拥有连接权限的所有数据库上的扩展。如果打算让委派用户管理单个数据库上的扩展，那么一个好的做法是撤销每个数据库上的公有权限，然后向委派用户显式授予对该特定数据库的连接权限。

有几个扩展可以允许用户访问多个数据库中的信息。在向 rds.allowed_delegated_extensions 添加这些扩展之前，请确保您授予 rds_extension 的用户具有跨数据库功能。例如，postgres_fdw 和 dblink 提供在同一实例或远程实例上跨数据库进行查询的功能。log_fdw 读取 postgres 引擎日志文件，这些文件适用于实例中的所有数据库，可能包含来自多个数据库的慢速查询或错误消息。pg_cron 允许在数据库实例上运行计划的后台任务，并且可以将任务配置为在不同的数据库中运行。

已禁用删除扩展级联

具有 rds_extension 角色的用户使用级联选项删除扩展的能力由 rds.delegated_extension_allow_drop_cascade 参数控制。默认情况下，将 rds-delegated_extension_allow_drop_cascade 设置为 off。这意味着不允许具有 rds_extension 角色的用户使用级联选项删除扩展，如以下查询所示。

```
DROP EXTENSION CASCADE;
```

因为这将自动删除依赖于扩展的对象，进而删除依赖于这些对象的所有对象。尝试使用级联选项将会导致错误。

要授予该能力，rds.delegated_extension_allow_drop_cascade 参数应设置为 on。

更改 rds.delegated_extension_allow_drop_cascade 动态参数不需要重启数据库。您可以在以下级别之一执行此操作：

- 在集群或实例参数组中，通过 Amazon Web Services Management Console或 API。
- 在数据库级别使用以下命令：

  ```
  alter database database_name set rds.delegated_extension_allow_drop_cascade = 'on';
  ```

- 在用户级别使用以下命令：

  ```
  alter role tenant_user set rds.delegated_extension_allow_drop_cascade = 'on';
  ```

可以使用委派扩展支持添加的扩展示例

- `rds_tools`

```
extension_test_db=> create extension rds_tools;
CREATE EXTENSION
extension_test_db=> SELECT * from rds_tools.role_password_encryption_type() where
 rolname = 'pg_read_server_files';
ERROR: permission denied for function role_password_encryption_type
```

- `amcheck`

```
extension_test_db=> CREATE TABLE amcheck_test (id int);
CREATE TABLE
extension_test_db=> INSERT INTO amcheck_test VALUES (generate_series(1,100000));
INSERT 0 100000
extension_test_db=> CREATE INDEX amcheck_test_btree_idx ON amcheck_test USING btree
 (id);
CREATE INDEX
extension_test_db=> create extension amcheck;
CREATE EXTENSION
extension_test_db=> SELECT bt_index_check('amcheck_test_btree_idx'::regclass);
ERROR: permission denied for function bt_index_check
extension_test_db=> SELECT bt_index_parent_check('amcheck_test_btree_idx'::regclass);
ERROR: permission denied for function bt_index_parent_check
```

- `pg_freespacemap`

```
extension_test_db=> create extension pg_freespacemap;
CREATE EXTENSION
extension_test_db=> SELECT * FROM pg_freespace('pg_authid');
ERROR: permission denied for function pg_freespace
extension_test_db=> SELECT * FROM pg_freespace('pg_authid',0);
ERROR: permission denied for function pg_freespace
```

- `pg_visibility`

```
extension_test_db=> create extension pg_visibility;
CREATE EXTENSION
extension_test_db=> select * from pg_visibility('pg_database'::regclass);
ERROR: permission denied for function pg_visibility
```

- `postgres_fdw`

```
extension_test_db=> create extension postgres_fdw;
CREATE EXTENSION
extension_test_db=> create server myserver foreign data wrapper postgres_fdw options
 (host 'foo', dbname 'foodb', port '5432');
ERROR: permission denied for foreign-data wrapper postgres_fdw
```

使用 pg_partman 扩展管理 PostgreSQL 分区

PostgreSQL 表分区为数据输入和报告的高性能处理提供了框架。对于需要非常快速地输入大量数据的数据库，请使用分区。分区还可以更快地查询大型表。分区有助于在不影响数据库实例的情况下维护数据，因为它需要的 I/O 资源较少。

通过使用分区，您可以将数据拆分为自定义大小的块进行处理。例如，您可以将时间序列数据分区为范围，例如每小时、每日、每周、每月、每季度、每年、自定义或这些范围的任意组合。对于时间序列数据示例，如果您按小时对表进行分区，则每个分区会包含一小时的数据。如果您按天对时间序列表进行分区，则分区会保存一天的数据，以此类推。分区键控制分区的大小。

在分区表上使用 `INSERT` 或 `UPDATE` SQL 命令时，数据库引擎会将数据路由到相应的分区。存储数据的 PostgreSQL 表分区是主表的子表。

在数据库查询读取期间，PostgreSQL 优化程序会检查查询的 `WHERE` 子句，如果可能的话，将数据库扫描定向到仅相关分区。

从版本 10 开始，PostgreSQL 使用声明性分区来实现表分区，这也称为本机 PostgreSQL 分区。在 PostgreSQL 版本 10 之前，使用触发器来实现分区。

PostgreSQL 表分区提供了以下功能：

- 随时创建新分区。
- 可变分区范围。
- 使用数据定义语言 (DDL) 语句可分离和可重新连接的分区。

  例如，可分离的分区对于从主分区中删除历史数据但保留历史数据以供分析很有用。
- 新分区继承了父数据库表的属性，包括以下属性：
  - 索引
  - 主键，其必须包括分区键列
  - 外键
  - 检查约束
  - 参考
- 为完整表或每个特定分区创建索引。

您不能更改单个分区的架构。但是，您可以更改传播到分区的父表（例如添加新列）。

主题

- PostgreSQL pg_partman 扩展概述
- 启用 pg_partman 扩展
- 使用 create_parent 函数配置分区
- 使用 run_maintenance_proc 函数配置分区维护

PostgreSQL pg_partman 扩展概述

您可以使用 PostgreSQL pg_partman 扩展自动创建和维护表分区。有关更多一般信息，请参阅 pg_partman 文档中的 PG 分区管理器。

> **Note**
>
> RDS for PostgreSQL 版本 12.5 及更高版本支持该 pg_partman 扩展。

您可以使用以下设置来配置 pg_partman，而不必手动创建每个分区：

- 要分区的表
- 分区类型
- 分区键
- 分区粒度
- 分区预创建和管理选项

创建 PostgreSQL 分区表后，您可以通过调用 `create_parent` 函数向 pg_partman 注册该表。此举会根据传递给函数的参数创建必要的分区。

该 pg_partman 扩展还提供了 `run_maintenance_proc` 函数，您可以按计划调用该函数来自动管理分区。为确保根据需要创建正确的分区，请将此函数计划为定期运行（例如每小时）。您还可以确保自动删除分区。

启用 pg_partman 扩展

如果要管理分区的同一 PostgreSQL 数据库实例中有多个数据库，请为每个数据库分别启用 pg_partman 扩展。要为特定数据库启用 pg_partman 扩展，请创建分区维护架构，然后按如下所示创建 pg_partman 扩展。

```
CREATE SCHEMA partman;
```

```
CREATE EXTENSION pg_partman WITH SCHEMA partman;
```

> **Note**
>
> 要创建 pg_partman 扩展，请确保您具有 `rds_superuser` 权限。

如果您收到以下错误，请向该账户授予 `rds_superuser` 权限或使用您的超级用户帐户。

```
ERROR: permission denied to create extension "pg_partman"
HINT: Must be superuser to create this extension.
```

要授予 `rds_superuser` 权限，请连接您的超级用户账户并运行以下命令。

```
GRANT rds_superuser TO user-or-role;
```

对于显示使用 pg_partman 扩展的示例，我们使用以下示例数据库表和分区。此数据库使用基于时间戳的分区表。架构 `data_mart` 包含名为 `events` 的表，当中包含名为 `created_at` 的列。`events` 表中包含以下设置：

- 主键 `event_id` 和 `created_at`，其必须具有用于指导分区的列。
- 用于强制 `ck_valid_operation` 表列值的检查约束 `operation`。
- 两个外键，其中一个 (`fk_orga_membership`)) 指向外部表 `organization`，另一个 (`fk_parent_event_id`) 是自引用的外键。
- 两个索引，其中一个 (`idx_org_id`) 用于外键，另一个 (`idx_event_type`) 用于事件类型。

以下 DDL 语句创建这些对象，这些对象自动包含在每个分区中。

```
CREATE SCHEMA data_mart;
CREATE TABLE data_mart.organization ( org_id BIGSERIAL,
        org_name TEXT,
        CONSTRAINT pk_organization PRIMARY KEY (org_id)
    );

CREATE TABLE data_mart.events(
        event_id        BIGSERIAL,
        operation       CHAR(1),
```

```
        value           FLOAT(24),
        parent_event_id BIGINT,
        event_type      VARCHAR(25),
        org_id          BIGSERIAL,
        created_at      timestamp,
        CONSTRAINT pk_data_mart_event PRIMARY KEY (event_id, created_at),
        CONSTRAINT ck_valid_operation CHECK (operation = 'C' OR operation = 'D'),
        CONSTRAINT fk_orga_membership
            FOREIGN KEY(org_id)
            REFERENCES data_mart.organization (org_id),
        CONSTRAINT fk_parent_event_id
            FOREIGN KEY(parent_event_id, created_at)
            REFERENCES data_mart.events (event_id,created_at)
    ) PARTITION BY RANGE (created_at);

CREATE INDEX idx_org_id     ON  data_mart.events(org_id);
CREATE INDEX idx_event_type ON  data_mart.events(event_type);
```

使用 create_parent 函数配置分区

启用 pg_partman 扩展后，使用 create_parent 函数在分区维护架构中配置分区。以下示例使用在 events 中创建的 启用 pg_partman 扩展 表示例。按如下方式调用 create_parent 函数。

```
SELECT partman.create_parent(
 p_parent_table => 'data_mart.events',
 p_control      => 'created_at',
 p_type         => 'range',
 p_interval     => '1 day',
 p_premake      => 30);
```

参数如下所示：

- p_parent_table – 父分区表。此表必须已存在并完全限定（包括架构在内）。
- p_control – 分区所依据的列。数据类型必须是整数或基于时间的。
- p_type – 该类型是 'range' 或者 'list'。
- p_interval – 每个分区的时间间隔或整数范围。示例值包括 1 day、1 hour 等。
- p_premake – 为支持新插入而提前创建的分区数量。

有关 create_parent 函数的完整说明，请参阅 pg_partman 文档中的创建函数。

使用 run_maintenance_proc 函数配置分区维护

您可以运行分区维护操作来自动创建新分区、分离分区或删除旧分区。分区维护依赖于 pg_partman 扩展和 pg_cron 扩展的 run_maintenance_proc 函数，其将启动内部调度程序。调度程序 pg_cron 自动执行数据库中定义的 SQL 语句、函数和程序。

以下示例使用在 events 中创建的 启用 pg_partman 扩展 表示例将分区维护操作设置为自动运行。作为先决条件，请将 pg_cron 添加到数据库实例的参数组中的 shared_preload_libraries 参数。

```
CREATE EXTENSION pg_cron;

UPDATE partman.part_config
SET infinite_time_partitions = true,
    retention = '3 months',
    retention_keep_table=true
WHERE parent_table = 'data_mart.events';
SELECT cron.schedule('@hourly', $$CALL partman.run_maintenance_proc()$$);
```

以下是前述示例的分步说明：

1. 修改与数据库实例关联的参数组并将 pg_cron 添加到 shared_preload_libraries 参数值中。此更改需要重启数据库实例才能生效。有关更多信息，请参阅“在 Amazon RDS 中修改数据库参数组中的参数”。
2. CREATE EXTENSION pg_cron; 使用具有 rds_superuser 权限的账户运行此命令。这将启用 pg_cron 扩展。有关更多信息，请参阅“使用 PostgreSQL pg_cron 扩展计划维护”。
3. 运行命令 UPDATE partman.part_config 以调整 data_mart.events 表的 pg_partman 设置。
4. 运行命令 SET... 配置 data_mart.events 表，其中包含以下子句：
    a. infinite_time_partitions = true, – 将表配置为能够在没有任何限制的情况下自动创建新分区。
    b. retention = '3 months', – 将表配置为最长保留三个月。
    c. retention_keep_table=true – 对表进行配置，以便在保留期到期时表不会自动删除。相反，早于保留期的分区只能从父表中分离。
5. 运行命令 SELECT cron.schedule... 创建一个 pg_cron 函数调用。此调用定义了计划程序运行 pg_partman 维护程序的频率，partman.run_maintenance_proc。对于此示例，该程序每小时运行一次。

有关 `run_maintenance_proc` 函数的完整说明，请参阅 `pg_partman` 文档中的维护函数。

使用 pgAudit 记录数据库活动

金融机构、政府机构和许多行业需要保留审计日志以满足监管要求。通过将 PostgreSQL Audit 扩展（pgAudit）与 RDS for PostgreSQL 数据库实例结合使用，您可以捕获审计人员通常需要或满足监管要求的详细记录。例如，您可以设置 pgAudit 扩展来跟踪对特定数据库和表所做的更改，记录进行更改的用户以及许多其他详细信息。

pgAudit 扩展通过更详细地扩展日志消息，进一步构建原生 PostgreSQL 日志记录基础设施的功能。换句话说，您将使用与查看任何日志消息相同的方法来查看审计日志。有关 PostgreSQL 日志记录的更多信息，请参阅 RDS for PostgreSQL 数据库日志文件。

pgAudit 扩展会编辑日志中的敏感数据，例如明文密码。如果您的 RDS for PostgreSQL 数据库实例配置为记录数据操作语言（DML）语句（详情请见为您的 RDS for PostgreSQL 数据库实例开启查询日志记录），则可以使用 PostgreSQL Audit 扩展来避免明文密码问题。

您可以在数据库实例上配置具有高度明确性的审计。您可以审计所有数据库和所有用户。或者，您可以选择仅审计某些数据库、用户和其他对象。您也可以明确排除对某些用户和数据库进行审计。有关更多信息，请参阅 从审计日志记录中排除用户或数据库。

考虑到可以捕获的详细信息量，我们建议您在使用 pgAudit 时监控存储消耗。

所有可用的 RDS for PostgreSQL 版本。有关可用 RDS for PostgreSQL 版本支持的 pgAudit 版本的列表，请参阅《Amazon RDS for PostgreSQL 版本注释》中的 Amazon RDS for PostgreSQL 的扩展版本。

主题

设置 pgAudit 扩展

要在 RDS for PostgreSQL 数据库实例 上设置 pgAudit 扩展，首先要将 pgAudit 添加到 RDS for PostgreSQL 数据库实例的自定义数据库参数组 上的共享库中。有关创建自定义数据库参数组的信息，请参阅 Amazon RDS 的参数组。接下来，安装 pgAudit 扩展。最后，指定要审计的数据库和对象。本部分中的过程向您展示如何操作。您可以使用 Amazon Web Services Management Console或 Amazon CLI。

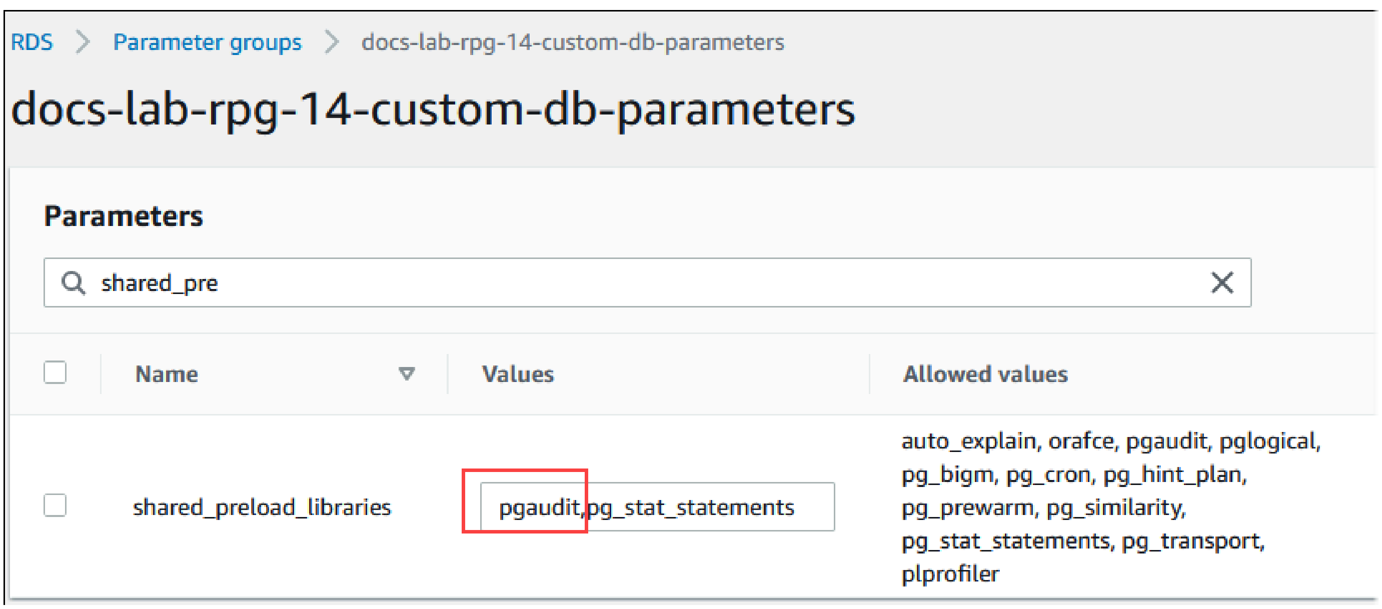
您必须拥有 rds_superuser 角色的权限才能执行所有这些任务。

以下步骤假设您的 RDS for PostgreSQL 数据库实例与自定义 数据库参数组相关联。

控制台

设置 pgAudit 扩展

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 RDS for PostgreSQL 数据库实例。
3. 打开 的配置选项卡。RDS for PostgreSQL 数据库实例的 Configuration（配置）选项卡。在实例详细信息中，找到 Parameter group（参数组）链接。
4. 选择此链接以打开与您的 RDS for PostgreSQL 数据库实例关联的自定义参数。
5. 在 Parameters（参数）搜索字段中，键入 shared_pre 以查找 shared_preload_libraries 参数。
6. 选择 Edit parameters（编辑参数）以访问属性值。
7. 将 pgaudit 添加到 Values（值）字段的列表中。使用逗号分隔值列表中的项目。



8. 重启 RDS for PostgreSQL 数据库实例，以使对 shared_preload_libraries 参数的更改生效。
9. 当实例可用时，请验证 pgAudit 是否已初始化。使用 psql 连接到 RDS for PostgreSQL 数据库实例，然后运行以下命令。

```
SHOW shared_preload_libraries;
```

```
shared_preload_libraries
--------------------------
rdsutils,pgaudit
(1 row)
```

10. 初始化 pgAudit 后，您现在可以创建扩展了。您需要在初始化库后创建扩展，因为 pgaudit 扩展会为审计数据定义语言（DDL）语句安装事件触发器。

```
CREATE EXTENSION pgaudit;
```

11. 关闭 psql 会话。

```
labdb=> \q
```

12. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
13. 在列表中找到 pgaudit.log 参数并设置为适合您的使用案例的值。例如，将 pgaudit.log 参数设置为 write（如下图所示），可以捕获对日志的插入、更新、删除和其他一些类型的更改。

还可以为 pgaudit.log 参数选择以下值之一。

- none – 这是默认值。不记录任何数据库更改。
- all – 记录所有内容（read、write、function、role、ddl、misc）。
- ddl – 记录所有数据定义语言（DDL）语句（不包括在 ROLE 类中）。
- function – 记录函数调用和 DO 块。
- misc – 记录其他命令，例如 DISCARD、FETCH、CHECKPOINT、VACUUM 和 SET。

- read –当源为关系（例如表）或查询时记录 `SELECT` 和 `COPY`。
- role – 记录与角色和权限相关的语句，例如 `GRANT`、`REVOKE`、`CREATE ROLE`、`ALTER ROLE` 和 `DROP ROLE`。
- write – 当目标为关系（表）时，记录 `INSERT`、`UPDATE`、`DELETE`、`TRUNCATE` 和 `COPY`。

14. 选择 Save changes（保存更改）。
15. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
16. 从“数据库”列表中选择 RDS for PostgreSQL 数据库实例。

Amazon CLI

设置 pgAudit

要使用 Amazon CLI 设置 pgAudit，您可以调用 modify-db-parameter-group 操作来修改自定义参数组中的审计日志参数，如以下过程所示。

1. 使用以下 Amazon CLI 命令向 `shared_preload_libraries` 参数中添加 `pgaudit`。

```
aws rds modify-db-parameter-group \
   --db-parameter-group-name custom-param-group-name \
   --parameters
 "ParameterName=shared_preload_libraries,ParameterValue=pgaudit,ApplyMethod=pending-reboot" \
   --region aws-region
```

2. 使用以下 Amazon CLI 命令重启 RDS for PostgreSQL 数据库实例，以便初始化 pgaudit 库。

```
aws rds reboot-db-instance \
    --db-instance-identifier your-instance \
    --region aws-region
```

3. 当实例可用时，您可以验证 `pgaudit` 是否已初始化。使用 `psql` 连接到 RDS for PostgreSQL 数据库实例，然后运行以下命令。

```
SHOW shared_preload_libraries;
shared_preload_libraries
--------------------------
rdsutils,pgaudit
(1 row)
```

初始化 pgAudit 后，您现在可以创建扩展了。

```
CREATE EXTENSION pgaudit;
```

4. 关闭 psql 会话以便您可以使用 Amazon CLI。

```
labdb=> \q
```

5. 使用以下 Amazon CLI 命令指定要由会话审计日志记录所记录的语句类别。该示例将 pgaudit.log 参数设置为 write，用于捕获对日志的插入、更新和删除。

```
aws rds modify-db-parameter-group \
   --db-parameter-group-name custom-param-group-name \
   --parameters
 "ParameterName=pgaudit.log,ParameterValue=write,ApplyMethod=pending-reboot" \
   --region aws-region
```

还可以为 pgaudit.log 参数选择以下值之一。

- none – 这是默认值。不记录任何数据库更改。
- all – 记录所有内容（read、write、function、role、ddl、misc）。
- ddl – 记录所有数据定义语言（DDL）语句（不包括在 ROLE 类中）。
- function – 记录函数调用和 DO 块。
- misc – 记录其他命令，例如 DISCARD、FETCH、CHECKPOINT、VACUUM 和 SET。
- read –当源为关系（例如表）或查询时记录 SELECT 和 COPY。
- role – 记录与角色和权限相关的语句，例如 GRANT、REVOKE、CREATE ROLE、ALTER ROLE 和 DROP ROLE。
- write – 当目标为关系（表）时，记录 INSERT、UPDATE、DELETE、TRUNCATE 和 COPY。

使用以下 Amazon CLI 命令重启 RDS for PostgreSQL 数据库实例。

```
aws rds reboot-db-instance \
    --db-instance-identifier your-instance \
    --region aws-region
```

审计数据库对象

在 RDS for PostgreSQL 数据库实例上设置 pgAudit 并根据您的要求进行配置后，将在 PostgreSQL 日志中捕获更多详细信息。例如，虽然默认 PostgreSQL 日志记录配置标识数据库表中发生更改的日期和时间，但使用 pgAudit 扩展后，日志条目可以包括模式、进行更改的用户和其他详细信息，具体取决于扩展参数的配置方式。您可以设置审计以通过以下方法跟踪更改。

- 对于每个会话，按用户进行跟踪。对于会话级别，您可以捕获完全限定的命令文本。
- 对于每个对象，按用户和数据库进行跟踪。

当您在系统上创建 `rds_pgaudit` 角色，然后将此角色添加到自定义参数组中的 `pgaudit.role` 参数时，将激活对象审计功能。默认情况下，`pgaudit.role` 参数处于未设置状态，唯一允许的值是 `rds_pgaudit`。以下步骤假设 `pgaudit` 已初始化，并且您已按照设置 pgAudit 扩展中的步骤创建了 `pgaudit` 扩展。

```
2022-10-07 23:36:51 UTC:52.95.4.10(14410):postgres@labdb:[1374]:LOG: statement: SELECT feedback, s.sentiment,s.confidence
FROM support,aws_comprehend.detect_sentiment(feedback, 'en') s
ORDER BY s.confidence DESC;
2022-10-07 23:36:51 UTC:52.95.4.10(14410):postgres@labdb:[1374]:LOG: AUDIT: SESSION,2,1,READ,SELECT,TABLE,public.support,"SELECT
feedback, s.sentiment,s.confidence
FROM support,aws_comprehend.detect_sentiment(feedback, 'en') s
ORDER BY s.confidence DESC;",<none>
2022-10-07 23:36:51 UTC:52.95.4.10(14410):postgres@labdb:[1374]:LOG: QUERY STATISTICS
2022-10-07 23:36:51 UTC:52.95.4.10(14410):postgres@labdb:[1374]:DETAIL: ! system usage stats:
! 0.009494 s user, 0.007442 s system, 0.141985 s elapsed
! [0.022327 s user, 0.007442 s system total]
```

如本示例所示，“LOG: AUDIT: SESSION”行提供了有关表及其架构的信息以及其他详细信息。

设置对象审计

1. 使用 `psql` 连接到 RDS for PostgreSQL 数据库实例。

```
psql --host=your-instance-name.aws-region.rds.amazonaws.com --port=5432 --
username=postgrespostgres --password --dbname=labdb
```

2. 使用以下命令创建名为 `rds_pgaudit` 的数据库角色。

```
labdb=> CREATE ROLE rds_pgaudit;
CREATE ROLE
labdb=>
```

3. 关闭 `psql` 会话。

```
labdb=> \q
```

在接下来的几步中，使用 Amazon CLI 修改自定义参数组中的审计日志参数。

4. 使用以下 Amazon CLI 命令将 `pgaudit.role` 参数设置为 `rds_pgaudit`。默认情况下，此参数为空，`rds_pgaudit` 是唯一允许的值。

```
aws rds modify-db-parameter-group \
   --db-parameter-group-name custom-param-group-name \
   --parameters
 "ParameterName=pgaudit.role,ParameterValue=rds_pgaudit,ApplyMethod=pending-reboot"
 \
   --region aws-region
```

5. 使用以下 Amazon CLI 命令重启 RDS for PostgreSQL 数据库实例，以使对参数的更改生效。

```
aws rds reboot-db-instance \
    --db-instance-identifier your-instance \
    --region aws-region
```

6. 运行以下命令确认 `pgaudit.role` 设置为 `rds_pgaudit`。

```
SHOW pgaudit.role;
pgaudit.role
------------------
rds_pgaudit
```

要测试 pgAudit 日志记录，您可以运行几条要审计的示例命令。例如，您可以运行以下命令。

```
CREATE TABLE t1 (id int);
GRANT SELECT ON t1 TO rds_pgaudit;
SELECT * FROM t1;
id
----
(0 rows)
```

数据库日志应包含类似于以下内容的条目。

```
...
2017-06-12 19:09:49 UTC:...:rds_test@postgres:[11701]:LOG: AUDIT:
```

```
OBJECT,1,1,READ,SELECT,TABLE,public.t1,select * from t1;
...
```

有关查看日志的信息，请参阅监控 Amazon RDS 日志文件。

要了解关于 pgAudit 扩展的更多信息，请参阅 GitHub 上的 pgAudit。

从审计日志记录中排除用户或数据库

如 RDS for PostgreSQL 数据库日志文件中所述，PostgreSQL 日志会消耗存储空间。使用 pgAudit 扩展会在不同程度上增加日志中收集的数据量，具体取决于您跟踪的更改。您可能不需要审计 RDS for PostgreSQL 数据库实例中的每个用户或数据库。

为了最大限度地减少对存储的影响，避免不必要地捕获审计记录，您可以将用户和数据库排除在审计范围之外。您还可以在给定会话中更改日志记录。下面的示例向您演示如何操作。

> **Note**
>
> 会话级别的参数设置优先于 RDS for PostgreSQL 数据库实例的自定义数据库参数组中的设置。如果您不希望数据库用户绕过您的审计日志记录配置设置，请务必更改其权限。

假设您的 RDS for PostgreSQL 数据库实例配置为审计所有用户和数据库的相同级别的活动。然后，您决定不想对用户 `myuser` 进行审计。您可以使用以下 SQL 命令对 `myuser` 关闭审计功能。

```
ALTER USER myuser SET pgaudit.log TO 'NONE';
```

然后，您可以使用以下查询来检查 `pgaudit.log` 的 `user_specific_settings` 列，以确认该参数已设置为 `NONE`。

```
SELECT
    usename AS user_name,
    useconfig AS user_specific_settings
FROM
    pg_user
WHERE
    usename = 'myuser';
```

您将看到如下输出。

```
 user_name | user_specific_settings
-----------+------------------------
 myuser    | {pgaudit.log=NONE}
(1 row)
```

在给定用户与数据库的会话期间，您可以使用以下命令对此用户关闭日志记录功能。

```
ALTER USER myuser IN DATABASE mydatabase SET pgaudit.log TO 'none';
```

使用以下查询，对于特定用户和数据库组合检查 pgaudit.log 的设置列。

```
SELECT
    usename AS "user_name",
    datname AS "database_name",
    pg_catalog.array_to_string(setconfig, E'\n') AS "settings"
FROM
    pg_catalog.pg_db_role_setting s
    LEFT JOIN pg_catalog.pg_database d ON d.oid = setdatabase
    LEFT JOIN pg_catalog.pg_user r ON r.usesysid = setrole
WHERE
    usename = 'myuser'
    AND datname = 'mydatabase'
ORDER BY
    1,
    2;
```

您将看到类似以下内容的输出。

```
  user_name | database_name |     settings
-----------+---------------+------------------
 myuser    | mydatabase    | pgaudit.log=none
(1 row)
```

对 `myuser` 关闭审计后，您决定不想跟踪对 `mydatabase` 的更改。您可以使用以下命令对该特定数据库关闭审计。

```
ALTER DATABASE mydatabase SET pgaudit.log to 'NONE';
```

然后，使用以下查询检查 database_specific_settings 列，以确认 pgaudit.log 已设置为 NONE。

```
SELECT
a.datname AS database_name,
b.setconfig AS database_specific_settings
FROM
pg_database a
FULL JOIN pg_db_role_setting b ON a.oid = b.setdatabase
WHERE
a.datname = 'mydatabase';
```

您将看到如下输出。

```
 database_name | database_specific_settings
---------------+----------------------------
 mydatabase    | {pgaudit.log=NONE}
(1 row)
```

要将 myuser 的设置恢复为默认设置，请使用以下命令：

```
ALTER USER myuser RESET pgaudit.log;
```

要将数据库的设置恢复为默认设置，请使用以下命令。

```
ALTER DATABASE mydatabase RESET pgaudit.log;
```

要将用户和数据库重置为默认设置，请使用以下命令。

```
ALTER USER myuser IN DATABASE mydatabase RESET pgaudit.log;
```

还可以通过将 `pgaudit.log` 设置为 `pgaudit.log` 参数的其他允许值之一，将特定事件捕获到日志中。有关更多信息，请参阅 [pgaudit.log 参数允许的设置列表]。

```
ALTER USER myuser SET pgaudit.log TO 'read';
ALTER DATABASE mydatabase SET pgaudit.log TO 'function';
ALTER USER myuser IN DATABASE mydatabase SET pgaudit.log TO 'read,function'
```

pgAudit 扩展的参考

您可以通过更改本节中列出的一个或多个参数来为审计日志指定所需的详细级别。

控制 pgAudit 行为

您可以通过更改下表中列出的一个或多个参数来控制审计日志记录。

参数	描述
`pgaudit.log`	指定会话审计日志记录将记录哪些语句类。允许的值包括 ddl、function、misc、read、role、write、none、all。有关更多信息，请参阅 `pgaudit.log` 参数允许的设置列表。
`pgaudit.log_catalog`	启用（设置为 1）时，如果语句中的所有关系都在 pg_catalog 中，则将语句添加到审计跟踪中。
`pgaudit.log_level`	指定要用于日志条目的日志级别。允许的值：debug5、debug4、debug3、debug2、debug1、info、notice、warning、log
`pgaudit.log_parameter`	启用（设置为 1）时，将在审计日志中捕获随语句传递的参数。
`pgaudit.log_relation`	启用（设置为 1）时，会话的审计日志为 SELECCT 或 DML 语句中引用的每个关系（TABLE、VIEW 等）创建单独的日志条目。
`pgaudit.log_statement_once`	指定日志记录在语句/子语句组合中的第一个日志条目中包含语句文本和参数，还是在每个条目中都包含。
`pgaudit.role`	指定用于对象审计日志记录的主角色。唯一允许的条目是 `rds_pgaudit` 。

`pgaudit.log` 参数允许的设置列表

值	描述
none	这是默认模式。不记录任何数据库更改。
all	记录所有内容（read、write、function、role、ddl、misc）。
ddl	记录所有数据定义语言（DDL）语句（不包括在 `ROLE` 类中）。

值	描述
函数	记录函数调用和 DO 块。
misc	记录其他命令，例如 DISCARD、FETCH、CHECKPOINT 、VACUUM 和 SET。
read	当源为关系（例如表）或查询时记录 SELECT 和 COPY。
role	记录与角色和权限相关的语句，例如 GRANT、REVOKE、CREATE ROLE、ALTER ROLE 和 DROP ROLE。
写入	当目标为关系（表）时，记录 INSERT、UPDATE、DELETE、TRUNCATE 和 COPY。

要使用会话审计记录多种事件类型，请使用逗号分隔的列表。要记录所有事件类型，请将 pgaudit.log 设置为 ALL。重启数据库实例以应用更改。

通过对象审计，您可以细化审计日志记录以使用特定的关系。例如，您可以指定要对一个或多个表上的 READ 操作进行审计日志记录。

使用 PostgreSQL pg_cron 扩展计划维护

您可以使用 PostgreSQL pg_cron 扩展来计划 PostgreSQL 数据库中的维护命令。有关扩展的更多信息，请参阅 pg_cron 文档中的什么是 pg_cron？

RDS for PostgreSQL 引擎版本 12.5 及更高版本支持 pg_cron 扩展。

要了解有关使用 pg_cron 的更多信息，请参阅在 RDS for PostgreSQL 或 Aurora PostgreSQL 兼容版数据库上使用 pg_cron 计划任务。

主题

设置 pg_cron 扩展

按如下方式设置 pg_cron 扩展：

1. 通过向 shared_preload_libraries 参数值添加 pg_cron，修改与 PostgreSQL 数据库实例关联的自定义参数组。
   - 如果 RDS for PostgreSQL 数据库实例使用 rds.allowed_extensions 参数显式列出可安装的扩展，则需要将 pg_cron 扩展添加到列表中。只有某些版本的 RDS for PostgreSQL 支持 rds.allowed_extensions 参数。原定设置情况下，允许使用所有可用的扩展。有关更多信息，请参阅 限制 PostgreSQL 扩展的安装。

   重新启动 PostgreSQL 数据库实例，以使对参数组的更改生效。要了解有关使用参数组的更多信息，请参阅在 Amazon RDS 中修改数据库参数组中的参数。
2. 重新启动 PostgreSQL 数据库实例后，使用具有 rds_superuser 权限的账户运行以下命令。例如，如果在创建 RDS for PostgreSQL 数据库实例时使用默认设置，请以用户 postgres 身份进行连接，然后创建扩展。

   ```
   CREATE EXTENSION pg_cron;
   ```

   pg_cron 调度程序是在名为 postgres 的默认 PostgreSQL 数据库中设置的。这些 pg_cron 对象是在此 postgres 数据库中创建的，所有调度操作都在此数据库中运行。

3. 您可以使用默认设置，也可以计划作业在 PostgreSQL 数据库实例的其他数据库中运行。要为 PostgreSQL 数据库实例中的其他数据库计划作业，请参阅 为原定设置数据库以外的数据库计划 cron 任务 中的示例。

授予数据库用户使用 pg_cron 的权限

安装 pg_cron 扩展需要 rds_superuser 权限。但是，可以（由 pg_cron 组/角色的成员）将使用 rds_superuser 的权限授予其他数据库用户，以便他们可以计划自己的任务。我们建议您仅在需要时才授予对 cron 架构的权限，前提是它可以改进生产环境中的操作。

要在 cron 架构中授予数据库用户权限，请运行以下命令：

```
postgres=> GRANT USAGE ON SCHEMA cron TO db-user;
```

这向 *db-user* 授予访问 cron 架构的权限，以便为他们有权限访问的对象计划 cron 任务。如果数据库用户没有权限，则在将错误消息发布到 postgresql.log 文件后，任务会失败，如下所示：

```
2020-12-08 16:41:00 UTC::@:[30647]:ERROR: permission denied for table table-name
2020-12-08 16:41:00 UTC::@:[27071]:LOG: background worker "pg_cron" (PID 30647) exited
 with exit code 1
```

换言之，请确保被授予对 cron 架构的权限的数据库用户也对他们计划安排的对象（表、架构等）拥有权限。

cron 任务的详细信息及其成功或失败情况也在 cron.job_run_details 表中捕获。有关更多信息，请参阅 用于计划任务和捕获状态的表 。

计划 pg_cron 作业

以下各节介绍了如何使用 pg_cron 作业安排各种管理任务。

> **Note**
>
> 创建 pg_cron 任务时，请检查 max_worker_processes 设置是否大于 cron.max_running_jobs 的数量。如果 pg_cron 任务耗尽后台工作进程，它将失败。原定设置的 pg_cron 任务数量为 5。有关更多信息，请参阅 用于管理 pg_cron 扩展的参数。

主题

- 对表执行清理操作
- 清除 pg_cron 历史记录表
- 仅将错误记录到 postgresql.log 文件中
- 为原定设置数据库以外的数据库计划 cron 任务

对表执行清理操作

Autovacuum 在大多数情况下处理清理维护。但是，您可能希望在选择的时间计划对特定表执行清理操作。

另请参阅在 Amazon RDS for PostgreSQL 上使用 PostgreSQL autovacuum。

以下示例介绍了使用 `cron.schedule` 函数设置作业，以便每天 22:00 (GMT) 在特定表上使用 `VACUUM FREEZE`。

```
SELECT cron.schedule('manual vacuum', '0 22 * * *', 'VACUUM FREEZE pgbench_accounts');
 schedule
----------
1
(1 row)
```

运行上述示例之后，您可以按如下方式检查 `cron.job_run_details` 表中的历史记录。

```
postgres=> SELECT * FROM cron.job_run_details;
jobid  | runid | job_pid | database | username | command                         |
 status     | return_message | start_time                    | end_time
-------+-------+---------+----------+----------+--------------------------------
+-----------+----------------+-------------------------------
+-------------------------------
 1     | 1     | 3395    | postgres | adminuser| vacuum freeze pgbench_accounts
 | succeeded | VACUUM         | 2020-12-04 21:10:00.050386+00 | 2020-12-04
 21:10:00.072028+00
(1 row)
```

下面的内容说明如何查询 `cron.job_run_details` 表以查看失败的任务。

```
postgres=> SELECT * FROM cron.job_run_details WHERE status = 'failed';
jobid | runid | job_pid | database | username | command                       | status
 | return_message                                    | start_time                      |
 end_time
```

```
------+-------+---------+----------+----------+-------------------------------+--------
+-----------------------------------------------+------------------------------
+------------------------------
 5    | 4     | 30339   | postgres | adminuser| vacuum freeze pgbench_account | failed
 | ERROR: relation "pgbench_account" does not exist | 2020-12-04 21:48:00.015145+00 |
 2020-12-04 21:48:00.029567+00
(1 row)
```

有关更多信息，请参阅 用于计划任务和捕获状态的表 。

清除 pg_cron 历史记录表

cron.job_run_details 表包含 cron 作业的历史记录，随着时间的推移，这些历史记录可能会变得非常大。我们建议您计划清除此表的作业。例如，保留一周的条目可能足以进行故障排除。

以下示例使用 cron.schedule 函数计划每天午夜运行以清除 cron.job_run_details 表的作业。这项工作只保留了过去七天的历史记录。使用您的 rds_superuser 账户计划作业，如下所示。

```
SELECT cron.schedule('0 0 * * *', $$DELETE
    FROM cron.job_run_details
    WHERE end_time < now() - interval '7 days'$$);
```

有关更多信息，请参阅 用于计划任务和捕获状态的表 。

仅将错误记录到 postgresql.log 文件中

要防止向 cron.job_run_details 表中进行写入，请修改与 PostgreSQL 数据库实例关联的参数组，然后将 cron.log_run 参数设置为 off。pg_cron 扩展不再写入表，只会将错误捕获到 postgresql.log 文件中。有关更多信息，请参阅 在 Amazon RDS 中修改数据库参数组中的参数。

使用以下命令检查 cron.log_run 参数的值。

```
postgres=> SHOW cron.log_run;
```

有关更多信息，请参阅 用于管理 pg_cron 扩展的参数。

为原定设置数据库以外的数据库计划 cron 任务

pg_cron 的元数据全部保存在名为 postgres 的 PostgreSQL 默认数据库中。由于后台工件用于运行维护 cron 作业，因此您可以在 PostgreSQL 数据库实例中的任何数据库中计划作业。

> **Note**
>
> 只有具有 rds_superuser 角色或 rds_superuser 权限的用户才能列出数据库中的所有 cron 作业。其他用户只能在 cron.job 表中查看其自己的作业。

1. 在 cron 数据库中，以与平常使用 cron.schedule 相同的方式计划作业。

```
postgres=> SELECT cron.schedule('database1 manual vacuum', '29 03 * * *', 'vacuum
 freeze test_table');
```

2. 作为具有 rds_superuser 角色的用户，请更新刚创建的作业的数据库列，使其在 PostgreSQL 数据库实例中的另一个数据库中运行。

```
postgres=> UPDATE cron.job SET database = 'database1' WHERE jobid = 106;
```

3. 通过查询 cron.job 表进行验证。

```
postgres=> SELECT * FROM cron.job;
jobid | schedule     | command                         | nodename   | nodeport |
 database | username   | active | jobname
------+-------------+--------------------------------+-----------+----------
+----------+-----------+--------+------------------------
106   | 29 03 * * * | vacuum freeze test_table        | localhost | 8192     |
 database1| adminuser | t      | database1 manual vacuum
  1   | 59 23 * * * | vacuum freeze pgbench_accounts | localhost | 8192     |
 postgres | adminuser | t      | manual vacuum
(2 rows)
```

> **Note**
>
> 在某些情况下，您可以添加打算在其他数据库上运行的 cron 作业。在这些情况下，在您更新正确的数据库列之前，该作业可能会尝试在默认数据库 (postgres) 中运行。如果用户名具有权限，则作业将在默认数据库中成功运行。

pg_cron 扩展的参考

您可以将以下参数、函数和表与 pg_cron 扩展搭配使用。有关更多信息，请参阅 pg_cron 文档中的什么是 pg_cron？。

主题

用于管理 pg_cron 扩展的参数

以下是用于控制 pg_cron 扩展行为的参数列表。

参数	描述
cron.database_name	保存 pg_cron 元数据的数据库。
`cron.host`	要连接到 PostgreSQL 的主机名。您无法修改此值。
`cron.log_run`	在 `job_run_details` 表中记录运行的每个任务。值为 on 或 `off`。有关更多信息，请参阅“用于计划任务和捕获状态的表”。
`cron.log_statement`	在运行所有 cron 语句之前将其记入日志。值为 on 或 `off`。
`cron.max_running_jobs`	可以同时运行的最大作业数。
`cron.use_background_workers`	使用后台工作程序而不是客户端会话。您无法修改此值。

使用以下 SQL 命令来显示这些参数及其值。

```
postgres=> SELECT name, setting, short_desc FROM pg_settings WHERE name LIKE 'cron.%'
 ORDER BY name;
```

函数参考：cron.schedule

此函数计划 cron 作业。作业最初是在默认 `postgres` 数据库中计划的。该函数返回一个表示作业标识符的 `bigint` 值。要计划作业在 PostgreSQL 数据库实例的其他数据库中运行，请参阅 为原定设置数据库以外的数据库计划 cron 任务 中的示例。

该函数有两种语法格式。

语法

```
cron.schedule (job_name,
    schedule,
    command
);

cron.schedule (schedule,
    command
);
```

参数

参数	描述
`job_name`	cron 作业的名字。
`schedule`	表示 cron 作业时间表的文本。格式是标准 cron 格式。
`command`	要运行的命令的文本。

示例

```
postgres=> SELECT cron.schedule ('test','0 10 * * *', 'VACUUM pgbench_history');
 schedule
----------
      145
(1 row)
```

```
postgres=> SELECT cron.schedule ('0 15 * * *', 'VACUUM pgbench_accounts');
 schedule
----------
      146
(1 row)
```

函数参考：cron.unschedule

此函数删除 cron 作业。您可以指定 job_name 或 job_id。策略可以确保您是删除作业计划的拥有者。该函数返回一个布尔值，指示成功或失败。

该函数使用以下语法格式。

语法

```
cron.unschedule (job_id);

cron.unschedule (job_name);
```

参数

参数	描述
job_id	计划 cron 作业时从 cron.schedule 函数返回的作业标识符。
job_name	使用该 cron.schedule 函数计划的 cron 作业的名称。

示例

```
postgres=> SELECT cron.unschedule(108);
 unschedule
------------
 t
(1 row)

postgres=> SELECT cron.unschedule('test');
```

```
 unschedule
------------
 t
(1 row)
```

用于计划任务和捕获状态的表

将以下各表用于计划 cron 作业和记录作业完成的方式。

表	描述
`cron.job`	包含有关每个计划作业的元数据。与此表的大多数交互应使用 `cron.schedule` 和 `cron.unschedule` 函数完成。 **Important** 我们不建议直接授予对此表的更新或插入权限。这样做将允许用户更新 `username` 列，从而以 `rds-superuser` 身份运行。
`cron.job_run_details`	包含过去运行的计划作业的历史信息。这对于调查运行的作业的状态、返回消息以及开始和结束时间非常有用。 **Note** 为了防止此表无限增长，请定期清除此表。有关示例，请参阅清除 pg_cron 历史记录表。

使用 pglogical 跨实例同步数据

所有当前可用的 RDS for PostgreSQL 版本都支持 `pglogical` 扩展。pglogic 扩展早于 PostgreSQL 在版本 10 中引入的功能类似的逻辑复制特征。有关更多信息，请参阅为 Amazon RDS for PostgreSQL 执行逻辑复制。

`pglogical` 扩展支持在两个或更多 RDS for PostgreSQL 数据库实例之间进行逻辑复制。它还支持在不同的 PostgreSQL 版本之间进行复制，以及在 RDS for PostgreSQL 数据库实例和 Aurora PostgreSQL 数据库集群上运行的数据库之间进行复制。`pglogical` 扩展使用发布-订阅模型将对表和其他对象（例如序列）的更改从发布者复制到订阅者。它依赖于复制插槽来确保更改从发布者节点同步到订阅者节点，定义如下。

- 发布者节点是作为要复制到其他节点的数据来源的 RDS for PostgreSQL 数据库实例。发布者节点定义要在发布集中复制的表。
- 订阅者节点是用于接收来自发布商的 WAL 更新的 RDS for PostgreSQL 数据库实例。订阅者创建订阅以连接到发布者并获取解码后的 WAL 数据。订阅者创建订阅时，将在发布者节点上创建复制插槽。

在下文中，您可以了解有关设置 `pglogical` 扩展的信息。

主题

- pglogical 扩展的要求和限制
- 设置 pglogical 扩展
- 为 RDS for PostgreSQL 数据库实例设置逻辑复制
- 在主要升级后重新建立逻辑复制
- 管理 RDS for PostgreSQL 的逻辑复制查槽
- pglogical 扩展的参数参考

pglogical 扩展的要求和限制

所有当前可用的 RDS for PostgreSQL 版本都支持 `pglogical` 扩展。

发布者节点和订阅者节点都必须设置为进行逻辑复制。

要从发布者复制到订阅用户的表必须具有相同的名称和相同的架构。这些表还必须包含相同的列，并且这些列必须使用相同的数据类型。发布者表和订阅者表必须具有相同的主键。我们建议您仅使用 PRIMARY KEY 作为唯一约束。

对于 CHECK 约束和 NOT NULL 约束，订阅者节点上的表可能比发布者节点上的表具有更宽松的约束。

pglogical 扩展提供了诸如双向复制之类的特征，PostgreSQL（版本 10 及更高版本）中内置的逻辑复制特征不支持这些特征。有关更多信息，请参阅使用 pglogic 进行 PostgreSQL 双向复制。

设置 pglogical 扩展

要在 RDS for PostgreSQL 数据库实例上设置 pglogical 扩展，首先要将 pglogical 添加到 RDS for PostgreSQL 数据库实例的自定义数据库参数组上的共享库中。您还需要将 rds.logical_replication 参数的值设置为 1，以开启逻辑解码。最后，在数据库中创建此扩展。您可以使用 Amazon Web Services Management Console或 Amazon CLI 执行这些任务。

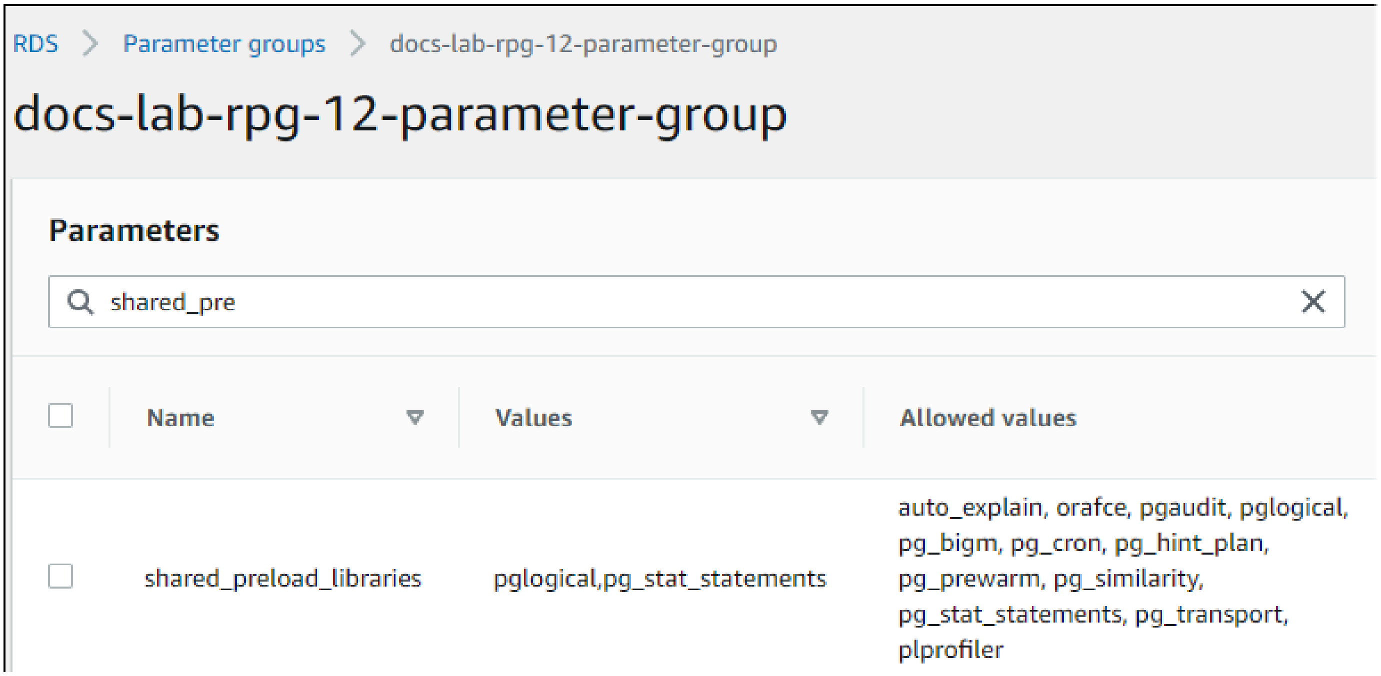
您必须拥有 rds_superuser 角色的权限才能执行这些任务。

以下步骤假设您的 RDS for PostgreSQL 数据库实例与自定义 数据库参数组相关联。有关创建自定义数据库参数组的信息，请参阅 Amazon RDS 的参数组。

控制台

设置 pglogical 扩展

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 RDS for PostgreSQL 数据库实例。
3. 打开 的配置选项卡。RDS for PostgreSQL 数据库实例的 Configuration（配置）选项卡。在实例详细信息中，找到 Parameter group（参数组）链接。
4. 选择此链接以打开与您的 RDS for PostgreSQL 数据库实例关联的自定义参数。
5. 在 Parameters（参数）搜索字段中，键入 shared_pre 以查找 shared_preload_libraries 参数。
6. 选择 Edit parameters（编辑参数）以访问属性值。
7. 将 pglogical 添加到 Values（值）字段的列表中。使用逗号分隔值列表中的项目。



8. 找到 `rds.logical_replication` 参数并将其设置为 1，以开启逻辑复制。
9. 重启 RDS for PostgreSQL 数据库实例，以使更改生效。
10. 当实例可用时，可以使用 `psql`（或 pgAdmin）连接到 RDS for PostgreSQL 数据库实例。

```
psql --host=111122223333.aws-region.rds.amazonaws.com --port=5432 --
username=postgres --password --dbname=labdb
```

11. 要验证 pglogical 是否初始化，可以运行以下命令。

```
SHOW shared_preload_libraries;
shared_preload_libraries
--------------------------
rdsutils,pglogical
(1 row)
```

12. 验证启用逻辑解码的设置，如下所示。

```
SHOW wal_level;
wal_level
-----------
 logical
(1 row)
```

13. 创建扩展，如下所示。

```
CREATE EXTENSION pglogical;
EXTENSION CREATED
```

14. 选择 Save changes（保存更改）。
15. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
16. 从数据库列表中选择 RDS for PostgreSQL 数据库实例以将其选中，然后从 Actions（操作）菜单中选择 Reboot（重启）。

Amazon CLI

设置 pglogical 扩展

要使用 Amazon CLI 设置 pglogical，您可以调用 modify-db-parameter-group 操作来修改自定义参数组中的某些参数，如以下过程所示。

1. 使用以下 Amazon CLI 命令向 `shared_preload_libraries` 参数中添加 `pglogical`。

```
aws rds modify-db-parameter-group \
   --db-parameter-group-name custom-param-group-name \
   --parameters
 "ParameterName=shared_preload_libraries,ParameterValue=pglogical,ApplyMethod=pending-
reboot" \
   --region aws-region
```

2. 使用以下 Amazon CLI 命令将 `rds.logical_replication` 设置为 1，以针对 RDS for PostgreSQL 数据库实例开启逻辑解码功能。

```
aws rds modify-db-parameter-group \
   --db-parameter-group-name custom-param-group-name \
   --parameters
 "ParameterName=rds.logical_replication,ParameterValue=1,ApplyMethod=pending-
reboot" \
   --region aws-region
```

3. 使用以下 Amazon CLI 命令重启 RDS for PostgreSQL 数据库实例，以便初始化 pglogical 库。

```
aws rds reboot-db-instance \
    --db-instance-identifier your-instance \
    --region aws-region
```

4. 当实例可用时，使用 psql 连接到 RDS for PostgreSQL 数据库实例。

```
psql --host=111122223333.aws-region.rds.amazonaws.com --port=5432 --
username=postgres --password --dbname=labdb
```

5. 创建扩展，如下所示。

```
CREATE EXTENSION pglogical;
EXTENSION CREATED
```

6. 使用以下 Amazon CLI 命令重启 RDS for PostgreSQL 数据库实例。

```
aws rds reboot-db-instance \
    --db-instance-identifier your-instance \
    --region aws-region
```

为 RDS for PostgreSQL 数据库实例设置逻辑复制

以下过程说明如何在两个 RDS for PostgreSQL 数据库实例之间启动逻辑复制。这些步骤假设来源（发布者）和目标（订阅者）都如设置 pglogical 扩展中所述设置了 pglogical 扩展。

> **Note**
>
> 订阅用户节点的 node_name 不能以 rds 开头。

创建发布者节点并定义要复制的表

这些步骤假设您的 RDS for PostgreSQL 数据库实例有一个数据库，其中包含一个或多个您要复制到另一个节点的表。您需要在订阅者上根据发布者重新创建表结构，因此，如果需要，首先获取表结构。为此，您可以使用 psql 元命令 \d *tablename*，然后在订阅者实例上创建相同的表。以下过程在发布者（来源）上创建示例表以用于演示目的。

1. 使用 psql 连接到具有要用作订阅者来源的表的实例。

```
psql --host=source-instance.aws-region.rds.amazonaws.com --port=5432 --
username=postgres --password --dbname=labdb
```

如果没有现有表要复制，可以按如下方式创建示例表。

a. 使用以下 SQL 语句创建一个示例表。

```
CREATE TABLE docs_lab_table (a int PRIMARY KEY);
```

b. 使用以下 SQL 语句用生成的数据填充表。

```
INSERT INTO docs_lab_table VALUES (generate_series(1,5000));
INSERT 0 5000
```

c. 使用以下 SQL 语句验证表中是否存在数据。

```
SELECT count(*) FROM docs_lab_table;
```

2. 将这一 RDS for PostgreSQL 数据库实例标识为发布者节点，如下所示。

```
SELECT pglogical.create_node(
    node_name := 'docs_lab_provider',
    dsn := 'host=source-instance.aws-region.rds.amazonaws.com port=5432
 dbname=labdb');
 create_node
-------------
   3410995529
(1 row)
```

3. 将要复制的表添加到默认的复制集。有关复制集的更多信息，请参阅 pglogical 文档中的复制集。

```
SELECT pglogical.replication_set_add_table('default', 'docs_lab_table', 'true',
 NULL, NULL);
 replication_set_add_table
  ---------------------------
  t
  (1 row)
```

发布者节点设置已完成。现在，您可以设置订阅者节点以接收来自发布者的更新。

设置订阅者节点并创建订阅以接收更新

这些步骤假设已使用 `pglogical` 扩展设置了 RDS for PostgreSQL 数据库实例。有关更多信息，请参阅 设置 pglogical 扩展。

1. 使用 psql 连接到要从发布者接收更新的实例。

```
psql --host=target-instance.aws-region.rds.amazonaws.com --port=5432 --
username=postgres --password --dbname=labdb
```

2. 在订阅者 RDS for PostgreSQL 数据库实例上，创建与发布者上存在的相同表。在本例中，该表为 docs_lab_table。您可以按如下所示创建表。

```
CREATE TABLE docs_lab_table (a int PRIMARY KEY);
```

3. 验证此表为空。

```
SELECT count(*) FROM docs_lab_table;
 count
-------
  0
(1 row)
```

4. 将这一 RDS for PostgreSQL 数据库实例标识为订阅者节点，如下所示。

```
SELECT pglogical.create_node(
    node_name := 'docs_lab_target',
    dsn := 'host=target-instance.aws-region.rds.amazonaws.com port=5432
 sslmode=require dbname=labdb user=postgres password=********');
 create_node
-------------
   2182738256
(1 row)
```

5. 创建订阅。

```
SELECT pglogical.create_subscription(
   subscription_name := 'docs_lab_subscription',
   provider_dsn := 'host=source-instance.aws-region.rds.amazonaws.com port=5432
 sslmode=require dbname=labdb user=postgres password=*******',
   replication_sets := ARRAY['default'],
   synchronize_data := true,
   forward_origins := '{}' );
 create_subscription
---------------------
1038357190
(1 row)
```

完成此步骤后，将在订阅者上的表中创建发布者上表中的数据。您可以使用以下 SQL 查询来验证是否已发生这种情况。

```
SELECT count(*) FROM docs_lab_table;
 count
-------
  5000
(1 row)
```

此后，对发布者上的表所做的更改将复制到订阅者上的表中。

在主要升级后重新建立逻辑复制

对于设置为逻辑复制的发布者节点的 RDS for PostgreSQL 数据库实例，在可以对其执行主要版本升级之前，您必须删除所有复制插槽，即使是不活动的复制插槽也是如此。我们建议您暂时从发布者节点转移数据库事务，删除复制插槽，升级 RDS for PostgreSQL 数据库实例，然后重新建立并重新启动复制。

复制插槽仅托管在发布者节点上。逻辑复制场景中的 RDS for PostgreSQL 订阅者节点没有可删除的插槽，但当它被指定为对发布者具有订阅的订阅者节点时，它无法升级到主要版本。在升级 RDS for PostgreSQL 订阅者节点之前，请删除订阅和节点。有关更多信息，请参阅。[管理 RDS for PostgreSQL 的逻辑复制查槽](#)

确定逻辑复制已中断

您可以通过查询发布者节点或订阅者节点来确定复制过程是否已中断，如下所示。

检查发布者节点

- 使用 `psql` 连接到发布者节点，然后查询 `pg_replication_slots` 函数。注意活动列中的值。通常，这将返回 `t`（true），表明复制处于活动状态。如果查询返回 `f`（false），则表明向订阅者的复制已停止。

```
SELECT slot_name,plugin,slot_type,active FROM pg_replication_slots;
                    slot_name                  |      plugin      | slot_type | active
-----------------------------------------------+------------------+-----------+--------
 pgl_labdb_docs_labcb4fa94_docs_lab3de412c | pglogical_output | logical   | f
(1 row)
```

检查订阅者节点

在订阅者节点上，您可以通过三种不同的方式检查复制的状态。

- 浏览订阅者节点上的 PostgreSQL 日志，以查找失败消息。该日志使用包含退出代码 1 的消息来标识故障，如下所示。

```
2022-07-06 16:17:03 UTC::@:[7361]:LOG: background worker "pglogical apply
 16404:2880255011" (PID 14610) exited with exit code 1
2022-07-06 16:19:44 UTC::@:[7361]:LOG: background worker "pglogical apply
 16404:2880255011" (PID 21783) exited with exit code 1
```

- 查询 `pg_replication_origin` 函数。使用 `psql` 连接到订阅者节点上的数据库并查询 `pg_replication_origin` 函数，如下所示。

```
SELECT * FROM pg_replication_origin;
 roident | roname
---------+--------
(0 rows)
```

  结果集为空表示复制已中断。正常情况下，您将看到如下输出。

```
   roident |                       roname
  ---------+----------------------------------------------------
         1 | pgl_labdb_docs_labcb4fa94_docs_lab3de412c
  (1 row)
```

- 查询 `pglogical.show_subscription_status` 函数，如以下示例所示。

```
SELECT subscription_name,status,slot_name FROM pglogical.show_subscription_status();
     subscription_name | status |              slot_name
---====----------------+--------+------------------------------------
 docs_lab_subscription | down   | pgl_labdb_docs_labcb4fa94_docs_lab3de412c
(1 row)
```

  此输出显示复制已中断。它的状态为 `down`。通常，输出将状态显示为 `replicating`。

如果您的逻辑复制过程已中断，则可以按照以下步骤重新建立复制。

在发布者节点和订阅者节点之间重新建立逻辑复制

要重新建立复制，请先断开订阅者与发布者节点的连接，然后重新建立订阅，如这些步骤所述。

1. 使用 psql 连接到订阅者节点，如下所示。

```
psql --host=222222222222.aws-region.rds.amazonaws.com --port=5432 --
username=postgres --password --dbname=labdb
```

2. 通过使用 pglogical.alter_subscription_disable 函数停用订阅。

```
SELECT pglogical.alter_subscription_disable('docs_lab_subscription',true);
 alter_subscription_disable
----------------------------
 t
(1 row)
```

3. 通过查询 pg_replication_origin 获取发布者节点的标识符，如下所示。

```
SELECT * FROM pg_replication_origin;
 roident |                roname
---------+-------------------------------------
       1 | pgl_labdb_docs_labcb4fa94_docs_lab3de412c
(1 row)
```

4. 将上一步的响应与 pg_replication_origin_create 命令一起使用，以分配重新建立时订阅可以使用的标识符。

```
SELECT pg_replication_origin_create('pgl_labdb_docs_labcb4fa94_docs_lab3de412c');
  pg_replication_origin_create
------------------------------
                            1
(1 row)
```

5. 通过传递其状态为 true 的名称来打开订阅，如下面的示例所示。

```
SELECT pglogical.alter_subscription_enable('docs_lab_subscription',true);
  alter_subscription_enable
---------------------------
 t
(1 row)
```

检查节点的状态。其状态应为 `replicating`，如本例所示。

```
SELECT subscription_name,status,slot_name
  FROM pglogical.show_subscription_status();
             subscription_name |   status    |              slot_name
-------------------------------+-------------+-------------------------------------
 docs_lab_subscription         | replicating |
 pgl_labdb_docs_lab98f517b_docs_lab3de412c
(1 row)
```

检查发布者节点上订阅者的复制插槽的状态。插槽的 `active` 列应返回 `t`（true），表示已重新建立复制。

```
SELECT slot_name,plugin,slot_type,active
  FROM pg_replication_slots;
                    slot_name              |      plugin      | slot_type | active
-------------------------------------------+------------------+-----------+--------
 pgl_labdb_docs_lab98f517b_docs_lab3de412c | pglogical_output | logical   | t
(1 row)
```

管理 RDS for PostgreSQL 的逻辑复制查槽

对于在逻辑复制场景中充当发布者节点的 RDS for PostgreSQL 数据库实例，在对其执行主要版本升级之前，必须删除该实例上的复制插槽。主要版本升级预检查过程会通知您，在插槽被删除之前，升级无法继续。

要从 RDS for PostgreSQL 数据库实例中删除插槽，请先删除订阅，然后删除插槽。

要识别使用 `pglogical` 扩展创建的复制插槽，请登录到每个数据库并获取节点的名称。当您查询订阅者节点时，您会在输出中得到发布者节点和订阅者节点，如本示例所示。

```
SELECT * FROM pglogical.node;
node_id   |     node_name
------------+-------------------
 2182738256 | docs_lab_target
 3410995529 | docs_lab_provider
(2 rows)
```

您可以通过以下查询获取有关订阅的详细信息。

```
SELECT sub_name,sub_slot_name,sub_target
```

```
  FROM pglogical.subscription;
 sub_name |         sub_slot_name          | sub_target
----------+--------------------------------+------------
  docs_lab_subscription     | pgl_labdb_docs_labcb4fa94_docs_lab3de412c | 2182738256
(1 row)
```

现在您可以删除订阅，如下所示。

```
SELECT pglogical.drop_subscription(subscription_name := 'docs_lab_subscription');
 drop_subscription
-------------------
                 1
(1 row)
```

删除订阅后，您可以删除该节点。

```
SELECT pglogical.drop_node(node_name := 'docs-lab-subscriber');
 drop_node
-----------
 t
(1 row)
```

您可以验证该节点是否不再存在，如下所示。

```
SELECT * FROM pglogical.node;
 node_id | node_name
---------+-----------
(0 rows)
```

pglogical 扩展的参数参考

在表中，您可以找到与 pglogical 扩展关联的参数。pglogical.conflict_log_level 和 pglogical.conflict_resolution 等参数用于处理更新冲突。当对订阅来自发布者的更改的相同表进行本地更改时，可能会出现冲突。在不同情况下也可能发生冲突，例如双向复制或当多个订阅者从同一个发布者进行复制时。有关更多信息，请参阅使用 pglogical 进行 PostgreSQL 双向复制。

参数	描述
pglogical.batch_inserts	在可能时执行批量插入。默认情况下未设置。更改为“1”将打开，更改为“0”将关闭。

参数	描述
pglogical.conflict_log_level	设置用于记录已解决的冲突的日志级别。支持的字符串值为 debug5、debug4、debug3、debug2、debug1、info、notice、warning、error、log、fatal、panic。
pglogical.conflict_resolution	设置在冲突可以解决时用来解决冲突的方法。支持的字符串值为 error、apply_remote、keep_local、last_update_wins、first_update_wins。
pglogical.extra_connection_options	要添加到所有对等节点连接的连接选项。
pglogical.synchronous_commit	pglogical 专用的同步提交值
pglogical.use_spi	使用 SPI（服务器编程接口）而非低级 API 来应用更改。设置为“1”将打开，设置为“0”将关闭。有关 SPI 的更多信息，请参阅 PostgreSQL 文档中的服务器编程接口。

使用 pgactive 支持主动-主动复制

`pgactive` 扩展使用主动-主动复制来支持和协调多个 RDS for PostgreSQL 数据库上的写入操作。Amazon RDS for PostgreSQL 支持以下版本的 `pgactive` 扩展。

- RDS for PostgreSQL 17.0 和所有更高版本
- RDS for PostgreSQL 16.1 及更高的 16 版本
- RDS for PostgreSQL 15.4-R2 及更高的 15 版本
- RDS for PostgreSQL 14.10 及更高的 14 版本
- RDS for PostgreSQL 13.13 及更高的 13 版本
- RDS for PostgreSQL 12.17 及更高的 12 版本
- RDS for PostgreSQL 11.22

> **Note**
>
> 当复制配置中的多个数据库上有写入操作时，可能会发生冲突。有关更多信息，请参阅 处理主动-主动复制中的冲突。

主题

pgactive 扩展的局限性

- 所有表都需要主键，否则不允许使用更新和删除。不应更新“主键”列中的值。
- 序列可能存在间隙，有时可能不遵循顺序。不会复制序列。有关更多信息，请参阅 处理主动-主动复制中的序列。
- 不会复制 DDL 和大型对象。
- 辅助唯一索引可能会导致数据差异。
- 组中所有节点上的排序规则需要相同。
- 跨节点负载均衡是一种反模式。
- 大型事务可能会导致复制滞后。

初始化 pgactive 扩展功能

要在 RDS for PostgreSQL 数据库实例上初始化 `pgactive` 扩展功能，请将参数 `rds.enable_pgactive` 的值设置为 `1`，然后在数据库中创建扩展。这样做会自动开启参数 `rds.logical_replication` 和 `track_commit_timestamp` 并将 `wal_level` 的值设置为 `logical`。

您必须拥有 `rds_superuser` 角色的权限才能执行这些任务。

您可以使用 Amazon Web Services Management Console或 Amazon CLI 创建所需的 RDS for PostgreSQL 数据库实例。以下步骤假设您的 RDS for PostgreSQL 数据库实例与自定义数据库参数组相关联。有关创建自定义数据库参数组的信息，请参阅Amazon RDS 的参数组。

控制台

初始化 pgactive 扩展功能

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 RDS for PostgreSQL 数据库实例。
3. 打开 RDS for PostgreSQL 数据库实例的配置选项卡。在实例详细信息中，找到数据库实例参数组链接。
4. 选择此链接以打开与 RDS for PostgreSQL 数据库实例关联的自定义参数。
5. 找到 `rds.enable_pgactive` 参数，并将其设置为 `1` 以初始化 `pgactive` 功能。

6. 选择保存更改。
7. 在 Amazon RDS 控制台的导航窗格中，选择数据库。
8. 选择您的 RDS for PostgreSQL 数据库实例，然后从操作菜单中选择重启。
9. 确认数据库实例重启，以便您的更改生效。
10. 当数据库实例可用时，您可以使用 psql 或任何其它 PostgreSQL 客户端连接到 RDS for PostgreSQL 数据库实例。

    以下示例假设 RDS for PostgreSQL 数据库实例有一个名为 *postgres* 的原定设置数据库。

    ```
    psql --host=mydb.111122223333.aws-region.rds.amazonaws.com --port=5432 --
    username=postgres --password=PASSWORD --dbname=postgres
    ```

11. 要验证 pgactive 是否初始化，可以运行以下命令。

    ```
    postgres=>SELECT setting ~ 'pgactive' 
    FROM pg_catalog.pg_settings
    WHERE name = 'shared_preload_libraries';
    ```

    如果 pgactive 在 shared_preload_libraries 中，则前面的命令将返回以下内容：

    ```

    ?column?
    ----------
     t
    ```

Amazon CLI

初始化 pgactive 扩展功能

要使用 Amazon CLI 设置 pgactive，请调用 [modify-db-parameter-group] 操作来修改自定义参数组中的某些参数，如以下过程所示。

1. 使用以下 Amazon CLI 命令将 rds.enable_pgactive 设置为 1，以初始化 RDS for PostgreSQL 数据库实例的 pgactive 功能。

    ```
    postgres=>aws rds modify-db-parameter-group \
       --db-parameter-group-name custom-param-group-name \
       --parameters
     "ParameterName=rds.enable_pgactive,ParameterValue=1,ApplyMethod=pending-reboot" \
    ```

```
    --region aws-region
```

2. 使用以下 Amazon CLI 命令重启 RDS for PostgreSQL 数据库实例，以便初始化 pgactive 库。

```
aws rds reboot-db-instance \
    --db-instance-identifier your-instance \
    --region aws-region
```

3. 当实例可用时，使用 psql 连接到 RDS for PostgreSQL 数据库实例。

```
psql --host=mydb.111122223333.aws-region.rds.amazonaws.com --port=5432 --
username=master user --password=PASSWORD --dbname=postgres
```

4. 要验证 pgactive 是否初始化，可以运行以下命令。

```
postgres=>SELECT setting ~ 'pgactive'
FROM pg_catalog.pg_settings
WHERE name = 'shared_preload_libraries';
```

如果 pgactive 在 shared_preload_libraries 中，则前面的命令将返回以下内容：

```

?column?
----------
 t
```

为 RDS for PostgreSQL 数据库实例设置主动-主动复制

以下过程说明如何在 pgactive 可用的两个 RDS for PostgreSQL 数据库实例之间启动主动-主动复制。要运行多区域高可用性示例，您需要在两个不同的区域中部署 Amazon RDS for PostgreSQL 实例，并设置 VPC 对等。有关更多信息，请参阅 VPC 对等。

> **Note**
> 在多个区域之间发送流量可能会产生额外费用。

这些步骤假定已使用 pgactive 扩展启用 RDS for PostgreSQL 数据库实例。有关更多信息，请参阅初始化 pgactive 扩展功能。

使用 **pgactive** 扩展配置第一个 RDS for PostgreSQL 数据库实例

以下示例说明如何创建 pgactive 组，以及在 RDS for PostgreSQL 数据库实例上创建 pgactive 扩展所需的其它步骤。

1. 使用 psql 或其它客户端工具连接第一个 RDS for PostgreSQL 数据库实例。

```
psql --host=firstinstance.111122223333.aws-region.rds.amazonaws.com --port=5432 --
username=postgres --password=PASSWORD --dbname=postgres
```

2. 使用以下命令在 RDS for PostgreSQL 实例上创建数据库：

```
postgres=> CREATE DATABASE app;
```

3. 使用以下命令将连接切换到新数据库：

```
\c app
```

4. 使用以下 SQL 语句创建并填充示例表：

   a. 使用以下 SQL 语句创建一个示例表。

   ```
   app=> CREATE SCHEMA inventory;
   CREATE TABLE inventory.products (
   id int PRIMARY KEY, product_name text NOT NULL,
   created_at timestamptz NOT NULL DEFAULT CURRENT_TIMESTAMP);
   ```

   b. 使用以下 SQL 语句用一些示例数据填充表。

   ```
   app=> INSERT INTO inventory.products (id, product_name)
   VALUES (1, 'soap'), (2, 'shampoo'), (3, 'conditioner');
   ```

   c. 使用以下 SQL 语句验证表中是否存在数据。

   ```
    app=>SELECT count(*) FROM inventory.products;

    count
   -------
    3
   ```

5. 在现有数据库上创建 pgactive 扩展。

```
app=> CREATE EXTENSION pgactive;
```

6. 要安全地创建并初始化 pgactive 组，请使用以下命令：

```
app=>
-- connection info for endpoint1
CREATE SERVER pgactive_server_endpoint1
    FOREIGN DATA WRAPPER pgactive_fdw
    OPTIONS (host '<endpoint1>', dbname 'app');
CREATE USER MAPPING FOR postgres
    SERVER pgactive_server_endpoint1
    OPTIONS (user 'postgres', password '<password>');
      -- connection info for endpoint2
CREATE SERVER pgactive_server_endpoint2
    FOREIGN DATA WRAPPER pgactive_fdw
    OPTIONS (host '<endpoint2>', dbname 'app');
CREATE USER MAPPING FOR postgres
    SERVER pgactive_server_endpoint2
    OPTIONS (user 'postgres', password '<password>');
```

现在，您可以初始化复制组并添加此第一个实例：

```

SELECT pgactive.pgactive_create_group(
    node_name := 'endpoint1-app',
    node_dsn := 'user_mapping=postgres
 pgactive_foreign_server=pgactive_server_endpoint1'

);
```

使用以下命令作为替代方案（安全性较低）来创建并初始化 pgactive 组：

```
app=> SELECT pgactive.pgactive_create_group(
    node_name := 'node1-app',
    node_dsn := 'dbname=app host=firstinstance.111122223333.aws-
region.rds.amazonaws.com user=postgres password=PASSWORD');
```

node1-app 是您分配的名称，用于唯一标识 `pgactive` 组中的节点。

> **Note**
>
> 要在可公开访问的数据库实例上成功执行此步骤，必须通过将 `rds.custom_dns_resolution` 参数设置为 1 将其开启。

7. 要检查数据库实例是否就绪，请使用以下命令：

```
app=> SELECT pgactive.pgactive_wait_for_node_ready();
```

如果命令成功，您可以看到以下输出：

```

pgactive_wait_for_node_ready
------------------------------
(1 row)
```

配置第二个 RDS for PostgreSQL 实例并将其加入 **pgactive** 组

以下示例说明如何将 RDS for PostgreSQL 数据库实例加入 pgactive 组，以及在数据库实例上创建 pgactive 扩展所需的其它步骤。

这些步骤假设已使用 pgactive 扩展设置了其它 RDS for PostgreSQL 数据库实例。有关更多信息，请参阅 初始化 pgactive 扩展功能。

1. 使用 psql 连接到要从发布者接收更新的实例。

```
psql --host=secondinstance.111122223333.aws-region.rds.amazonaws.com --port=5432 --
username=postgres --password=PASSWORD --dbname=postgres
```

2. 使用以下命令在第二个 RDS for PostgreSQL 数据库实例上创建数据库：

```
postgres=> CREATE DATABASE app;
```

3. 使用以下命令将连接切换到新数据库：

```
\c app
```

4. 在现有数据库上创建 pgactive 扩展。

```
app=> CREATE EXTENSION pgactive;
```

5. 使用以下命令以更安全的方式将 RDS for PostgreSQL 第二个数据库实例加入 `pgactive` 组：

```
-- connection info for endpoint1
CREATE SERVER pgactive_server_endpoint1
    FOREIGN DATA WRAPPER pgactive_fdw
    OPTIONS (host '<endpoint1>', dbname 'app');
CREATE USER MAPPING FOR postgres
    SERVER pgactive_server_endpoint1
    OPTIONS (user 'postgres', password '<password>');

-- connection info for endpoint2
CREATE SERVER pgactive_server_endpoint2
    FOREIGN DATA WRAPPER pgactive_fdw
    OPTIONS (host '<endpoint2>', dbname 'app');
CREATE USER MAPPING FOR postgres
    SERVER pgactive_server_endpoint2
    OPTIONS (user 'postgres', password '<password>');
```

```
SELECT pgactive.pgactive_join_group(
    node_name := 'endpoint2-app',
    node_dsn := 'user_mapping=postgres
 pgactive_foreign_server=pgactive_server_endpoint2',
    join_using_dsn := 'user_mapping=postgres
 pgactive_foreign_server=pgactive_server_endpoint1'
);
```

使用以下命令作为替代方案（安全性较低）将 RDS for PostgreSQL 第二个数据库实例加入 `pgactive` 组

```
app=> SELECT pgactive.pgactive_join_group(
node_name := 'node2-app',
node_dsn := 'dbname=app host=secondinstance.111122223333.aws-
region.rds.amazonaws.com user=postgres password=PASSWORD',
```

```
join_using_dsn := 'dbname=app host=firstinstance.111122223333.aws-
region.rds.amazonaws.com user=postgres password=PASSWORD');
```

node2-app 是您分配的名称，用于唯一标识 pgactive 组中的节点。

6. 要检查数据库实例是否就绪，请使用以下命令：

```
app=> SELECT pgactive.pgactive_wait_for_node_ready();
```

如果命令成功，您可以看到以下输出：

```

pgactive_wait_for_node_ready
------------------------------
(1 row)
```

如果第一个 RDS for PostgreSQL 数据库相对较大，则可以看到 pgactive.pgactive_wait_for_node_ready() 正在发出还原操作的进度报告。输出看上去类似于以下内容：

```
NOTICE:  restoring database 'app', 6% of 7483 MB complete
NOTICE:  restoring database 'app', 42% of 7483 MB complete
NOTICE:  restoring database 'app', 77% of 7483 MB complete
NOTICE:  restoring database 'app', 98% of 7483 MB complete
NOTICE:  successfully restored database 'app' from node node1-app in
 00:04:12.274956
 pgactive_wait_for_node_ready
------------------------------
(1 row)
```

此后，pgactive 将在两个数据库实例之间同步数据。

7. 您可以使用以下命令来验证第二个数据库实例的数据库是否具有数据：

```
app=> SELECT count(*) FROM inventory.products;
```

如果数据成功同步，您将看到以下输出：

```

 count
-------
```

```
 3
```

8. 运行以下命令插入新值：

```
app=> INSERT INTO inventory.products (id, product_name) VALUES (4, 'lotion');
```

9. 连接到第一个数据库实例的数据库并运行以下查询：

```
app=> SELECT count(*) FROM inventory.products;
```

如果已初始化主动-主动复制，则输出类似于以下内容：

```
count
-------
 4
```

从 **pgactive** 组中分离和移除数据库实例

您可以使用以下步骤从 pgactive 组中分离和移除数据库实例：

1. 您可以使用以下命令将第二个数据库实例与第一个数据库实例分离：

```
app=> SELECT * FROM pgactive.pgactive_detach_nodes(ARRAY['node2-app']);
```

2. 使用以下命令从第二个数据库实例中移除 pgactive 扩展：

```
app=> SELECT * FROM pgactive.pgactive_remove();
```

要强制移除此扩展，请执行以下操作：

```
app=> SELECT * FROM pgactive.pgactive_remove(true);
```

3. 使用以下命令删除扩展：

```
app=> DROP EXTENSION pgactive;
```

衡量 pgactive 成员之间的复制延迟

您可以使用以下查询来查看 pgactive 成员之间的复制延迟。在每个 pgactive 节点上运行此查询以了解全部信息。

```
app=> SELECT * FROM pgactive.pgactive_get_replication_lag_info();
#-[ RECORD 1 ]--------+---------------------------------------------
#node_name            | node2-app
#node_sysid           | 7481018224801653637
#application_name     | pgactive:7481018224801653637:send
#slot_name            | pgactive_16385_7481018224801653637_0_16385__
#active               | t
#active_pid           | 783486
#pending_wal_decoding | 0
#pending_wal_to_apply | 0
#restart_lsn          | 0/2108150
#confirmed_flush_lsn  | 0/2154690
#sent_lsn             | 0/2154690
#write_lsn            | 0/2154690
#flush_lsn            | 0/2154690
#replay_lsn           | 0/2154690
#-[ RECORD 2 ]--------+---------------------------------------------
#node_name            | node1-app
#node_sysid           | 7481018033434600853
#application_name     | pgactive:7481018033434600853:send
#slot_name            | pgactive_16385_7481018033434600853_0_16385__
#active               | t
#active_pid           | 783488
#pending_wal_decoding | 0
#pending_wal_to_apply | 0
#restart_lsn          | 0/20F5AD0
#confirmed_flush_lsn  | 0/214EF68
#sent_lsn             | 0/214EF68
#write_lsn            | 0/214EF68
#flush_lsn            | 0/214EF68
#replay_lsn           | 0/214EF68
```

至少需要监控以下诊断信息：

active

在 active 为 false 时设置警报，这表示该插槽当前未被使用（订阅用户实例已断开与发布者的连接）。

pending_wal_decoding

在 PostgreSQL 的逻辑复制中，WAL 文件以二进制格式存储。发布者必须解码这些 WAL 更改并将其转换为逻辑更改（例如插入、更新或删除操作）。

指标 pending_wal_decoding 显示发布者端等待解码的 WAL 文件的数量。

该数量可能因以下因素增大：

- 当未连接订阅用户时，active 状态将为 false，并且 pending_wal_decoding 将增大
- 该插槽处于 active 状态，但发布者无法跟上 WAL 更改的量

pending_wal_to_apply

指标 pending_wal_apply 表示订阅用户端等待应用的 WAL 文件的数量。

有以下几个因素可能会阻止订阅用户应用更改，并且可能导致出现磁盘已满的情况：

- 架构差异 – 例如，当您在名为 sample 的表的 WAL 流中做出了更改，但订阅用户端却不存在该表时
- 主键列中的值已更新
- 辅助唯一索引可能会导致数据差异

配置 pgactive 扩展的参数设置

您可以使用以下查询来查看与 `pgactive` 扩展关联的所有参数。

```
app=> SELECT * FROM pg_settings WHERE name LIKE 'pgactive.%';
```

您可以使用各种参数配置 `pgactive` 扩展。可通过 Amazon Web Services Management Console或 Amazon CLI 界面设置这些参数。

pgactive 扩展的主要参数

下表提供了 `pgactive` 扩展的主要参数的参考：

参数	单位	默认值	描述
pgactive.conflict_logging_include_tuples	boolean	–	记录 pgactive 扩展的完整元组信息。 Note 要使更改生效，必须先重启服务器。
pgactive.log_conflicts_to_table	boolean	–	确定 pgactive 扩展是否已将检测到的冲突记录到 pgactive.pgactive_conflict_history 表中。有关更多信息，请参阅冲突日志记录。 Note 要使更改生效，必须先重启服务器。
pgactive.log_conflicts_to_logfile	boolean	–	确定 pgactive 扩展是否已将检测到的冲突记录到 PostgreSQL 日志文件中。有

参数	单位	默认值	描述
			关更多信息，请参阅冲突日志记录。 Note 要使更改生效，必须先重启服务器。

参数	单位	默认值	描述
pgactive.synchronous_commit	boolean	off	确定 pgactive 应用工作线程的提交行为。在此设置处于禁用（off）状态时，应用工作线程会执行异步提交，这将提高 PostgreSQL 在应用操作期间的吞吐量，但会延迟向上游发送的重播确认。将此设置设为 off 始终是安全的，不会导致事务丢失或跳过。此设置仅影响下游节点上的磁盘刷新时间以及向上游发送确认的时间。系统会延迟发送重播刷新确认，直到通过检查点或定期工作等不相关的操作将提交刷新到磁盘。但是，如果上游具有 synchronous_standby_names 中列出的下游，则将此设置设定为 off 会导致上游

参数	单位	默认值	描述
			的同步提交需要更长的时间才能向客户端返回成功响应。在此情况下，可将该参数设置为 on。 **Note** 即使将该参数设置为 on 并在 synchronous_standby_names 中列出了节点，主动-主动配置中也仍然可能发生复制冲突。这是因为系统缺少节点间锁定和全局快照管理，可能导致不同节点上的并发事务修改同一元组。此外，

参数	单位	默认值	描述
			事务仅在上游节点上提交后才开始复制。启用同步提交并不会将 pgactive 扩展转换为始终一致的系统。
`pgactive.temp_dump_directory`	`string`	–	定义数据库克隆操作在初始设置期间所需的临时存储路径。此目录必须可由 postgres 用户写入，并且拥有足够的存储空间来容纳完整的数据库转储。系统仅在通过逻辑复制操作进行初始数据库设置期间使用此位置。`pgactive_init_copy` command 不使用该参数。

参数	单位	默认值	描述
pgactive.max_ddl_lock_delay	milliseconds	-1	指定在强制中止并发写入事务之前，DDL 锁的最长等待时间。默认值为 -1，这将使用 max_standby_streaming_delay 中设置的值。该参数接受时间单位。例如，可以将它设置为 10s 来表示 10 秒。在此等待期间，系统会尝试获取 DDL 锁，同时等待正在进行的写入事务完成提交或回滚。有关更多信息，请参阅“DDL 锁定”。

参数	单位	默认值	描述
pgactive.ddl_lock_timeout	milliseconds	-1	指定 DDL 锁为获取锁而尝试等待的时长。默认值为 -1，这将使用 lock_timeout 中指定的值。您可以使用时间单位来设置此参数，例如 10s 表示 10 秒。此计时器仅控制为获取 DDL 锁而等待的时长。在系统获取锁并开始 DDL 操作后，计时器将停止。此参数不会限制可将 DDL 锁保持的总持续时间或总 DDL 操作时间。要控制操作的总持续时间，请改用 statement_timeout 。有关更多信息，请参阅“DDL 锁定”。

参数	单位	默认值	描述
pgactive.debug_trace_ddl_locks_level	boolean	–	覆盖 pgactive 扩展中 DDL 锁定操作的默认调试日志级别。配置此设置后，会导致系统在 LOG 调试级别而非默认级别发出与 DDL 锁相关的消息。使用此参数可监控 DDL 锁定活动，而无需在整个服务器上启用详细 DEBUG1 或 DEBUG2 日志级别。 可用日志级别（按详细程度升序排列）： • none – DDL 锁消息仅在 DEBUG1 及更低的服务器日志级别显示。 • statement – 为 DDL 锁获取尝试添加 LOG 输出。 • acquire_release – 记录远程 DDL 锁的锁获取、释放、拒绝事件

参数	单位	默认值	描述
			以及对等节点应用。 • peers – 提供有关对等节点之间的 DDL 锁协商的更多详细信息。 • debug – 在 LOG 级别记录所有与 DDL 锁相关的活动。 有关监控选项的更多信息，请参阅“监控全局 DDL 锁”。 Note 对此设置所做的更改会在重新加载配置后生效。您无需重新启动服务器。

pgactive 扩展的其他参数

下表列出了 `pgactive` 扩展中不常用的配置选项和可用的内部配置选项。

参数	单位	默认值	描述
pgactive.debug_apply_delay	integer	–	为 pgactive.pgactive_connections 条目中未明确设定应用延迟的已配置连接设置应用延迟（以毫秒为单位）。此延迟是在节点创建或加入时设置的，pgactive 不会在对等节点上重播事务，直到提交后至少经过指定的毫秒数。 主要用于在测试环境中模拟高延迟网络，以便更轻松地制造冲突。例如，如果在节点 A 和 B 上设置了 500 毫秒的延迟，则在节点 A 上插入一个值后，至少为您提供了 500 毫秒的时间在节点 B 上执行冲突插入。

参数	单位	默认值	描述
			Note 需要重新加载服务器或重新启动应用工作线程，设置才能生效。
pgactive.connectability_check_duration	integer	–	指定数据库工作线程在尝试连接失败后，继续尝试建立连接的持续时间（以秒为单位）。工作线程每秒尝试连接一次，直到连接成功或达到此超时值。当数据库引擎在工作线程准备好建立连接之前启动时，此设置很有用。

参数	单位	默认值	描述
pgactive.skip_ddl_replication	boolean	on	控制如何在启用 pgactive 时，在 Amazon RDS 中复制或处理 DDL 更改。在设置为 on 时，该节点会像非 pgactive 节点一样处理 DDL 更改。在使用此参数时，需符合以下要求： • 如果新节点与上游节点的 skip_ddl_replication 值不同，则新节点无法加入 pgactive 组。 • 如果现有节点与上游节点的参数值不匹配，则现有节点无法启动 pgactive 工作线程。 • 所有 pgactive 成员都必须使用同一个参数值。

参数	单位	默认值	描述
			您可以使用超级用户权限以全局和本地（会话级别）这两种方式修改此参数。 Note 错误地更改此参数可能会中断您的复制设置。
pgactive.do_not_replicate	boolean	–	此参数仅供内部使用。在事务中设置此参数时，所做的更改不会复制到数据库集群内的其他节点。 Note 错误地更改此参数可能会中断您的复制设置。

参数	单位	默认值	描述
pgactive.discard_mismatched_row_attributes	boolean	–	此参数仅供专业人员使用。建议仅在解决特定的复制问题时使用此参数。在以下情况下可使用此参数： • 传入的复制流包含的行所对应的列比本地表多。 • 这些远程行包含非空值。 此设置会覆盖以下错误消息，并允许产生数据差异，以便让复制进程持续进行：cannot right-pad mismatched attributes; attno %u is missing in local table and remote row has non-null, non-dropped value for this attribute

参数	单位	默认值	描述
			Note 错误地更改此参数可能会中断您的复制设置。

参数	单位	默认值	描述
pgactive.debug_trace_replay	boolean	–	在设置为 on 时，它会为下游应用工作线程处理的每个远程操作发出一条日志消息。日志包括： • 更改类型 • 受影响的表名称 • 自事务开始以来的更改次数 • 事务提交 LSN • 提交时间戳 • 上游节点标识符 • 转发节点标识符（如果适用） 日志还会捕获已排队的 DDL 命令和表删除事件。para> 默认情况下，日志不包括行字段内容。要在日志中包含行值，您必须在启用以下标志的情况下重新编译：

参数	单位	默认值	描述
			• VERBOSE_INSERT • VERBOSE_UPDATE • VERBOSE_DELETE Note 启用此日志记录设置可能会影响性能。建议仅在需要此设置来进行问题排查时启用它。对此设置所做的更改会在重新加载配置后生效。您无需重新启动服务器。

参数	单位	默认值	描述
`pgactive.extra_apply_connection_options`		–	您可以为所有与 pgactive 节点的对等节点连接配置连接参数。这些参数可控制 keepalives 和 SSL 模式等设置。默认情况下，pgactive 使用以下连接参数： • connect_timeout=30 • keepalives=1 • keepalives_idle=20 • keepalives_interval=20 • keepalives_count=5 要覆盖默认参数，请使用以下类似命令：`pgactive.extra_apply_connection_options = 'keepalives=0'`

参数	单位	默认值	描述
			单个节点连接字符串优先于这些设置和 pgactive 的内置连接选项。有关连接字符串格式的更多信息，请参阅 libpq connection strings。 我们建议将默认的 keepalive 设置保持启用状态。仅在您通过不可靠的网络完成大量事务时遇到问题的情况下，才禁用 keepalives。 Note 我们建议将默认的 keepalive 设置保持启用状态。仅在您通过不可靠的网络完成大量事务时遇到问题的情况下，

参数	单位	默认值	描述
			才禁用 keepalives。对此设置所做的更改会在重新加载配置后生效。您无需重新启动服务器。
`pgactive.init_node_parallel_jobs (int)`		–	指定在逻辑节点与 `pgactive.pgactive_join_group` 函数联接期间，pg_dump 和 `pg_restore` 可使用的并行作业数。 对此设置所做的更改会在重新加载配置后生效。您无需重新启动服务器。

参数	单位	默认值	描述
pgactive.max_nodes	int	4	指定 pgactive 扩展组中可包含的节点的最大数目。默认值为 4 个节点。在设置此参数的值时，必须注意以下几点： • pgactive 扩展组中的所有节点都必须使用同一个参数值。 • 如果新节点与上游节点的参数值不同，则新节点无法联接。 • 如果现有节点与上游节点的参数值不匹配，则现有节点无法启动 pgactive 工作线程。 • 由于较大的组所需的监控和维护工作量更大，因此需谨慎设置该参数的值。

参数	单位	默认值	描述
			可通过两种方式设置此参数，即在配置文件中进行设置或者使用 ALTER SYSTEM SET 命令进行设置 此参数的默认值为 4，这意味着，pgactive 扩展组始终最多只能包含 4 个节点。 **Note** 重新启动服务器后，此更改将生效。
pgactive.permit_node_identifier_getter_function_creation	boolean	–	此参数仅供内部使用。启用后，pgactive 扩展将允许创建 pgactive 节点标识符 getter 函数。

了解双活冲突

如果您在双活模式下使用 pgactive，在从多个节点写入同一个表时可能会造成数据冲突。一些集群化系统会使用分布式锁来防止并发访问，不过 pgactive 采用的是乐观方法，更适合分布在不同地理位置的应用程序。

一些数据库集群系统使用分布式锁来防止并发数据访问。但是，此方法只适合距离非常近的服务器，不支持分布在不同地理位置的应用程序，因为此方法需要极低的延迟才能实现良好的性能。pgactive 扩展没有使用分布式锁（这是一种悲观方法），而是使用乐观方法。这意味着：

- 可帮助您尽可能避免冲突。
- 允许出现某些类型的冲突。
- 发生冲突时会提供冲突解决方案。

通过这种方法，您在构建分布式应用程序时具有更好的灵活性。

发生冲突的原因

节点间冲突是由事件序列引起的，如果涉及的所有事务同时发生在同一个节点上，则不会出现节点间冲突。由于节点仅在事务提交后交换更改，因此每个事务在其提交的节点上都是分别有效的，但如果事务在另一个同时完成其他工作的节点上运行，就会无效。由于 pgactive 的应用操作本质上是在其他节点上重播事务，因此，如果正在应用的事务与在接收节点上提交的事务之间存在冲突，则重播操作会失败。

当所有事务都运行在单个节点上时，大多数冲突不会发生的原因是 PostgreSQL 采用了事务间通信机制来防止冲突，包括：

- UNIQUE 索引
- SEQUENCE
- 行和关系锁定
- SERIALIZABLE 依赖关系跟踪

所有这些机制都是在事务之间进行通信的方法，用于防止出现意外并发问题

pgactive 不使用分布式事务管理器或锁管理器，因此可以实现低延迟并很好地处理网络分区。但是，这意味着不同节点上事务的运行是彼此完全隔离的。虽然隔离通常可以提高数据库的一致性，但在这种情况下，您需要减少隔离以防止冲突。

冲突类型

可能发生的冲突包括：

主题

PRIMARY KEY 或 UNIQUE 冲突

当多个操作试图以单个节点上不可能出现的方式修改同一个行键时，就会发生行冲突。这些冲突体现了最常见的数据冲突类型。

pgactive 采用上次更新获胜处理方法或者您的自定义冲突处理程序，来解决检测到的冲突。

行冲突包括：

- INSERT 与 INSERT
- INSERT 与 UPDATE
- UPDATE 与 DELETE
- INSERT 与 DELETE
- DELETE 与 DELETE

- INSERT 与 DELETE

INSERT/INSERT 冲突

这是最常见的冲突，当两个不同节点上的 INSERT 操作创建具有相同 PRIMARY KEY 值的元组（或在没有 PRIMARY KEY 时，使用相同的 UNIQUE 约束值），就会发生这种冲突。

pgactivelink 使用来自原始主机的时间戳来保留最新的元组，以此解决 INSERT 冲突。您可以使用自定义冲突处理程序覆盖此默认行为。虽然此过程不需要管理员专门采取操作，但请注意，pgactivelink 会丢弃所有节点上的其中一个 INSERT 操作。除非您的自定义处理程序实施自动数据合并，否则不会自动进行数据合并。

pgactivelink 只能解决涉及单个约束违规的冲突。如果某个 INSERT 违反了多个 UNIQUE 约束，您必须实施其他冲突解决策略。

违反多个 UNIQUE 约束的 INSERT

INSERT/INSERT 冲突可能违反多个 UNIQUE 约束，包括 PRIMARY KEY。pgactivelink 只能处理涉及单个 UNIQUE 约束的冲突。当冲突违反多个 UNIQUE 约束时，应用工作线程会失败并返回以下错误：

```
multiple unique constraints violated by remotely INSERTed tuple.
```

在较早的版本中，这种情况生成的是“分歧唯一性冲突”错误。

要解决这些冲突，您必须手动采取操作。您可以对发生冲突的本地元组执行 DELETE，或者对元组执行 UPDATE 以使用新的远程元组来消除冲突。请注意，您可能需要解决多个相互冲突的元组。目前，pgactivelink 没有提供内置功能用于忽略、丢弃或合并违反多个唯一约束的元组。

> **Note**
>
> 有关更多信息，请参阅“违反多个 UNIQUE 约束的 UPDATE”。

UPDATE/UPDATE 冲突

当两个节点并发修改同一个元组而没有更改其 PRIMARY KEY 时，就会发生此冲突。pgactivelink 使用上次更新获胜逻辑或您的自定义冲突处理程序（如果已定义）来解决这些冲突。PRIMARY KEY 对于元组匹配和冲突解决至关重要。对于没有 PRIMARY KEY 的表，pgactivelink 拒绝 UPDATE 操作，并显示以下错误：

```
Cannot run UPDATE or DELETE on table (tablename) because it does not have a
primary key.
```

PRIMARY KEY 上的 UPDATE 冲突

pgactive 在处理 PRIMARY KEY 更新时存在限制。虽然您可以对 PRIMARY KEY 执行 UPDATE 操作，但对于这些操作，pgactive 无法使用最后更新获胜逻辑自动解决冲突。您必须确保 PRIMARY KEY 更新不会与现有值冲突。如果在 PRIMARY KEY 更新期间发生冲突，这些冲突就会成为分歧冲突，需要您手动干预。有关处理这些情况的更多信息，请参阅分歧冲突。

违反多个 UNIQUE 约束的 UPDATE

当传入的 UPDATE 违反多个 UNIQUE 约束或 PRIMARY KEY 值时，pgactivelink 无法应用上次更新获胜冲突解决方案。此行为类似于出现多个约束冲突的 INSERT 操作。这些情况会造成分歧冲突，需要您手动干预。有关更多信息，请参阅分歧冲突。

UPDATE/DELETE 冲突

当一个节点对一行执行 UPDATE 而另一个节点同时对该行执行 DELETE 时，就会出现此冲突。在这种情况下，重播时会发生 UPDATE/DELETE 冲突。除非您的自定义冲突处理程序另行规定，否则解决方案是丢弃在 DELETE 之后到达的所有 UPDATE。

pgactivelink 需要一个 PRIMARY KEY 来匹配元组并解决冲突。对于没有 PRIMARY KEY 的表，表会拒绝 DELETE 操作，并显示以下错误：

```
Cannot run UPDATE or DELETE on table (tablename) because it does not have a
primary key.
```

> **Note**
>
> pgactivelink 无法区分 UPDATE/DELETE 和 INSERT/UPDATE 冲突。在这两种情况下，UPDATE 均会影响不存在的行。由于异步复制以及节点之间缺乏重播排序，pgactivelink 无法确定是在对新行（尚未收到 INSERT）还是对已删除行执行 UPDATE 操作。在这两种情况下，pgactivelink 都会丢弃 UPDATE。

INSERT/UPDATE 冲突

这种冲突可能发生在多节点环境中。当一个节点 INSERT 一行数据，第二个节点对该行执行 UPDATE，而第三个节点在收到原始 INSERT 之前就接收到 UPDATE 时，就会发生这种情况。默认情况下，除非您的自定义冲突触发器另行规定，否则 pgactivelink 会通过放弃 UPDATE 来解决这些冲

突。请注意，这种解决方法可能会导致节点间的数据不一致。有关类似场景及其处理方法的更多信息，请参阅 UPDATE/DELETE 冲突。

DELETE/DELETE 冲突

当两个不同的节点并发删除同一个元组时，就会发生这种冲突。pgactivelink 认为这些冲突无害，因为两个 DELETE 操作会产生相同的最终结果。在此场景中，pgactivelink 可以安全地忽略其中一个 DELETE 操作，而不会影响数据一致性。

外键约束冲突

对现有本地数据应用远程事务时，FOREIGN KEY 约束可能会导致冲突。发生这些冲突的情况通常是应用事务的顺序，不同于事务在发起节点上的逻辑顺序。

默认情况下，pgactive 会将 session_replication_role 作为 `replica` 来应用更改，这会在复制过程中绕过外键检查。在双活配置中，这可能会导致外键违规。大多数违规行为都是暂时性的，当复制进度跟上时就会解决。但是，由于 pgactive 不支持跨节点行锁定，因此可能会出现悬挂外键。

这种行为是具备分区容错性的异步双活系统所固有的。例如，节点 A 可能会插入新的子行，而节点 B 同时在删除其父行。系统无法阻止节点之间这种类型的并发修改。

为尽可能减少外键冲突，我们给出了以下建议：

- 将外键关系限制为密切相关的实体。
- 尽可能从单个节点修改相关实体。
- 选择极少需要进行修改的实体。
- 针对修改实施应用程序级的并发控制。

排除约束冲突

pgactivelink 不支持排除约束，并会限制创建这种约束。

> **Note**
> 在将现有的独立数据库转换为 pgactivelink 数据库时，请手动删除所有排除约束。

在分布式异步系统中，不可能保证所有行集都没有违反约束。这是因为不同节点上的所有事务都是完全隔离的。排除约束可能导致重播死锁，在这种情况下，由于排除约束违规，重播无法从任何节点进展到另一个节点。

如果您强制 pgactivelink 创建排除约束，或者在将独立数据库转换到 pgactivelink 时不删除现有排除约束，则复制很可能会中断。要恢复复制进度，请删除或更改与传入的远程元组冲突的本地元组，这样便可以应用远程事务。

全局数据冲突

使用 pgactivelink 时，当节点具有不同的全局 PostgreSQL 系统级数据（例如角色）时，就可能会发生冲突。这些冲突的结果是操作（主要是 DDL）会成功并在一个节点上提交，但无法应用于其他节点。

如果一个节点上有某个用户但另一个节点上没有，就会出现复制问题：

- Node1 有名为 `fred` 的用户，但是 Node2 上没有该用户
- 当 `fred` 在 Node1 上创建表时，表在复制时的所有者是 `fred`
- 将此 DDL 命令应用于 Node2 时，命令会失败，因为用户 `fred` 不存在
- 在失败后，会在 Node2 上的 PostgreSQL 日志中生成 ERROR 条目并递增 `pgactive.pgactive_stats.nr_rollbacks` 计数器

解决方案：在 Node2 上创建用户 `fred`。该用户不必具备相同的权限，但必须存在于两个节点上。

如果一个节点上有某个表但另一个节点上没有，则数据修改操作会失败：

- Node1 上有名为 `foo` 的表，但 Node2 上没有
- Node1 上对 `foo` 表执行的任何 DML 操作，在复制到 Node2 时会失败

解决方案：在 Node2 上创建具有相同结构的表 `foo`。

> **Note**
>
> pgactivelink 目前不复制 CREATE USER 命令或 DDL 操作。DDL 复制功能计划在未来版本中发布。

锁冲突和死锁中止

由于 pgactive 应用进程的运行方式类似于普通用户会话，因此这些进程遵循标准的行和表锁定规则。这可能会导致 pgactivelink 应用进程需要等待用户事务或其他应用进程持有的锁被释放。

以下类型的锁会影响应用进程：

- 用户会话施加的显式表级锁（LOCK TABLE...）
- 用户会话施加的显式行级锁（SELECT ... FOR UPDATE/FOR SHARE）
- 来自外键的锁定
- 由于行 UPDATE、INSERT 或 DELETE 而导致的隐式锁定，这可能源自本地活动，也可能是从其他服务器应用的

在下列操作之间可能会出现死锁：

- pgactivelink 应用进程和用户事务
- 两个应用进程

发生死锁时，PostgreSQL 的死锁检测器会终止其中一个出现问题的事务。如果 pgactivelink 应用工作线程的进程终止，则会自动重试而且通常会成功。

> **Note**
>
> - 这些问题是暂时性的，通常无需管理员干预。如果由于空闲用户会话锁定导致某个应用进程被长时间阻止，您可以终止该用户会话来恢复复制。这种情况类似于某个用户持有的长时间锁定影响到了其他用户会话。
> - 要识别与锁定相关的重播延迟，请在 PostgreSQL 中启用 `log_lock_waits` 功能。

分歧冲突

当各节点之间本应相同的数据意外地变得不同时，就会发生分歧冲突。虽然这些冲突不应该发生，但在当前的实施中，无法可靠地预防所有冲突。

> **Note**
>
> 在一行上，如果有某个节点在所有节点处理更改之前更改该行的主键，则修改行的 PRIMARY KEY 会导致分歧冲突。请避免更改主键，或将更改限制在某个指定节点上。有关更多信息，请参阅[the section called “PRIMARY KEY 上的 UPDATE 冲突”]。

涉及行数据的分歧冲突通常需要管理员干预。要解决这些冲突，您必须手动调整一个节点上的数据来与其他节点相匹配，同时使用 `pgactive.pgactive_do_not_replicate` 暂时禁用复制。如果您按照文档说明使用 pgactive 并避免使用标记为不安全的设置或函数，则不应出现这些冲突。

作为管理员，您必须手动解决这些冲突。根据冲突类型，您需要使用诸如 `pgactive.pgactive_do_not_replicate` 之类的高级选项。请谨慎使用这些选项，因为使用不当会导致情况恶化。由于可能出现的冲突各种各样，我们无法提供通用的解决方案说明。

当不同节点之间本应相同的数据意外地变得不同时，就会发生分歧冲突。虽然这些冲突不应该发生，但在当前的实施中，无法可靠地预防所有此类冲突。

避免或容忍冲突

在大多数情况下，您可以使用合适的应用程序设计，从而避免冲突或使应用程序具备容忍冲突的能力。

只有在多个节点上同时进行多个操作时才会发生冲突。为避免冲突，请采取以下方法：

- 仅向一个节点写入
- 写入各个节点上的独立数据库子集（例如，向每个节点分配一个单独的架构）

对于 INSERT 与 INSERT 冲突，请使用全局序列来彻底防止冲突。

如果您的应用场景不能接受冲突，请考虑在应用程序级别实施分布式锁定。通常，最好的方法是对应用程序进行设计，使其与 pgactive 的冲突解决机制配合使用，而不是尝试预防所有冲突。有关更多信息，请参阅the section called “冲突类型”。

冲突日志记录

pgactivelink 会将冲突事件记录到 `pgactive.pgactive_conflict_history` 表中，供您诊断和处理双活冲突。仅当 `pgactive.log_conflicts_to_table` 设置为 true 时，才会将冲突记录到此表。无论 `pgactive.log_conflicts_to_table` 的设置如何，当 log_min_messages 设置为 `LOG` 或 `lower` 时，pgactive 扩展还会将冲突记录到 PostgreSQL 日志文件中。

使用冲突历史记录表可以：

- 衡量您的应用程序造成冲突的频率
- 确定在什么位置发生冲突
- 改进应用程序以降低冲突率
- 检测冲突解决方案无法达成所需结果的情况
- 确定哪些地方需要用户定义的冲突触发器或更改应用程序设计

对于行冲突，您可以选择记录行值。这由 `pgactive.log_conflicts_to_table` 设置控制。请注意：

- 这是一个全局数据库级选项
- 无法按各个表来控制对行值的记录
- 字段数量、数组元素或字段长度没有限制
- 如果您在处理数 MB 的行而这些行可能会触发冲突，则不建议启用此功能

由于冲突历史记录表包含来自数据库中各个表的数据（每个表都可能采用不同架构），因此记录的行值存储为 JSON 字段。JSON 使用 `row_to_json` 创建，类似于直接从 SQL 中调用。PostgreSQL 不提供 `json_to_row` 函数，因此您需要特定于表的代码（采用 PL/pgSQL、PL/Python、PL/Perl 等）才能从记录的 JSON 中重新构造复合类型的元组。

> **Note**
> 对用户定义冲突的支持计划作为未来的扩展功能推出。

了解 pgactive 架构

pgactive 架构管理 RDS for PostgreSQL 中的主动-主动复制。此架构包含用于存储复制配置和状态信息的表。

> **Note**
> pgactive 架构正在演进中，可能会发生变更。不要直接修改这些表中的数据。

pgactive 架构中的键表包括：

- `pgactive_nodes` – 存储有关主动-主动复制组中的节点的信息。
- `pgactive_connections` – 存储每个节点的连接详细信息。

pgactive_nodes

pgactive_nodes 存储有关参与主动-主动复制组的节点的信息。

列	类型	排序规则	是否可为 null	默认
node_sysid	文本	–	不为 null	–
node_timeline	oid	–	不为 null	–
node_dboid	oid	–	不为 null	–
node_status	char	–	不为 null	–
node_name	文本	–	不为 null	–
node_dsn	文本	–	不为 null	–
node_init _from_dsn	文本	–	不为 null	–
node_read _only	布尔值	–	–	false
node_seq_id	smallint	–	不为 null	–

node_sysid

在 `pgactive_create_group` 或 `pgactive_join_group` 期间生成的节点的唯一 ID

node_status

节点的准备情况：

- b – 开始设置
- i – 正在初始化
- c – 追赶
- o – 创建出站插槽
- r – 已就绪
- k – 已终止

> **Note**
> 此列不指明节点已连接还是已断开连接。

node_name

用户提供的唯一节点名称。

node_dsn

连接字符串或用户映射名称。

node_init_from_dsn

从中创建了此节点的 DSN。

pgactive_connection

pgactive_connections 存储每个节点的连接详细信息。

列	类型	排序规则	是否可为 null	默认
conn_sysid	文本	none	不为 null	none
conn_timeline	oid	none	不为 null	none
conn_dboid	oid	none	不为 null	none
conn_dsn	文本	none	不为 null	none
conn_apply_delay	整数	none	none	none
conn_replication_sets	文本	none	none	none

conn_sysid

此条目所指的节点的节点标识符。

conn_dsn

与 pgactive.pgactive_nodes node_dsn 相同。

conn_apply_delay

如果设置此参数，则将设定从远程节点应用每个事务前的等待时间（以毫秒为单位）。主要用于调试。如果为 null，则全局默认值将适用。

使用复制集

复制集决定要在复制操作中包含或排除的表。默认情况下，除非您使用以下函数另行指定，否则将复制所有表：

- `pgactive_exclude_table_replication_set()` – 从复制中排除指定的表
- `pgactive_include_table_replication_set()` – 在复制中包含指定的表

> **Note**
>
> 在配置复制集之前，请注意以下几点：
>
> - 仅在运行 `pgactive_create_group()` 后且在运行 `pgactive_join_group()` 之前，才能配置表包含或排除。
> - 使用 `pgactive_exclude_table_replication_set()` 后，您将无法使用 `pgactive_include_table_replication_set()`。
> - 使用 `pgactive_include_table_replication_set()` 后，您将无法使用 `pgactive_exclude_table_replication_set()`。

系统会根据您的初始配置通过不同的方式处理新创建的表：

- 如果排除表：在 `pgactive_join_group()` 之后创建的任何新表都将自动包含在复制中。
- 如果包含表：在 `pgactive_join_group()` 之后创建的任何新表都将自动从复制中排除。

要查看特定表的复制集配置，请使用 `pgactive.pgactive_get_table_replication_sets()` 函数。

pgactive 函数参考

在下文中，您可以找到 pgactive 函数及其参数、返回值和实际使用说明，以助您高效使用它们：

get_last_applied_xact_info

检索指定节点的上次应用的事务信息。

Arguments (参数)

- sysid (text) - timeline OID
- dboid (OID)

返回类型

它将记录以下内容：

- last_applied_xact_id（OID）
- last_applied_xact_committs（带时区的时间戳）
- last_applied_xact_at（带时区的时间戳）

使用说明

使用此函数可检索指定节点的上次应用的事务信息。

pgactive_apply_pause

暂停复制应用进程。

Arguments (参数)

无

返回类型

布尔值

使用说明

调用此函数可暂停复制应用进程。

pgactive_apply_resume

恢复复制应用进程。

Arguments (参数)

无

返回类型

void

使用说明

调用此函数可恢复复制应用进程。

pgactive_is_apply_paused

检查复制应用当前是否已暂停。

Arguments (参数)

无

返回类型

布尔值

使用说明

使用此函数可检查复制应用当前是否已暂停。

pgactive_create_group

通过将独立数据库转换为初始节点来创建 pgactive 组。

Arguments (参数)

- node_name (text)
- node_dsn (text)
- apply_delay integer DEFAULT NULL::integer - replication_sets text[] DEFAULT ARRAY['default'::text]

返回类型

void

使用说明

通过将独立数据库转换为初始节点来创建 pgactive 组。此函数在将节点转换为 pgactive 节点之前执行完整性检查。在使用此函数之前，请确保您的 PostgreSQL 集群有足够的可用 `max_worker_processes` 来支持 pgactive 后台工作线程。

pgactive_detach_nodes

从 pgactive 组中移除指定节点。

Arguments (参数)

- p_nodes (text[])

返回类型

void

使用说明

使用此函数可从 pgactive 组中移除指定节点。

pgactive_exclude_table_replication_set

将特定表排除在复制之外。

Arguments (参数)

- p_relation（regclass）

返回类型

void

使用说明

使用此函数可将特定表排除在复制之外。

pgactive_get_replication_lag_info

检索详细的复制滞后信息，包括节点详细信息、WAL 状态和 LSN 值。

Arguments (参数)

无

返回类型

SETOF 记录 - node_name text - node_sysid text - application_name text - slot_name text - active boolean - active_pid integer - pending_wal_decoding bigint - 发送者节点上待解码的 WAL 的近似大小（以字节为单位）- pending_wal_to_apply bigint - 要在接收节点上应用的 WAL 的近似大小（以字节为单位）- restart_lsn pg_lsn - confirmed_flush_lsn pg_lsn - sent_lsn pg_lsn - write_lsn pg_lsn - flush_lsn pg_lsn - replay_lsn pg_lsn

使用说明

调用此函数可检索复制滞后信息，包括节点详细信息、WAL 状态和 LSN 值。

pgactive_get_stats

检索 pgactive 复制统计信息。

Arguments (参数)

无

返回类型

SETOF 记录 - rep_node_id oid - rilocalid oid - riremoteid text - nr_commit bigint - nr_rollback bigint - nr_insert bigint - nr_insert_conflict bigint - nr_update bigint - nr_update_conflict bigint - nr_delete bigint - nr_delete_conflict bigint - nr_disconnect bigint

使用说明

使用此函数可检索 pgactive 复制统计信息。

pgactive_get_table_replication_sets

获取特定关系的复制集配置。

Arguments (参数)

- relation (regclass)

返回类型

SETOF 记录

使用说明

调用此函数可获取特定关系的复制集配置。

pgactive_include_table_replication_set

将特定表包含在复制中。

Arguments (参数)

- p_relation（regclass）

返回类型

void

使用说明

使用此函数可将特定表包含在复制中。

pgactive_join_group

向现有 pgactive 组添加节点。

Arguments (参数)

- node_name (text)
- node_dsn (text)
- join_using_dsn (text)
- apply_delay (integer，可选)
- replication_sets (text[]，默认值：['default'])
- bypass_collation_check (boolean，默认值：false)
- bypass_node_identifier_creation (boolean，默认值：false)
- bypass_user_tables_check (boolean，默认值：false)

返回类型

void

使用说明

调用此函数可将节点添加到现有 pgactive 组中。确保您的 PostgreSQL 集群有足够的 max_worker_processes 以供 pgactive 后台工作线程使用。

pgactive_remove

从本地节点中移除所有 pgactive 组件。

Arguments (参数)

- force (boolean，默认值：false)

返回类型

void

使用说明

调用此函数可从本地节点中移除所有 pgactive 组件。

pgactive_snowflake_id_nextval

生成特定于节点的唯一序列值。

Arguments (参数)

- regclass

返回类型

bigint

使用说明

使用此函数可生成特定于节点的唯一序列值。

pgactive_update_node_conninfo

更新 pgactive 节点的连接信息。

Arguments (参数)

- node_name_to_update (text)
- node_dsn_to_update (text)

返回类型

void

使用说明

使用此函数可更新 pgactive 节点的连接信息。

pgactive_wait_for_node_ready

监控组创建或加入操作的进度。

Arguments (参数)

- timeout (integer，默认值：0)
- progress_interval (integer，默认值：60)

返回类型

void

使用说明

调用此函数可监控组创建或加入操作的进度。

处理主动-主动复制中的冲突

pgactive 扩展适用于每个数据库，而不是每个集群。使用 pgactive 的每个数据库实例都是一个独立的实例，可以接受来自任何来源的数据更改。将更改发送到数据库实例时，PostgreSQL 会在本地提交更改，然后使用 pgactive 将更改异步复制到其它数据库实例。当两个 PostgreSQL 数据库实例几乎同时更新同一记录时，可能会发生冲突。

pgactive 扩展提供了冲突检测和自动解决机制。它跟踪在这两个数据库实例上提交事务的时间戳，并自动应用带有最新时间戳的更改。pgactive 扩展还会记录 pgactive.pgactive_conflict_history 表中发生冲突的时间。

pgactive.pgactive_conflict_history 会继续增加。您可能需要定义清除策略。可以通过定期删除一些记录或为此关系定义分区方案（然后分离、删除、截断感兴趣的分区）来完成此操作。要定期实施清除策略，一种选择是使用 pg_cron 扩展。请参阅 pg_cron 历史记录表示例的以下信息：使用 PostgreSQL pg_cron 扩展安排维护计划。

处理主动-主动复制中的序列

带有 pgactive 扩展的 RDS for PostgreSQL 数据库实例使用两种不同的序列机制来生成唯一值。

全局序列

要使用全局序列，请使用 CREATE SEQUENCE 语句创建一个本地序列。使用 pgactive.pgactive_snowflake_id_nextval(seqname) 而非 usingnextval(seqname) 来获取序列的下一个唯一值。

以下示例创建全局序列：

```
app=> CREATE TABLE gstest (
      id bigint primary key,
      parrot text
    );
```

```
app=>CREATE SEQUENCE gstest_id_seq OWNED BY gstest.id;
```

```
app=> ALTER TABLE gstest \
      ALTER COLUMN id SET DEFAULT \
      pgactive.pgactive_snowflake_id_nextval('gstest_id_seq');
```

分区序列

在分步或分区序列中，每个节点上都使用普通的 PostgreSQL 序列。每个序列的增量相同，从不同的偏移量开始。例如，在步骤 100 中，节点 1 生成序列为 101、201、301，依此类推，而节点 2 生成序列为 102、202、302，依此类推。即使节点长时间无法通信，该方案也能正常工作，但要求设计人员在建立模式时指定最大节点数，并且需要按节点进行配置。错误很容易导致序列重叠。

通过在节点上创建所需的序列，使用 `pgactive` 配置这种方法相对简单，如下所示：

```

CREATE TABLE some_table (generated_value bigint primary key);
```

```
app=> CREATE SEQUENCE some_seq INCREMENT 100 OWNED BY some_table.generated_value;
```

```
app=> ALTER TABLE some_table ALTER COLUMN generated_value SET DEFAULT
 nextval('some_seq');
```

然后，对每个节点调用 `setval` 以给出不同的偏移起始值，如下所示。

```
app=>
-- On node 1
SELECT setval('some_seq', 1);

-- On node 2
```

```
SELECT setval('some_seq', 2);
```

使用 pg_repack 扩展减少表和索引的膨胀

您可以使用 pg_repack 扩展从表和索引中移除多余内容，作为 VACUUM FULL 的备选方法。RDS for PostgreSQL 版本 9.6.3 及更高版本支持该扩展。有关 pg_repack 扩展和完整表重新打包的更多信息，请参阅 GitHub 项目文档。

与 VACUUM FULL 不同，在以下情况下，pg_repack 扩展只需在表重建操作期间短时间使用独占锁（AccessExclusiveLock）：

- 初始创建日志表 - 创建日志表以记录在数据初始复制期间发生的更改，如以下示例所示：

```
postgres=>\dt+ repack.log_*
List of relations
-[ RECORD 1 ]-+----------
Schema        | repack
Name          | log_16490
Type          | table
Owner         | postgres
Persistence   | permanent
Access method | heap
Size          | 65 MB
Description   |
```

- 最后的交换和删除阶段。

在重建操作的其余部分中，它只需对原始表使用 ACCESS SHARE 锁，即可将行从该表复制到新表。这有助于 INSERT、UPDATE 和 DELETE 操作照常进行。

建议

当您使用 pg_repack 扩展从表和索引中移除膨胀内容时，以下建议适用：

- 在非工作时间或在维护时段内执行重新打包，以最大限度地减少它对其它数据库活动性能的影响。
- 在重建活动期间，密切监视阻止会话，并确保原始表上不存在可能阻止 pg_repack 的活动，特别是在最后的交换和删除阶段，此时它需要对原始表进行独占锁定。有关更多信息，请参阅识别阻止查询的内容。

  当您看到阻止会话时，经慎重考虑后，可以使用以下命令将其终止。这有助于继续执行 pg_repack 以完成重建：

```
SELECT pg_terminate_backend(pid);
```

- 在事务速率非常高的系统上应用 pg_repack's 日志表中的应计更改时，应用过程可能无法跟上更改速率。在这种情况下，pg_repack 将无法完成应用过程。有关更多信息，请参阅 在重新打包期间监控新表。如果索引严重膨胀，另一种解决方案是执行仅限索引的重新打包。这还有助于 VACUUM 的索引清理周期更快地完成。

  您可以使用 PostgreSQL 版本 12 中的手动 VACUUM 跳过索引清理阶段，在 PostgreSQL 版本 14 中的紧急 autovacuum 期间会自动跳过索引清理阶段。这有助于在不消除索引膨胀的情况下更快地完成 VACUUM，并且仅适用于紧急情况，例如防止重叠 VACUUM。有关更多信息，请参阅《Amazon Aurora 用户指南》中的避免索引膨胀。

先决条件

- 该表必须具有 PRIMARY KEY 或非 null UNIQUE 约束。
- 客户端和服务器的扩展版本必须相同。
- 确保 RDS 实例的 FreeStorageSpace 超过表的总大小，而不会出现膨胀。例如，假设表（包括 TOAST 和索引）的总大小为 2TB，表中的总膨胀为 1TB。所需 FreeStorageSpace 必须大于以下计算返回的值：

  2TB (Table size) - 1TB (Table bloat) = 1TB

  您可以使用以下查询来检查表的总大小，并使用 pgstattuple 来派生膨胀。有关更多信息，请参阅《Amazon Aurora 用户指南》中的诊断表和索引膨胀

  ```
  SELECT pg_size_pretty(pg_total_relation_size('table_name')) AS total_table_size;
  ```

  活动完成后，将回收此空间。
- 确保 RDS 实例有足够的计算和 IO 容量来处理重新打包操作。您可以考虑纵向扩展实例类以实现性能的最佳平衡。

使用 **pg_repack** 扩展

1. 通过运行以下命令在 RDS for PostgreSQL 数据库实例上安装 pg_repack 扩展。

   ```
   CREATE EXTENSION pg_repack;
   ```

2. 运行以下命令以授予对由 pg_repack 创建的临时日志表的写入访问权限。

```
ALTER DEFAULT PRIVILEGES IN SCHEMA repack GRANT INSERT ON TABLES TO PUBLIC;
ALTER DEFAULT PRIVILEGES IN SCHEMA repack GRANT USAGE, SELECT ON SEQUENCES TO
 PUBLIC;
```

3. 使用 pg_repack 客户端实用程序连接到数据库。使用具有 rds_superuser 特权的账户。例如，假设 rds_test 角色有 rds_superuser 特权。以下语法对完整表执行 pg_repack，包括 postgres 数据库中的所有表索引。

```
pg_repack -h db-instance-name.111122223333.aws-region.rds.amazonaws.com -U rds_test
 -k postgres
```

> **Note**
>
> 必须使用 -k 选项进行连接。不支持 -a 选项。

来自 pg_repack 客户端的响应提供有关重新打包的数据库实例上的表的信息。

```
INFO: repacking table "pgbench_tellers"
INFO: repacking table "pgbench_accounts"
INFO: repacking table "pgbench_branches"
```

4. 以下语法对 postgres 数据库中包含索引的单个表 orders 进行重新打包。

```
pg_repack -h db-instance-name.111122223333.aws-region.rds.amazonaws.com -U rds_test
 --table orders -k postgres
```

以下语法仅重新打包 postgres 数据库中 orders 表的索引。

```
pg_repack -h db-instance-name.111122223333.aws-region.rds.amazonaws.com -U rds_test
 --table orders --only-indexes -k postgres
```

在重新打包期间监控新表

- 在重新打包的交换和删除阶段之前，数据库的大小增加量为表的总大小减去膨胀。您可以监控数据库大小的增长率，计算重新打包的速度，并粗略估计完成初始数据传输所需的时间。

例如，假设表的总大小为 2TB，数据库的大小为 4TB，表中的总膨胀为 1TB。重新打包操作结束时计算返回的数据库总大小值如下：

```
2TB (Table size) + 4 TB (Database size) - 1TB (Table bloat) = 5TB
```

您可以通过对两个时间点之间的增长率（以字节为单位）进行采样来粗略估计重新打包操作的速度。如果增长率为每分钟 1GB，则可能需要 1000 分钟或大约 16.6 小时才能完成初始表构建操作。除了初始表构建外，pg_repack 还需要应用应计更改。所需时间取决于应用持续更改以及应计更改的速率。

> **Note**
>
> 您可以使用 pgstattuple 扩展来计算表中的膨胀。有关更多信息，请参阅 pgstattuple。

- 重新打包架构下 pg_repack's 日志表中的行数表示在初始加载后待应用于新表的更改量。

  您可以检查 pg_stat_all_tables 中的 pg_repack's 日志表以监控应用于新表的更改。pg_stat_all_tables.n_live_tup 表示待应用于新表的记录数。有关更多信息，请参阅 pg_stat_all_tables。

```
postgres=>SELECT relname,n_live_tup FROM pg_stat_all_tables WHERE schemaname =
 'repack' AND relname ILIKE '%log%';

-[ RECORD 1 ]---------
relname     | log_16490
n_live_tup  | 2000000
```

- 您可以使用 pg_stat_statements 扩展来找出重新打包操作中每个步骤所花费的时间。这有助于准备在生产环境中应用相同的重新打包操作。您可以调整 LIMIT 子句以进一步扩展输出。

```
postgres=>SELECT
     SUBSTR(query, 1, 100) query,
     round((round(total_exec_time::numeric, 6) / 1000 / 60),4)
 total_exec_time_in_minutes
 FROM
     pg_stat_statements
 WHERE
     query ILIKE '%repack%'
 ORDER BY
```

```
    total_exec_time DESC LIMIT 5;

 query                                                                 |
 total_exec_time_in_minutes
-----------------------------------------------------------------------
+-----------------------------
 CREATE UNIQUE INDEX index_16493 ON repack.table_16490 USING btree (a) |
       6.8627
 INSERT INTO repack.table_16490 SELECT a FROM ONLY public.t1           |
       6.4150
 SELECT repack.repack_apply($1, $2, $3, $4, $5, $6)                    |
       0.5395
 SELECT repack.repack_drop($1, $2)                                     |
       0.0004
 SELECT repack.repack_swap($1)                                         |
       0.0004
(5 rows)
```

重新打包完全是一项不合时宜的操作，因此原始表不会受到影响，我们预计不会出现任何需要恢复原始表的意外挑战。如果重新打包意外失败，则必须检查错误的原因并加以解决。

问题解决后，在表所在的数据库中删除并重新创建 pg_repack 扩展，然后重试 pg_repack 步骤。此外，计算资源的可用性和表的并行可访问性在及时完成重新打包操作方面起着至关重要的作用。

升级和使用 PLV8 扩展

PLV8 是适用于 PostgreSQL 的可信 Javascript 语言扩展。您可以将其用于存储过程、触发条件和其他可从 SQL 调用的过程代码。PostgreSQL 的所有当前版本都支持此语言扩展。

如果使用 PLV8 并将 PostgreSQL 升级到新的 PLV8 版本，您可以立即利用新扩展。按下列步骤将目录元数据与新版本的 PLV8 进行同步。这些步骤是可选的，但我们强烈建议您完成这些步骤以避免元数据错配警告。

升级过程会丢弃所有现有的 PLV8 功能。因此，建议您在升级之前创建 RDS for PostgreSQL 数据库实例的快照。有关更多信息，请参阅 为 Amazon RDS 的单可用区数据库实例创建数据库快照。

> **Important**
>
> 从 PostgreSQL 版本 18 开始，Amazon RDS for PostgreSQL 将弃用 `plcoffee` 和 `plls` PostgreSQL 扩展。建议您停止在应用程序中使用 CoffeeScript 和 LiveScript，以确保为未来的引擎版本升级提供升级途径。

将目录元数据与新版本的 PLV8 同步

1. 确认您需要更新。为此，请在连接到实例的情况下运行以下命令。

   ```
   SELECT * FROM pg_available_extensions WHERE name IN ('plv8','plls','plcoffee');
   ```

   如果您的结果包含的已安装版本的值低于默认版本，请继续该过程以更新扩展。例如，以下结果集指示您应更新。

   ```
   name    | default_version | installed_version |                     comment
   --------+-----------------+-------------------
   +--------------------------------------------------
   plls    | 2.1.0           | 1.5.3             | PL/LiveScript (v8) trusted
    procedural language
   plcoffee| 2.1.0           | 1.5.3             | PL/CoffeeScript (v8) trusted
    procedural language
   plv8    | 2.1.0           | 1.5.3             | PL/JavaScript (v8) trusted
    procedural language
   (3 rows)
   ```

2. 如果尚未创建 RDS for PostgreSQL 数据库实例的快照，请先创建一个。创建快照时，您可以继续下列步骤。
3. 获取数据库实例中 PLV8 函数的数量，以便在升级后验证是否遗漏函数。例如，以下 SQL 查询会返回用 plv8、plcoffee 和 plls 编写的函数数量。

```
SELECT proname, nspname, lanname
FROM pg_proc p, pg_language l, pg_namespace n
WHERE p.prolang = l.oid
AND n.oid = p.pronamespace
AND lanname IN ('plv8','plcoffee','plls');
```

4. 使用 pg_dump 创建仅架构转储文件。例如，在客户端计算机上的 `/tmp` 目录中创建文件。

```
./pg_dump -Fc --schema-only -U master postgres >/tmp/test.dmp
```

   该示例使用以下选项：

   - `-Fc` - 自定义格式
   - --仅架构 - 仅转储创建架构所需的命令（在本例中为功能）
   - `-U` - RDS 主用户名
   - `database` - 数据库实例的数据库名称

   有关 pg_dump 的更多信息，请参阅 PostgreSQL 文档中的 pg_dump。
5. 提取位于转储文件中的“CREATE FUNCTION”DDL 语句。以下示例使用 `grep` 命令提取创建函数的 DDL 语句并将函数保存到文件中。将在后续步骤中使用该语句重新创建函数。

```
./pg_restore -l /tmp/test.dmp | grep FUNCTION > /tmp/function_list
```

   有关 pg_restore 的更多信息，请参阅 PostgreSQL 文档中的 pg_restore。
6. 删掉函数和扩展。以下示例将删除基于 PLV8 的任何对象。级联选项确保删除任何依赖项。

```
DROP EXTENSION plv8 CASCADE;
```

   如果您的 PostgreSQL 实例包含基于 plcoffee 或 plls 的对象，请对这些扩展重复此步骤。
7. 创建扩展。以下示例将创建 plv8、plcoffee 和 plls 扩展。

```
CREATE EXTENSION plv8;
CREATE EXTENSION plcoffee;
CREATE EXTENSION plls;
```

8. 使用转储文件和“驱动程序”文件创建函数。

   以下示例将重新创建之前提取的函数。

```
./pg_restore -U master -d postgres -Fc -L /tmp/function_list /tmp/test.dmp
```

9. 使用以下查询验证是否已重新创建所有函数。

```
SELECT * FROM pg_available_extensions WHERE name IN ('plv8','plls','plcoffee');
```

   PLV8 版本 2 在结果集中添加以下额外行：

```
    proname     |  nspname   | lanname
---------------+------------+----------
 plv8_version  | pg_catalog | plv8
```

使用 PL/Rust 通过 Rust 语言编写 PostgreSQL 函数

PL/Rust 是 PostgreSQL 的可信 Rust 语言扩展。您可以将其用于存储过程、函数和其他可从 SQL 调用的过程代码。PL/Rust 语言扩展有以下版本可用：

- RDS for PostgreSQL 17.1 及更高的 17 版本
- RDS for PostgreSQL 16.1 及更高的 16 版本
- RDS for PostgreSQL 15.2-R2 和更高的 15 版本
- RDS for PostgreSQL 14.9 和更高的 14 版本
- RDS for PostgreSQL 13.12 和更高的 13 版本

有关更多信息，请参阅 GitHub 上的 PL/Rust。

主题

- 将 crate 与 PL/Rust 结合使用
- PL/Rust 限制

设置 PL/Rust

要在数据库实例上安装 plrust 扩展，请将 plrust 添加到与数据库实例相关联的数据库参数组中的 `shared_preload_libraries` 参数。安装了 plrust 扩展后，您可以创建函数。

要修改 `shared_preload_libraries` 参数，数据库实例必须与自定义参数组相关联。有关创建自定义数据库参数组的信息，请参阅Amazon RDS 的参数组。

您可以使用 Amazon Web Services Management Console或 Amazon CLI 安装 plrust 扩展程序。

以下步骤假设您的数据库实例与自定义数据库参数组相关联。

控制台

在 **`shared_preload_libraries`** 参数中安装 plrust 扩展

使用作为 `rds_superuser` 组（角色）成员的账户完成以下步骤。

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases（数据库）。
3. 选择数据库实例的名称以显示其详细信息。
4. 打开数据库实例的配置选项卡，并找到数据库实例参数组链接。
5. 选择此链接以打开与您的数据库实例关联的自定义参数。
6. 在 Parameters（参数）搜索字段中，键入 `shared_pre` 以查找 **`shared_preload_libraries`** 参数。
7. 选择 Edit parameters（编辑参数）以访问属性值。
8. 将 plrust 添加到值字段的列表中。使用逗号分隔值列表中的项目。
9. 重启数据库实例以使对 `shared_preload_libraries` 参数的更改生效。初次重启可能需要额外的时间才能完成。
10. 当实例可用时，验证 plrust 是否已初始化。使用 `psql` 连接到数据库实例，然后并运行以下命令。

```
SHOW shared_preload_libraries;
```

您的输出应类似于以下内容：

```
shared_preload_libraries
--------------------------
rdsutils,plrust
(1 row)
```

Amazon CLI

在 shared_preload_libraries 参数中安装 plrust 扩展

使用作为 `rds_superuser` 组（角色）成员的账户完成以下步骤。

1. 使用 [modify-db-parameter-group](https://docs.aws.amazon.com/cli/latest/reference/rds/modify-db-parameter-group.html) Amazon CLI 命令将 plrust 添加到 `shared_preload_libraries` 参数。

```
aws rds modify-db-parameter-group \
   --db-parameter-group-name custom-param-group-name \
   --parameters
 "ParameterName=shared_preload_libraries,ParameterValue=plrust,ApplyMethod=pending-
reboot" \
   --region aws-region
```

2. 使用 [reboot-db-instance](https://docs.aws.amazon.com/cli/latest/reference/rds/reboot-db-instance.html) Amazon CLI 命令重启数据库实例并初始化 plrust 库。初次重启可能需要额外的时间才能完成。

```
aws rds reboot-db-instance \
    --db-instance-identifier your-instance \
    --region aws-region
```

3. 当实例可用时，您可以验证 plrust 已初始化。使用 `psql` 连接到数据库实例，然后并运行以下命令。

```
SHOW shared_preload_libraries;
```

您的输出应类似于以下内容：

```
shared_preload_libraries
--------------------------
```

```
rdsutils,plrust
(1 row)
```

使用 PL/Rust 创建函数

PL/Rust 会将函数编译为动态库、加载动态库并执行该库。

以下 Rust 函数从数组中筛选出倍数。

```
postgres=> CREATE LANGUAGE plrust;
CREATE EXTENSION
```

```
CREATE OR REPLACE FUNCTION filter_multiples(a BIGINT[], multiple BIGINT) RETURNS
 BIGINT[]
    IMMUTABLE STRICT
    LANGUAGE PLRUST AS
$$
    Ok(Some(a.into_iter().filter(|x| x.unwrap() % multiple != 0).collect()))
$$;

WITH gen_values AS (
SELECT ARRAY(SELECT * FROM generate_series(1,100)) as arr)
SELECT filter_multiples(arr, 3)
from gen_values;
```

将 crate 与 PL/Rust 结合使用

在 RDS for PostgreSQL 版本 16.3-R2 及更高版本、15.7-R2 及更高的 15 版本、14.12-R2 及更高的 14 版本、13.15-R2 及更高的 13 版本中，PL/Rust 支持额外的 crate：

- `url`
- `regex`
- `serde`
- `serde_json`

在 RDS for PostgreSQL 版本 15.5-R2 及更高版本、14.10-R2 及更高的 14 版本、13.13-R2 及更高的 13 版本中，PL/Rust 支持另外两个 crate：

- `croaring-rs`

- `num-bigint`

从 Amazon RDS for PostgreSQL 版本 15.4、14.9 和 13.12 开始，PL/Rust 支持以下 crate：

- `aes`
- `ctr`
- `rand`

这些 crate 仅支持默认特征。新的 RDS for PostgreSQL 版本可能包含更新版本的 crate，并且可能不再支持旧版本的 crate。

按照执行主要版本升级的最佳实践来测试您的 PL/Rust 函数是否与新的主要版本兼容。有关更多信息，请参阅博客将 Amazon RDS 升级到 PostgreSQL 主要和次要版本的最佳实践，以及《Amazon RDS 用户指南》中的为 Amazon RDS 升级 PostgreSQL 数据库引擎。

有关在创建 PL/Rust 函数时使用依赖关系的示例，请访问使用依赖关系。

PL/Rust 限制

默认情况下，数据库用户无法使用 PL/Rust。要提供对 PL/Rust 的访问权限，请以具有 rds_superuser 权限的用户身份进行连接，然后运行以下命令：

```
postgres=> GRANT USAGE ON LANGUAGE PLRUST TO user;
```

使用 PostGIS 扩展管理空间数据

PostGIS 是一个 PostgreSQL 扩展，用于存储和管理空间信息。要了解有关 PostGIS 的更多信息，请参阅 PostGIS.net。

从版本 10.5 开始，PostgreSQL 支持 PostGIS 用于处理地图框矢量平铺数据的 libprotobuf 1.3.0 库。

设置 PostGIS 扩展需要 `rds_superuser` 权限。我们建议您创建一个用户（角色），以管理 PostGIS 和您的空间数据。PostGIS 扩展及其相关组件为 PostgreSQL 添加了数千个函数。如果这对您的使用案例有意义，请考虑在自己的架构中创建 PostGIS 扩展。以下示例说明了如何在其各自的数据库中安装扩展，但这并不是必需的。

主题

步骤 1：创建用户（角色）来管理 PostGIS 扩展

首先，以具有 `rds_superuser` 权限的用户身份连接到 RDS for PostgreSQL 数据库实例。如果您在设置实例时保留原定设置名称，则以 `postgres` 进行连接。

```
psql --host=111122223333.aws-region.rds.amazonaws.com --port=5432 --username=postgres
 --password
```

创建单独的角色（用户）来管理 PostGIS 扩展。

```
postgres=>  CREATE ROLE gis_admin LOGIN PASSWORD 'change_me';
CREATE ROLE
```

向此角色授予 `rds_superuser` 权限，以允许角色安装扩展。

```
postgres=> GRANT rds_superuser TO gis_admin;
GRANT
```

创建一个要用于 PostGIS 构件的数据库。此为可选步骤。或者，您可以在用户数据库中为 PostGIS 扩展创建架构，但这也不是必需的。

```
postgres=> CREATE DATABASE lab_gis;
CREATE DATABASE
```

为 gis_admin 提供 lab_gis 数据库的所有权限。

```
postgres=> GRANT ALL PRIVILEGES ON DATABASE lab_gis TO gis_admin;
GRANT
```

退出会话并以 gis_admin 身份重新连接 RDS for PostgreSQL 数据库实例。

```
postgres=> psql --host=111122223333.aws-region.rds.amazonaws.com --port=5432 --
username=gis_admin --password --dbname=lab_gis
Password for user gis_admin:...
lab_gis=>
```

按照后续步骤中的详细说明，继续设置扩展。

步骤 2：加载 PostGIS 扩展

PostGIS 扩展模块包括多个相关的扩展，它们协同工作以提供地理空间功能。您可能不需要在此步骤中创建的所有扩展，具体取决于您的使用案例。

使用 CREATE EXTENSION 语句加载 PostGIS 扩展。

```
CREATE EXTENSION postgis;
CREATE EXTENSION
CREATE EXTENSION postgis_raster;
CREATE EXTENSION
CREATE EXTENSION fuzzystrmatch;
CREATE EXTENSION
CREATE EXTENSION postgis_tiger_geocoder;
CREATE EXTENSION
CREATE EXTENSION postgis_topology;
```

```
CREATE EXTENSION
CREATE EXTENSION address_standardizer_data_us;
CREATE EXTENSION
```

您可以通过运行以下示例中显示的 SQL 查询来验证结果，该查询列出了扩展及其所有者。

```
SELECT n.nspname AS "Name",
  pg_catalog.pg_get_userbyid(n.nspowner) AS "Owner"
  FROM pg_catalog.pg_namespace n
  WHERE n.nspname !~ '^pg_' AND n.nspname <> 'information_schema'
  ORDER BY 1;
List of schemas
     Name     |   Owner
--------------+-----------
 public       | postgres
 tiger        | rdsadmin
 tiger_data   | rdsadmin
 topology     | rdsadmin
(4 rows)
```

步骤 3：移交扩展架构的所有权

使用 ALTER SCHEMA 语句将架构的所有权移交给 gis_admin 角色。

```
ALTER SCHEMA tiger OWNER TO gis_admin;
ALTER SCHEMA
ALTER SCHEMA tiger_data OWNER TO gis_admin;
ALTER SCHEMA
ALTER SCHEMA topology OWNER TO gis_admin;
ALTER SCHEMA
```

您可以通过运行以下 SQL 查询来确认所有权变更。或者，您也可以从 psql 命令行使用 \dn 元命令。

```
SELECT n.nspname AS "Name",
  pg_catalog.pg_get_userbyid(n.nspowner) AS "Owner"
  FROM pg_catalog.pg_namespace n
  WHERE n.nspname !~ '^pg_' AND n.nspname <> 'information_schema'
  ORDER BY 1;

       List of schemas
     Name     |     Owner
```

```
--------------+---------------
 public       | postgres
 tiger        | gis_admin
 tiger_data   | gis_admin
 topology     | gis_admin
(4 rows)
```

步骤 4：移交 PostGIS 表的所有权

> **Note**
>
> 请勿更改 PostGIS 函数的所有权。PostGIS 的正常运行和未来升级要求这些函数保留原始所有权。有关 PostGIS 权限的更多信息，请参阅 PostgreSQL Security。

使用以下函数将 PostGIS 表的所有权移交给 `gis_admin` 角色。从 psql 提示符处运行以下语句以创建此函数。

```
CREATE FUNCTION exec(text) returns text language plpgsql volatile AS $f$ BEGIN EXECUTE
 $1; RETURN $1; END; $f$;
CREATE FUNCTION
```

接下来，运行以下查询以运行 `exec` 函数，该函数进而将运行语句并更改权限。

```
SELECT exec('ALTER TABLE ' || quote_ident(s.nspname) || '.' || quote_ident(s.relname)
 || ' OWNER TO gis_admin;')
  FROM (
    SELECT nspname, relname
    FROM pg_class c JOIN pg_namespace n ON (c.relnamespace = n.oid)
    WHERE nspname in ('tiger','topology') AND
    relkind IN ('r','S','v') ORDER BY relkind = 'S')
s;
```

步骤 5：测试扩展

为避免需要指定架构名称，请使用以下命令将 `tiger` 架构添加到搜索路径中。

```
SET search_path=public,tiger;
SET
```

使用以下 SELECT 语句测试 tiger 架构。

```
SELECT address, streetname, streettypeabbrev, zip
 FROM normalize_address('1 Devonshire Place, Boston, MA 02109') AS na;
address | streetname | streettypeabbrev |  zip
---------+------------+------------------+-------
       1 | Devonshire | Pl               | 02109
(1 row)
```

要了解有关此扩展的更多信息，请参阅 PostGIS 文档中的 Tiger 地理编码器。

使用以下 SELECT 语句测试对 topology 架构的访问。这将调用 createtopology 函数，以使用指定的空间参考标识符（26986）和原定设置容差（0.5）注册新的拓扑对象（my_new_topo）。要了解更多信息，请参阅 PostGIS 文档中的 CreateTopology。

```
SELECT topology.createtopology('my_new_topo',26986,0.5);
 createtopology
----------------
              1
(1 row)
```

步骤 6：升级 PostGIS 扩展

PostgreSQL 的每个新发行版都支持与该发行版兼容的一个或多个 PostGIS 扩展版本。将 PostgreSQL 引擎升级到新版本不会自动升级 PostGIS 扩展。在升级 PostgreSQL 引擎之前，通常需要将 PostGIS 升级到当前 PostgreSQL 版本的最新可用版本。有关详细信息，请参阅PostGIS 扩展版本。

PostgreSQL 引擎升级后，您可以再次将 PostGIS 扩展升级到新升级的 PostgreSQL 引擎版本所支持的版本。有关升级 PostgreSQL 引擎的更多信息，请参阅如何执行 RDS for PostgreSQL 的主要版本升级。

您可以随时检查 RDS for PostgreSQL 数据库实例上可用的 PostGIS 扩展版本更新。为此，请运行以下命令。PostGIS 2.5.0 及更高版本可以使用此功能。

```
SELECT postGIS_extensions_upgrade();
```

如果您的应用程序不支持最新的 PostGIS 版本，您可以安装主要版本中提供的旧版本 PostGIS，如下所示。

```
CREATE EXTENSION postgis VERSION "2.5.5";
```

如果要从旧版本升级到特定 PostGIS 版本，还可以使用以下命令。

```
ALTER EXTENSION postgis UPDATE TO "2.5.5";
```

根据要从中进行升级的版本，您可能需要再次使用此函数。第一次运行该函数的结果决定是否需要额外的升级功能。例如，从 PostGIS 2 升级到 PostGIS 3 就是这种情况。有关更多信息，请参阅 将 PostGIS 2 升级到 PostGIS 3。

如果您升级此扩展是为了准备进行 PostgreSQL 引擎的主要版本升级，则可以继续执行其他初步任务。有关更多信息，请参阅如何执行 RDS for PostgreSQL 的主要版本升级。

PostGIS 扩展版本

我们建议您安装 《Amazon RDS for PostgreSQL 版本注释》的 Amazon RDS for PostgreSQL 的扩展版本中列出的所有扩展版本，如 PostGIS。要获取发行版中可用的版本列表，请使用以下命令。

```
SELECT * FROM pg_available_extension_versions WHERE name='postgis';
```

您可以在《Amazon RDS for PostgreSQL 版本注释》的以下部分中找到版本信息：

- Amazon RDS 上支持的 PostgreSQL 16 版本扩展
- Amazon RDS 上支持的 PostgreSQL 版本 15 扩展
- Amazon RDS 上支持的 PostgreSQL 版本 14 扩展
- Amazon RDS 上支持的 PostgreSQL 版本 13 扩展
- Amazon RDS 上支持的 PostgreSQL 版本 12 扩展
- Amazon RDS 上支持的 PostgreSQL 版本 11 扩展
- Amazon RDS 上支持的 PostgreSQL 版本 10 扩展
- Amazon RDS 上支持的 PostgreSQL 版本 9.6.x 扩展

将 PostGIS 2 升级到 PostGIS 3

从版本 3.0 开始，PostGIS 光栅功能现在是一个单独的扩展，即 `postgis_raster`。此扩展有自己的安装和升级路径。这将从核心 `postgis` 扩展中删除光栅图像处理所需的数十个函数、数据类型和其他构件。这意味着，如果您的使用案例不需要光栅处理，则不需要安装 `postgis_raster` 扩展。

在以下升级示例中，第一个升级命令将光栅功能提取到 `postgis_raster` 扩展。然后，需要使用第二个升级命令将 `postgis_raster` 升级到新版本。

从 PostGIS 2 升级到 PostGIS 3

1. 确定 RDS for PostgreSQL 数据库实例上 PostgreSQL 版本可用的 PostGIS 的默认版本。为此，请运行以下查询。

```
SELECT * FROM pg_available_extensions
    WHERE default_version > installed_version;
  name   | default_version | installed_version |                          comment
---------+-----------------+-------------------
+-------------------------------------------------------------
 postgis | 3.1.4           | 2.3.7             | PostGIS geometry and geography
 spatial types and functions
(1 row)
```

2. 确定 RDS for PostgreSQL 数据库实例上每个数据库中安装的 PostGIS 版本。换句话说，按如下方式查询每个用户数据库。

```
SELECT
    e.extname AS "Name",
    e.extversion AS "Version",
    n.nspname AS "Schema",
    c.description AS "Description"
FROM
    pg_catalog.pg_extension e
    LEFT JOIN pg_catalog.pg_namespace n ON n.oid = e.extnamespace
    LEFT JOIN pg_catalog.pg_description c ON c.objoid = e.oid
    AND c.classoid = 'pg_catalog.pg_extension'::pg_catalog.regclass
WHERE
    e.extname LIKE '%postgis%'
ORDER BY
    1;
  Name   | Version | Schema |                             Description
---------+---------+--------
+---------------------------------------------------------------------
 postgis | 2.3.7   | public | PostGIS geometry, geography, and raster spatial types
 and functions
(1 row)
```

原定设置版本（PostGIS 3.1.4）和已安装版本（PostGIS 2.3.7）之间的不匹配意味着您需要升级 PostGIS 扩展。

```
ALTER EXTENSION postgis UPDATE;
```

```
ALTER EXTENSION
WARNING: unpackaging raster
WARNING: PostGIS Raster functionality has been unpackaged
```

3. 运行以下查询，以验证光栅功能现在是否已包含在自己的程序包中。

```
SELECT
    probin,
    count(*)
FROM
    pg_proc
WHERE
    probin LIKE '%postgis%'
GROUP BY
    probin;
          probin          | count
--------------------------+-------
 $libdir/rtpostgis-2.3    | 107
 $libdir/postgis-3        | 487
(2 rows)
```

输出将显示版本之间仍然存在差异。PostGIS 函数为版本 3（postgis-3），而光栅函数（rtpostgis）为第 2 版（rtpostgis-2.3）。要完成升级，请再次运行升级命令，如下所示。

```
postgres=> SELECT postgis_extensions_upgrade();
```

您可以放心地忽略警告消息。再次运行以下查询，以验证升级已完成。当 PostGIS 和所有相关扩展未标记为需要升级时，升级即告完成。

```
SELECT postgis_full_version();
```

4. 使用以下查询查看已完成的升级过程和单独打包的扩展，并验证其版本是否匹配。

```
SELECT
    e.extname AS "Name",
    e.extversion AS "Version",
    n.nspname AS "Schema",
    c.description AS "Description"
FROM
    pg_catalog.pg_extension e
    LEFT JOIN pg_catalog.pg_namespace n ON n.oid = e.extnamespace
    LEFT JOIN pg_catalog.pg_description c ON c.objoid = e.oid
```

```
        AND c.classoid = 'pg_catalog.pg_extension'::pg_catalog.regclass
WHERE
    e.extname LIKE '%postgis%'
ORDER BY
    1;
      Name      | Version | Schema |                             Description
----------------+---------+--------
+-------------------------------------------------------------------
 postgis        | 3.1.5   | public | PostGIS geometry, geography, and raster
 spatial types and functions
 postgis_raster | 3.1.5   | public | PostGIS raster types and functions
(2 rows)
```

输出显示，PostGIS 2 扩展已升级到 PostGIS 3，并且 `postgis` 和现在独立的 `postgis_raster` 扩展都为版本 3.1.5。

升级完成后，如果您不打算使用光栅功能，则可以按如下方式删除此扩展。

```
DROP EXTENSION postgis_raster;
```

使用 Amazon RDS for PostgreSQL 支持的外部数据包装器

外部数据包装器 (FDW) 是一种特定类型的扩展，提供对外部数据的访问。例如，`oracle_fdw` 扩展允许您的 RDS for PostgreSQL 数据库集群使用 Oracle 数据库。再例如，通过使用 PostgreSQL 本机 `postgres_fdw` 扩展，您可以访问存储在 RDS for PostgreSQL 数据库实例外部的 PostgreSQL 数据库实例中的数据。

在下文中，您可以了解有关几种支持的 PostgreSQL 外部数据包装器的信息。

主题

使用 log_fdw 扩展通过 SQL 访问数据库日志

RDS for PostgreSQL 数据库实例支持 `log_fdw` 扩展，您可以使用该扩展通过 SQL 界面访问数据库引擎日志。此 `log_fdw` 扩展提供了两个新函数，便于创建数据库日志的外部表：

- `list_postgres_log_files` – 列出数据库日志目录中的文件，以及文件大小 (以字节为单位)。
- `create_foreign_table_for_log_file(table_name text, server_name text, log_file_name text)` – 针对当前数据库中的指定文件构建外部表。

`log_fdw` 创建的所有函数均归 `rds_superuser` 所有。`rds_superuser` 角色的成员可以将这些函数的访问权限授予其他数据库用户。

默认情况下，日志文件由 Amazon RDS 以 `stderr`（标准错误）格式生成，如 `log_destination` 参数中指定。此参数只有两个选项，即，`stderr` 和 `csvlog`（逗号分隔值，CSV）。如果您为参数添加 `csvlog` 选项，Amazon RDS 会同时生成 `stderr` 和 `csvlog` 日志。这可能会影响数据库集群的存储容量，因此您需要了解影响日志处理的其它参数。有关更多信息，请参阅 设置日志目标（`stderr`、`csvlog`）。

生成 `csvlog` 日志的一个优势是 `log_fdw` 扩展允许您构建将数据整齐地拆分为多个列的外部表。为此，您的实例需要与自定义数据库参数组关联，以便您可以更改 `log_destination` 的设置。有关如何执行此操作的更多信息，请参阅在 RDS for PostgreSQL 数据库实例上使用参数。

以下示例假设 log_destination 参数包含 cvslog。

使用 log_fdw 扩展

1. 安装 log_fdw 扩展。

```
postgres=> CREATE EXTENSION log_fdw;
CREATE EXTENSION
```

2. 创建日志服务器，作为外部数据包装器。

```
postgres=> CREATE SERVER log_server FOREIGN DATA WRAPPER log_fdw;
CREATE SERVER
```

3. 选择日志文件列表中的所有文件。

```
postgres=> SELECT * FROM list_postgres_log_files() ORDER BY 1;
```

示例响应如下所示。

```
          file_name           | file_size_bytes
------------------------------+-----------------
 postgresql.log.2023-08-09-22.csv |            1111
 postgresql.log.2023-08-09-23.csv |            1172
 postgresql.log.2023-08-10-00.csv |            1744
 postgresql.log.2023-08-10-01.csv |            1102
(4 rows)
```

4. 为所选文件创建包含单个“log_entry”列的表。

```
postgres=> SELECT create_foreign_table_for_log_file('my_postgres_error_log',
     'log_server', 'postgresql.log.2023-08-09-22.csv');
```

除了告知现在存在表格外，响应不提供详细信息。

```
-----------------------------------
(1 row)
```

5. 选择日志文件的示例。以下代码检索日志时间和错误消息描述。

```
postgres=> SELECT log_time, message FROM my_postgres_error_log ORDER BY 1;
```

示例响应如下所示。

```
              log_time              |                                  message
----------------------------------
+------------------------------------------------------------------------------
Tue Aug 09 15:45:18.172 2023 PDT | ending log output to stderr
Tue Aug 09 15:45:18.175 2023 PDT | database system was interrupted; last known up
 at 2023-08-09 22:43:34 UTC
Tue Aug 09 15:45:18.223 2023 PDT | checkpoint record is at 0/90002E0
Tue Aug 09 15:45:18.223 2023 PDT | redo record is at 0/90002A8; shutdown FALSE
Tue Aug 09 15:45:18.223 2023 PDT | next transaction ID: 0/1879; next OID: 24578
Tue Aug 09 15:45:18.223 2023 PDT | next MultiXactId: 1; next MultiXactOffset: 0
Tue Aug 09 15:45:18.223 2023 PDT | oldest unfrozen transaction ID: 1822, in
 database 1
(7 rows)
```

使用 postgres_fdw 扩展访问外部数据

您可以使用 postgres_fdw 扩展访问远程数据库服务器上表中的数据。如果您从 PostgreSQL 数据库实例设置远程连接，则访问还可用于您的只读副本。

使用 postgres_fdw 访问远程数据库服务器

1. 安装 postgres_fdw 扩展。

```
CREATE EXTENSION postgres_fdw;
```

2. 使用 CREATE SERVER 创建外部数据服务器。

```
CREATE SERVER foreign_server
FOREIGN DATA WRAPPER postgres_fdw
OPTIONS (host 'xxx.xx.xxx.xx', port '5432', dbname 'foreign_db');
```

3. 创建用户映射，用于标识将在远程服务器上使用的角色。

> **Important**
>
> 要掩蔽密码以使其不出现在日志中，请在会话级别设置 `log_statement=none`。在参数级别进行设置不会掩蔽密码。

```
CREATE USER MAPPING FOR local_user
SERVER foreign_server
OPTIONS (user 'foreign_user', password 'password');
```

4. 创建一个表，该表映射到远程服务器上的表。

```
CREATE FOREIGN TABLE foreign_table (
        id integer NOT NULL,
        data text)
SERVER foreign_server
OPTIONS (schema_name 'some_schema', table_name 'some_table');
```

使用 mysql_fdw 扩展处理 MySQL 数据库

要从 RDS for PostgreSQL 数据库实例访问 MySQL 兼容数据库，您可以安装并使用 `mysql_fdw` 扩展。这款外部数据包装器允许您使用 RDS for MySQL、Aurora MySQL、MariaDB 和其他 MySQL 兼容数据库。从 RDS for PostgreSQL 数据库实例到 MySQL 数据库的连接会尽可能加密，具体取决于客户端和服务器配置。但是，如果您愿意，可以强制加密。有关更多信息，请参阅 将传输中加密与扩展配合使用。

`mysql_fdw` 扩展在 Amazon RDS for PostgreSQL 版本 14.2、13.6 以及更高版本中受支持。它支持从 RDS for PostgreSQL 数据库到 MySQL 兼容数据库实例上的表的选择、插入、更新和删除。

主题

将 RDS for PostgreSQL 数据库设置为使用 mysql_fdw 扩展

在您的 RDS for PostgreSQL 数据库实例上设置 `mysql_fdw` 扩展涉及在您的数据库实例中加载扩展，然后创建到 MySQL 数据库实例的连接点。对于该任务，您需要了解有关 MySQL 数据库实例的以下详细信息：

- 主机名或终端节点。对于RDS for MySQL 数据库实例，您可以使用控制台查找终端节点。选择 Connectivity & security（连接和安全）选项卡，然后查看 Endpoint and port（终端节点和端口）部分。
- 端口号。MySQL 的默认端口是 3306。
- 数据库的名称。数据库标识符。

您还需要为 MySQL 端口 3306 提供对安全组或访问控制列表 (ACL) 的访问权限。RDS for PostgreSQL 数据库实例和 RDS for MySQL 数据库实例均需要访问端口 3306。如果访问权限配置不正确，当尝试连接到 MySQL 兼容表时，您会看到一条与以下内容类似的错误消息：

```
ERROR: failed to connect to MySQL: Can't connect to MySQL server on 'hostname.aws-region.rds.amazonaws.com:3306' (110)
```

在以下过程中，您（作为 `rds_superuser` 账户）创建外部服务器。然后，您将访问外部服务器的权限授予特定用户。然后，这些用户创建其自身到相应 MySQL 用户账户的映射以使用 MySQL 数据库实例。

使用 mysql_fdw 访问 MySQL 数据库服务器

1. 使用具有 `rds_superuser` 角色的账户连接到您的 PostgreSQL 数据库实例。如果在创建 RDS for PostgreSQL 数据库实例时接受默认认值，则用户名为 `postgres`，您可以使用 `psql` 命令行工具进行连接，如下所示：

   ```
   psql --host=your-DB-instance.aws-region.rds.amazonaws.com --port=5432 --username=postgres --password
   ```

2. 按如下方式安装 `mysql_fdw` 扩展：

   ```
   postgres=> CREATE EXTENSION mysql_fdw;
   CREATE EXTENSION
   ```

在 RDS for PostgreSQL 数据库实例上安装扩展后，您可以设置提供与 MySQL 数据库连接的外部服务器。

创建外部服务器

在 RDS for PostgreSQL 数据库实例上执行这些任务。这些步骤假定您以具有 `rds_superuser` 特权的用户身份连接，例如 `postgres`。

1. 在 RDS for PostgreSQL 数据库实例中创建外部服务器：

```
postgres=> CREATE SERVER mysql-db FOREIGN DATA WRAPPER mysql_fdw OPTIONS (host 'db-name.111122223333.aws-region.rds.amazonaws.com', port '3306');
CREATE SERVER
```

2. 向适当的用户授予访问外部服务器的权限。这些用户应该是非管理员用户，即，没有 `rds_superuser` 角色的用户。

```
postgres=> GRANT USAGE ON FOREIGN SERVER mysql-db to user1;
GRANT
```

PostgreSQL 用户通过外部服务器创建和管理其自身与 MySQL 数据库的连接。

示例：从 RDS for PostgreSQL 使用 RDS for MySQL 数据库

假设您在 RDS for PostgreSQL 数据库实例上有一个简单的表。您的 RDS for PostgreSQL 用户想要查询该表中的（`SELECT`）、`INSERT`、`UPDATE` 和 `DELETE` 项目。假设 `mysql_fdw` 扩展是在您的 RDS for PostgreSQL 数据库实例上创建的，如前面的过程中所述。以具有 `rds_superuser` 权限的用户身份连接到 RDS for PostgreSQL 数据库实例后，您可以继续执行以下步骤。

1. 在 RDS for PostgreSQL 数据库实例上，创建一个外部服务器：

```
test=> CREATE SERVER mysqldb FOREIGN DATA WRAPPER mysql_fdw OPTIONS (host 'your-DB.aws-region.rds.amazonaws.com', port '3306');
CREATE SERVER
```

2. 将使用权授予没有 `rds_superuser` 权限的用户，例如 `user1`：

```
test=> GRANT USAGE ON FOREIGN SERVER mysqldb TO user1;
GRANT
```

3. 作为 *`user1`* 连接，然后创建一个到 MySQL 用户的映射：

```
test=> CREATE USER MAPPING FOR user1 SERVER mysqldb OPTIONS (username 'myuser', password 'mypassword');
CREATE USER MAPPING
```

4. 创建链接到 MySQL 表的外部表：

```
test=> CREATE FOREIGN TABLE mytab (a int, b text) SERVER mysqldb OPTIONS (dbname
 'test', table_name '');
CREATE FOREIGN TABLE
```

5. 针对外表运行简单查询：

```
test=> SELECT * FROM mytab;
a |   b
---+-------
1 | apple
(1 row)
```

6. 您可以从 MySQL 表中添加、更改和删除数据。例如：

```
test=> INSERT INTO mytab values (2, 'mango');
INSERT 0 1
```

再次运行 SELECT 查询以查看结果：

```
test=> SELECT * FROM mytab ORDER BY 1;
 a |   b
---+-------
1 | apple
2 | mango
(2 rows)
```

将传输中加密与扩展配合使用

默认情况下，从 RDS for PostgreSQL 到 MySQL 的连接使用传输中加密（TLS/SSL）。但是，当客户端和服务器配置不同时，连接会回退为非加密状态。您可以通过在 RDS for MySQL 用户账户上指定 REQUIRE SSL 选项来对所有传出连接强制加密。这种方法也适用于 MariaDB 和 Aurora MySQL 用户账户。

对于配置为 REQUIRE SSL 的 MySQL 用户帐户，如果无法建立安全连接，则连接尝试将失败。

要对现有 MySQL 数据库用户帐户强制加密，可以使用 ALTER USER 命令。根据 MySQL 版本的不同，语法有所不同，如下表所示。有关更多信息，请参阅《MySQL 参考手册》中的 ALTER USER。

MySQL 5.7、MySQL 8.0	MySQL 5.6
`ALTER USER 'user'@'%' REQUIRE SSL;`	`GRANT USAGE ON *.* to 'user'@'%' REQUIRE SSL;`

有关 `mysql_fdw` 扩展的更多信息，请参阅 mysql_fdw 文档。

通过使用 oracle_fdw 扩展来使用 Oracle 数据库

要从 RDS for PostgreSQL 数据库实例访问 Oracle 数据库，您可以安装并使用 `oracle_fdw` 扩展。此扩展是 Oracle 数据库的外部数据包装器。要了解有关此扩展的更多信息，请参阅 oracle_fdw 文档。

RDS for PostgreSQL 12.7、13.3 及更高版本支持 `oracle_fdw` 扩展。

主题

- 启用 oracle_fdw 扩展
- 示例：使用链接到 Amazon RDS for Oracle Database 的外部服务器
- 在传输过程中使用加密
- 了解 pg_user_mappings 视图和权限

启用 oracle_fdw 扩展

要使用 oracle_fdw 扩展，请执行以下步骤。

启用 oracle_fdw 扩展

- 使用具有 `rds_superuser` 权限的账户运行以下命令。

```
CREATE EXTENSION oracle_fdw;
```

示例：使用链接到 Amazon RDS for Oracle Database 的外部服务器

以下示例展示如何使用链接到 Amazon RDS for Oracle Database 的外部服务器。

创建链接到 RDS for Oracle 数据库的外部服务器

1. 请注意 RDS for Oracle 数据库实例上的以下内容：

- Endpoint
- 端口
- 数据库名称

2. 创建外部服务器。

```
test=> CREATE SERVER oradb FOREIGN DATA WRAPPER oracle_fdw OPTIONS (dbserver
 '//endpoint:port/DB_name');
CREATE SERVER
```

3. 将使用权授予没有 rds_superuser 权限的用户，例如 user1。

```
test=> GRANT USAGE ON FOREIGN SERVER oradb TO user1;
GRANT
```

4. 作为 user1 连接并创建到 Oracle 用户的映射。

```
test=> CREATE USER MAPPING FOR user1 SERVER oradb OPTIONS (user 'oracleuser',
 password 'mypassword');
CREATE USER MAPPING
```

5. 创建链接到 Oracle 表的外部表。

```
test=> CREATE FOREIGN TABLE mytab (a int) SERVER oradb OPTIONS (table 'MYTABLE');
CREATE FOREIGN TABLE
```

6. 查询外部表。

```
test=>  SELECT * FROM mytab;
a
---
1
(1 row)
```

如果查询报告以下错误，请检查您的安全组和访问控制列表（ACL）以确保两个实例可以通信。

```
ERROR: connection for foreign table "mytab" cannot be established
DETAIL: ORA-12170: TNS:Connect timeout occurred
```

在传输过程中使用加密

PostgreSQL-to-Oracle 传输中加密基于客户端和服务器配置参数的组合。有关使用 Oracle 21c 的示例，请参阅 Oracle 文档中的关于协商加密和完整性的值。用于 Amazon RDS 上 oracle_fdw 的客户端配置有 ACCEPTED，这意味着加密取决于 Oracle 数据库服务器配置，并且使用 Oracle 安全库（libnnz）进行加密。

如果您的数据库位于 RDS for Oracle 上，请参阅 Oracle 本机网络加密来配置加密。

了解 pg_user_mappings 视图和权限

PostgreSQL 目录 pg_user_mapping 存储 RDS for PostgreSQL 用户到外部数据（远程）服务器上用户的映射。虽然对目录的访问受到限制，但是您可以使用 pg_user_mappings 视图来查看映射。在下面，您可以找到一个示例，该示例演示如何将权限应用于示例 Oracle 数据库，但此信息更普遍地适用于任何外部数据包装器。

在以下输出中，您可以找到映射到三个不同示例用户的角色和权限。用户 rdssu1 和 rdssu2 是 rds_superuser 角色的成员，而 user1 不是。此示例使用 psql 元命令 \du 列出现有角色。

```
test=>  \du
                                                               List of roles
    Role name     |                         Attributes                         |
                  Member of
-----------------+------------------------------------------------------------
+-------------------------------------------------------------
 rdssu1           |                                                            |
 {rds_superuser}
 rdssu2           |                                                            |
 {rds_superuser}
 user1            |                                                            | {}
```

所有用户，包括具有 rds_superuser 权限的用户，都可以查看 pg_user_mappings 表中他们自己的用户映射 (umoptions)。如以下示例所示，当 rdssu1 尝试获取所有用户映射时，即使存在 rdssu1rds_superuser 权限，也会出现错误：

```
test=> SELECT * FROM pg_user_mapping;
ERROR: permission denied for table pg_user_mapping
```

下面是一些示例。

```
test=> SET SESSION AUTHORIZATION rdssu1;
SET
test=> SELECT * FROM pg_user_mappings;
 umid  | srvid | srvname | umuser | usename    |            umoptions
-------+-------+---------+--------+------------+----------------------------------
 16414 | 16411 | oradb   |  16412 | user1      |
 16423 | 16411 | oradb   |  16421 | rdssu1     | {user=oracleuser,password=mypwd}
 16424 | 16411 | oradb   |  16422 | rdssu2     |
 (3 rows)

test=> SET SESSION AUTHORIZATION rdssu2;
SET
test=> SELECT * FROM pg_user_mappings;
 umid  | srvid | srvname | umuser | usename    |            umoptions
-------+-------+---------+--------+------------+----------------------------------
 16414 | 16411 | oradb   |  16412 | user1      |
 16423 | 16411 | oradb   |  16421 | rdssu1     |
 16424 | 16411 | oradb   |  16422 | rdssu2     | {user=oracleuser,password=mypwd}
 (3 rows)

test=> SET SESSION AUTHORIZATION user1;
SET
test=> SELECT * FROM pg_user_mappings;
 umid  | srvid | srvname | umuser | usename    |           umoptions
-------+-------+---------+--------+------------+--------------------------------
 16414 | 16411 | oradb   |  16412 | user1      | {user=oracleuser,password=mypwd}
 16423 | 16411 | oradb   |  16421 | rdssu1     |
 16424 | 16411 | oradb   |  16422 | rdssu2     |
 (3 rows)
```

由于 information_schema._pg_user_mappings 和 pg_catalog.pg_user_mappings 的实施差异，手动创建的 rds_superuser 需要额外的权限才能在 pg_catalog.pg_user_mappings 中查看密码。

rds_superuser 无需额外权限即可在 information_schema._pg_user_mappings 中查看密码。

没有 rds_superuser 角色的用户只能在以下条件下在 pg_user_mappings 中查看密码：

- 当前用户是被映射的用户，拥有服务器或对其具有 USAGE 权限。
- 当前用户是服务器所有者，此映射用于 PUBLIC。

通过使用 tds_fdw 扩展来使用 SQL Server 数据库

您可以使用 PostgreSQL `tds_fdw` 扩展来访问支持表格数据流 (TDS) 协议的数据库，例如 Sybase 和 Microsoft SQL Server 数据库。此外部数据包装器可让您从 RDS for PostgreSQL 数据库实例 连接到使用 TDS 协议的数据库，包括 Amazon RDS for Microsoft SQL Server。有关更多信息，请参阅 GitHub 上的 tds-fdw/tds_fdw 文档。

Amazon RDS for PostgreSQL 版本 14.2、13.6 及更高版本支持该 `tds_fdw` 扩展。

将 Aurora PostgreSQL 数据库设置为使用 tds_fdw 扩展

在以下过程中，您可以找到设置 `tds_fdw` 并将其与 RDS for PostgreSQL 数据库实例结合使用的示例。在可以使用 `tds_fdw` 连接到 SQL Server 数据库之前，您需要获取实例的以下详细信息：

- 主机名或终端节点。对于 RDS for SQL Server 数据库实例，您可以使用控制台查找终端节点。选择 Connectivity & security（连接和安全）选项卡，然后查看 Endpoint and port（终端节点和端口）部分。
- 端口号。Microsoft SQL Server 的默认端口是 1433。
- 数据库的名称。数据库标识符。

您还需要为 SQL Server 端口 1433 提供对安全组或访问控制列表 (ACL) 的访问权限。RDS for PostgreSQL 数据库实例和 RDS for SQL Server 数据库实例都需要访问端口 1433。如果访问权限配置不正确，当您尝试查询 Microsoft SQL Server 时，会看到以下错误消息：

```
ERROR: DB-Library error: DB #: 20009, DB Msg: Unable to connect:
Adaptive Server is unavailable or does not exist (mssql2019.aws-
region.rds.amazonaws.com), OS #: 0, OS Msg: Success, Level: 9
```

使用 tds_fdw 连接到 SQL Server 数据库

1. 使用具有 `rds_superuser` 角色的账户连接到您的 PostgreSQL 数据库实例：

```
psql --host=your-DB-instance.aws-region.rds.amazonaws.com --port=5432 --
username=test --password
```

2. 安装 `tds_fdw` 扩展：

```
test=> CREATE EXTENSION tds_fdw;
CREATE EXTENSION
```

在 RDS for PostgreSQL 数据库实例上安装扩展后，应设置外部服务器。

创建外部服务器

使用具有 rds_superuser 权限的账户在 RDS for PostgreSQL 数据库实例上执行这些任务。

1. 在 RDS for PostgreSQL 数据库实例中创建外部服务器：

```
test=> CREATE SERVER sqlserverdb FOREIGN DATA WRAPPER tds_fdw OPTIONS
 (servername 'mssql2019.aws-region.rds.amazonaws.com', port '1433', database
 'tds_fdw_testing');
CREATE SERVER
```

   要访问 SQLServer 端的非 ASCII 数据，请在 RDS for PostgreSQL 数据库实例中使用 character_set 选项创建服务器链接：

```
test=> CREATE SERVER sqlserverdb FOREIGN DATA WRAPPER tds_fdw OPTIONS (servername
 'mssql2019.aws-region.rds.amazonaws.com', port '1433', database 'tds_fdw_testing',
 character_set 'UTF-8');
CREATE SERVER
```

2. 向没有 rds_superuser 角色权限的用户授予权限，例如 user1：

```
test=> GRANT USAGE ON FOREIGN SERVER sqlserverdb TO user1;
```

3. 以 user1 身份进行连接，然后创建到 SQL Server 用户的映射：

```
test=> CREATE USER MAPPING FOR user1 SERVER sqlserverdb OPTIONS (username
 'sqlserveruser', password 'password');
CREATE USER MAPPING
```

4. 创建链接到 SQL Server 表的外部表：

```
test=> CREATE FOREIGN TABLE mytab (a int) SERVER sqlserverdb OPTIONS (table
 'MYTABLE');
CREATE FOREIGN TABLE
```

5. 查询外部表：

```
test=> SELECT * FROM mytab;
 a
---
```

```
 1
(1 row)
```

使用传输中的加密进行连接

RDS for PostgreSQL 到 SQL Server 的连接使用传输中加密 (TLS/SSL)，具体取决于 SQL Server 数据库配置。如果 SQL Server 未配置为加密，则向 SQL Server 数据库发出请求的 RDS for PostgreSQL 客户端将回退到未加密状态。

您可以通过设置 `rds.force_ssl` 参数对与 RDS for SQL Server 数据库实例的连接进行加密。要了解操作方法，请参阅强制与数据库实例的连接使用 SSL。有关 RDS for SQL Server 的 SSL/TLS 配置的更多信息，请参阅将 SSL 与 Microsoft SQL Server 数据库实例结合使用。

使用适用于 PostgreSQL 的可信语言扩展

适用于 PostgreSQL 的可信语言扩展是一个用于构建 PostgreSQL 扩展的开源开发套件。它允许您构建高性能 PostgreSQL 扩展，并在您的 RDS for PostgreSQL 数据库实例上安全地运行它们。通过使用适用于 PostgreSQL 的可信语言扩展（TLE），您可以创建 PostgreSQL 扩展，这些扩展遵循已记载的用于扩展 PostgreSQL 功能的方法。有关更多信息，请参阅 PostgreSQL 文档中的将相关对象打包为扩展。

TLE 的一个关键好处是，您可以在不提供对 PostgreSQL 实例底层文件系统的访问权限的环境中使用它。以前，安装新扩展需要访问文件系统。TLE 消除了这一约束。它提供了一个开发环境，用于为任何 PostgreSQL 数据库创建新扩展，包括在 RDS for PostgreSQL 数据库实例上运行的扩展。

TLE 旨在防止访问您使用 TLE 创建的扩展的不安全资源。它的运行时环境限制了任何扩展缺陷对单个数据库连接的影响。TLE 还让数据库管理员可以细粒度控制谁可以安装扩展，并为运行扩展提供了权限模型。

以下 RDS for PostgreSQL 版本支持 TLE：

- 版本 17.1 及更高的 17 版本
- 版本 16.1 及更高的 16 版本
- 版本 15.2 及更高的 15 版本
- 版本 14.5 及更高的 14 版本
- 版本 13.12 及更高的 13 版本

可信语言扩展开发环境和运行时打包为 `pg_tle` PostgreSQL 扩展，版本 1.0.1。它支持在 JavaScript、Perl、Tcl、PL/pgSQL 和 SQL 中创建扩展。您可以在 RDS for PostgreSQL 数据库实例中安装 `pg_tle` 扩展，方式与安装其他 PostgreSQL 扩展一样。设置 `pg_tle` 后，开发人员可以使用它来创建新的 PostgreSQL 扩展，称为 TLE 扩展。

在以下主题中，您可以了解有关如何设置可信语言扩展以及如何开始创建自己的 TLE 扩展的信息。

主题

术语

为了帮助您更好地了解可信语言扩展，请查看以下词汇表，了解本主题中使用的术语。

适用于 PostgreSQL 的可信语言扩展

适用于 PostgreSQL 的可信语言扩展是打包为 `pg_tle` 扩展的开源开发套件的正式名称。它可以在任何 PostgreSQL 系统上使用。有关更多信息，请参阅 GitHub 上的 aws/pg_tle。

可信语言扩展

可信语言扩展是适用于 PostgreSQL 的可信语言扩展的简称。本文档中也使用了这个缩写名称及其缩写（TLE）。

可信语言

可信语言是一种具有特定安全属性的编程或脚本语言。例如，可信语言通常限制对文件系统的访问，并限制对指定网络属性的使用。TLE 开发套件旨在支持可信语言。PostgreSQL 支持几种不同的用于创建可信扩展或不可信扩展的语言。有关示例，请参阅 PostgreSQL 文档中的可信和不可信 PL/Perl。当您使用可信语言扩展创建扩展时，该扩展本质上使用可信语言机制。

TLE 扩展

TLE 扩展是使用可信语言扩展（TLE）开发套件创建的 PostgreSQL 扩展。

使用适用于 PostgreSQL 的可信语言扩展的要求

以下是设置和使用 TLE 开发套件的要求。

- RDS for PostgreSQL 版本 – 仅在 RDS for PostgreSQL 版本 13.12 及更高的 13 版本、14.5 及更高的 14 版本，以及 15.2 及更高的版本上支持可信语言扩展。

  - 如果您需要升级 RDS for PostgreSQL 实例，请参阅升级 RDS for PostgreSQL 数据库引擎。
  - 如果您还没有运行 PostgreSQL 的 Amazon RDS 数据库实例，则可以创建一个。有关更多信息，请参阅 RDS for PostgreSQL 数据库实例，请参阅 创建 PostgreSQL 数据库实例并连接到该实例。
- 需要 **rds_superuser** 权限 - 要设置和配置 pg_tle 扩展，您的数据库用户角色必须具有 rds_superuser 角色的权限。默认情况下，此角色被授予 postgres 用户，此用户创建 RDS for PostgreSQL 数据库实例。
- 需要自定义数据库参数组 – 您的 RDS for PostgreSQL 数据库实例必须使用自定义数据库参数组进行配置。
  - 如果未使用自定义数据库参数组配置您的 RDS for PostgreSQL 数据库实例，则应创建一个参数组并将其与 RDS for PostgreSQL 数据库实例相关联。有关步骤的简短摘要，请参阅 创建和应用自定义数据库参数组。
  - 如果已经使用自定义数据库参数组配置了 RDS for PostgreSQL 数据库实例，则可以设置可信语言扩展。有关详细信息，请参阅在 RDS for PostgreSQL 数据库实例中设置可信语言扩展。

创建和应用自定义数据库参数组

使用以下步骤创建自定义数据库参数组，并将 RDS for PostgreSQL 数据库实例配置为使用该参数组。

控制台

创建自定义数据库参数组并将其与您的 RDS for PostgreSQL 数据库实例一起使用

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 从 Amazon RDS 菜单中选择 Parameter groups（参数组）。
3. 选择创建参数组。
4. 在 Parameter group details（参数组详细信息）页面中，输入以下信息。
   - 对于Parameter group family（参数组系列），选择 postgres14。
   - 对于 Type（类型），请选择 DB Parameter Group（数据库参数组）。
   - 对于 Group name（组名称），在操作上下文中为参数组指定一个有意义的名称。
   - 对于 Description（描述），输入有用的描述，以便团队中的其他人可以轻松找到它。
5. 选择创建。您的自定义数据库参数组是在您的 Amazon Web Services 区域中创建的。现在，您可以按照以下步骤修改 RDS for PostgreSQL 数据库实例以使用它。

6. 从 Amazon RDS 菜单中选择 Databases（数据库）。
7. 从列出的列表中选择要与 TLE 结合使用的 RDS for PostgreSQL 数据库实例，然后选择 Modify（修改）。
8. 在 Modify DB instance settings（修改数据库实例设置）页面中，在 Additional configuration（其他配置）部分中找到 Database options（数据库选项），并从选择器中选择您的自定义数据库参数组。
9. 选择 Continue（继续）以保存更改。
10. 选择 Apply immediately（立即应用），这样您就可以继续将 RDS for PostgreSQL 数据库实例设置为使用 TLE。

要继续为系统设置可信语言扩展，请参阅 在 RDS for PostgreSQL 数据库实例中设置可信语言扩展。

有关使用数据库参数组的更多信息，请参阅 Amazon RDS 数据库实例的数据库参数组。

Amazon CLI

在使用 CLI 命令时，您可以通过将您的 Amazon CLI 配置为使用默认 Amazon Web Services 区域来避免指定 `--region` 参数。有关更多信息，请参阅《Amazon Command Line Interface 用户指南》中的配置基础知识。

创建自定义数据库参数组并将其与您的 RDS for PostgreSQL 数据库实例一起使用

1. 使用 create-db-parameter-group Amazon CLI 命令为您的 Amazon Web Services 区域创建一个基于 postgres14 的自定义数据库参数组。

   对于 Linux、macOS 或 Unix：

```
aws rds create-db-parameter-group \
  --region aws-region \
  --db-parameter-group-name custom-params-for-pg-tle \
  --db-parameter-group-family postgres14 \
  --description "My custom DB parameter group for Trusted Language Extensions"
```

   对于 Windows：

```
aws rds create-db-parameter-group ^
  --region aws-region ^
  --db-parameter-group-name custom-params-for-pg-tle ^
  --db-parameter-group-family postgres14 ^
```

```
  --description "My custom DB parameter group for Trusted Language Extensions"
```

您的自定义数据库参数组在 Amazon Web Services 区域中可用，因此您可以修改 RDS for PostgreSQL 数据库实例以使用它。

2. 使用 modify-db-instance Amazon CLI 命令将自定义数据库参数组应用于 RDS for PostgreSQL 数据库实例。此命令立即重启活动实例。

   对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
  --region aws-region \
  --db-instance-identifier your-instance-name \
  --db-parameter-group-name custom-params-for-pg-tle \
  --apply-immediately
```

   对于 Windows：

```
aws rds modify-db-instance ^
  --region aws-region ^
  --db-instance-identifier your-instance-name ^
  --db-parameter-group-name custom-params-for-pg-tle ^
  --apply-immediately
```

要继续为系统设置可信语言扩展，请参阅 在 RDS for PostgreSQL 数据库实例中设置可信语言扩展。

有关更多信息，请参阅 Amazon RDS 的参数组。

在 RDS for PostgreSQL 数据库实例中设置可信语言扩展

以下步骤假设您的 RDS for PostgreSQL 数据库实例与自定义数据库参数组相关联。您可以使用 Amazon Web Services Management Console或 Amazon CLI 来执行这些步骤。

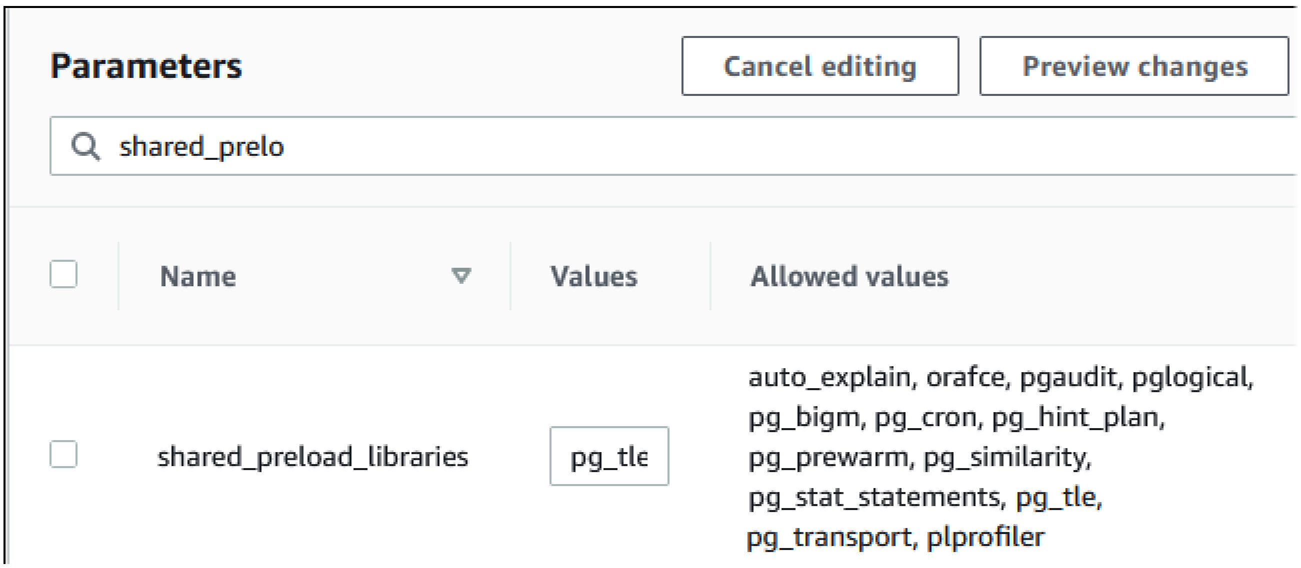
当您在 RDS for PostgreSQL 数据库实例中设置可信语言扩展时，您可以将其安装在特定的数据库中，供对该数据库拥有权限的数据库用户使用。

控制台

设置可信语言扩展

使用作为 `rds_superuser` 组（角色）成员的账户执行以下步骤。

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 RDS for PostgreSQL 数据库实例。
3. 打开 的配置选项卡。RDS for PostgreSQL 数据库实例的 Configuration（配置）选项卡。在实例详细信息中，找到 Parameter group（参数组）链接。
4. 选择此链接以打开与您的 RDS for PostgreSQL 数据库实例关联的自定义参数。
5. 在 Parameters（参数）搜索字段中，键入 `shared_pre` 以查找 `shared_preload_libraries` 参数。
6. 选择 Edit parameters（编辑参数）以访问属性值。
7. 将 `pg_tle` 添加到 Values（值）字段的列表中。使用逗号分隔值列表中的项目。



8. 重启 RDS for PostgreSQL 数据库实例，以使对 `shared_preload_libraries` 参数的更改生效。
9. 当实例可用时，验证 `pg_tle` 是否已初始化。使用 `psql` 连接到 RDS for PostgreSQL 数据库实例，然后运行以下命令。

```
SHOW shared_preload_libraries;
shared_preload_libraries
--------------------------
rdsutils,pg_tle
(1 row)
```

10. 初始化 `pg_tle` 扩展后，您现在可以创建此扩展了。

```
CREATE EXTENSION pg_tle;
```

可以使用以下 psql 元命令验证扩展是否已安装。

```
labdb=> \dx
                         List of installed extensions
  Name    | Version |   Schema   |                Description
---------+---------+------------+--------------------------------------------
 pg_tle   | 1.0.1   | pgtle      | Trusted-Language Extensions for PostgreSQL
 plpgsql | 1.0     | pg_catalog | PL/pgSQL procedural language
```

11. 在设置 RDS for PostgreSQL 数据库实例时，将 pgtle_admin 角色授予为其创建的主用户名。如果您接受了默认值，则它就是 postgres。

```
labdb=> GRANT pgtle_admin TO postgres;
GRANT ROLE
```

您可以使用 psql 元命令来验证授权是否已完成，如以下示例中所示。输出中仅显示 pgtle_admin 和 postgres 角色。有关更多信息，请参阅了解 rds_superuser 角色。

```
labdb=> \du
                          List of roles
    Role name     |           Attributes            |               Member of
-----------------+---------------------------------
+-----------------------------------
pgtle_admin      | Cannot login                     | {}
postgres         | Create role, Create DB          +| {rds_superuser,pgtle_admin}
                 | Password valid until infinity    |...
```

12. 使用 \q 元命令关闭 psql 会话。

```
\q
```

要开始创建 TLE 扩展，请参阅 示例：使用 SQL 创建可信语言扩展。

Amazon CLI

在使用 CLI 命令时，您可以通过将您的 Amazon CLI 配置为使用默认 Amazon Web Services 区域来避免指定 --region 参数。有关更多信息，请参阅《Amazon Command Line Interface 用户指南》中的配置基础知识。

设置可信语言扩展

1. 使用 modify-db-parameter-group Amazon CLI 命令将 pg_tle 添加到 shared_preload_libraries 参数。

```
aws rds modify-db-parameter-group \
   --db-parameter-group-name custom-param-group-name \
   --parameters
 "ParameterName=shared_preload_libraries,ParameterValue=pg_tle,ApplyMethod=pending-
reboot" \
   --region aws-region
```

2. 使用 reboot-db-instance Amazon CLI 命令重启 RDS for PostgreSQL 数据库实例并初始化 pg_tle 库。

```
aws rds reboot-db-instance \
    --db-instance-identifier your-instance \
    --region aws-region
```

3. 当实例可用时，您可以验证 pg_tle 是否已初始化。使用 psql 连接到 RDS for PostgreSQL 数据库实例，然后运行以下命令。

```
SHOW shared_preload_libraries;
shared_preload_libraries
--------------------------
rdsutils,pg_tle
(1 row)
```

   初始化 pg_tle 后，您现在可以创建扩展了。

```
CREATE EXTENSION pg_tle;
```

4. 在设置 RDS for PostgreSQL 数据库实例时，将 pgtle_admin 角色授予为其创建的主用户名。如果您接受了默认值，则它就是 postgres。

```
GRANT pgtle_admin TO postgres;
GRANT ROLE
```

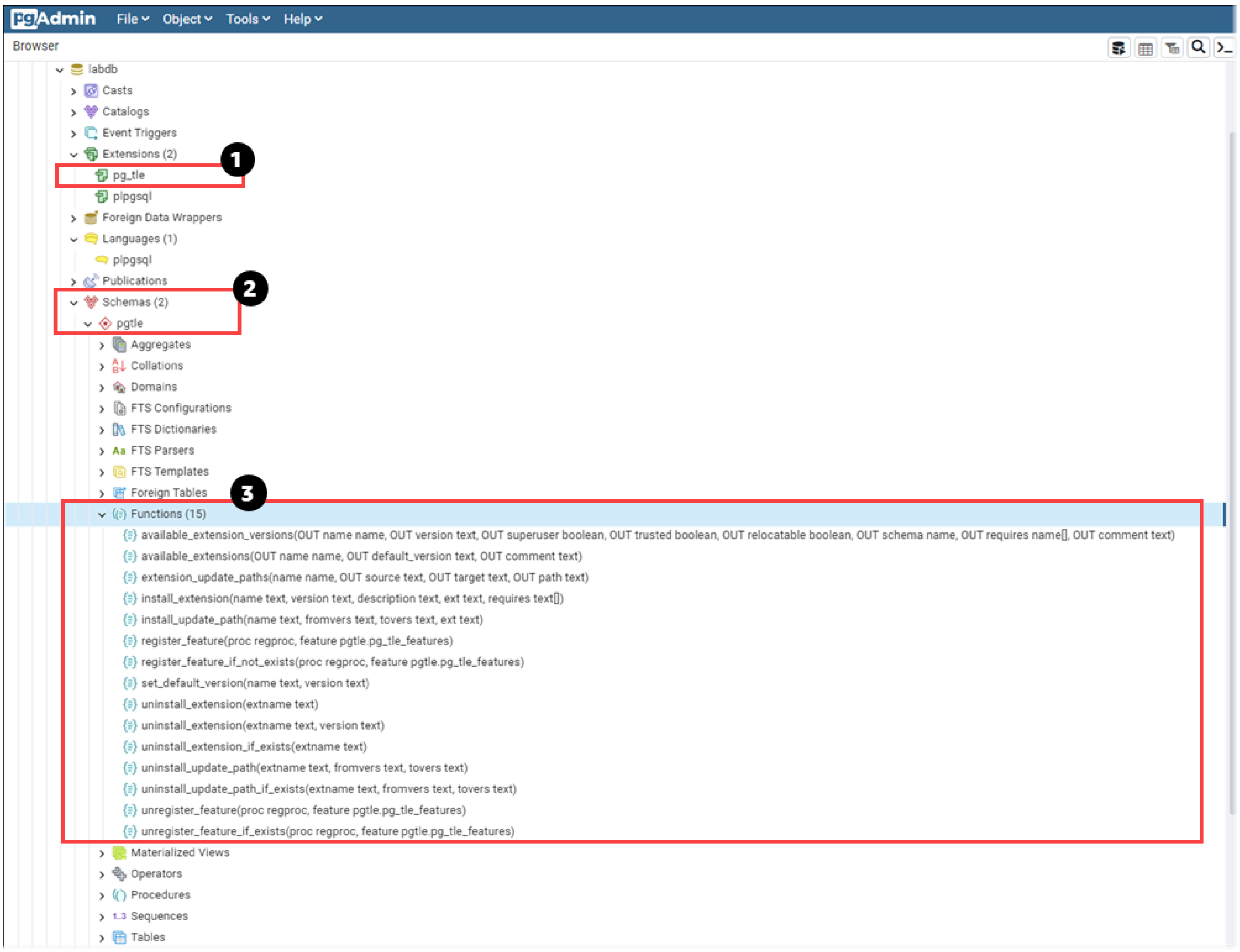
5. 按如下方式关闭 psql 会话。

```
labdb=> \q
```

要开始创建 TLE 扩展，请参阅 示例：使用 SQL 创建可信语言扩展。

适用于 PostgreSQL 的可信语言扩展概述

适用于 PostgreSQL 的可信语言扩展是一个 PostgreSQL 扩展，您可以像设置其他 PostgreSQL 扩展一样，将其安装在 RDS for PostgreSQL 数据库实例中。在下图的 pgAdmin 客户端工具的示例数据库中，您可以查看构成 `pg_tle` 扩展的部分组件。



您可以查看以下详细信息。

1. 适用于 PostgreSQL 的可信语言扩展（TLE）开发套件打包为 `pg_tle` 扩展。因此，`pg_tle` 添加到安装它的数据库的可用扩展中。
2. TLE 有它自己的模式，即 `pgtle`。此模式包含辅助函数（3 个），用于安装和管理您创建的扩展。
3. TLE 提供了十多个辅助函数，用于安装、注册和管理您的扩展。要了解有关这些函数的更多信息，请参阅 适用于 PostgreSQL 的可信语言扩展的函数参考。

`pg_tle` 扩展的其他组件包含以下各项：

- **`pgtle_admin` 角色** – `pgtle_admin` 角色是在安装 `pg_tle` 扩展时创建的。此角色获得了相关权限，应视为具有此类权限。我们强烈建议您在向数据库用户授予 `pgtle_admin` 角色时遵循最低权限原则。换句话说，仅向允许创建、安装和管理新 TLE 扩展（例如 `postgres`）的数据库用户授予 `pgtle_admin` 角色。
- **`pgtle.feature_info` 表** – `pgtle.feature_info` 表是一个受保护的表，其中包含有关您的 TLE、挂钩及其使用的自定义存储过程和函数的信息。如果您有 `pgtle_admin` 权限，则可以使用以下可信语言扩展函数在表中添加和更新该信息。
  - pgtle.register_feature
  - pgtle.register_feature_if_not_exists
  - pgtle.unregister_feature
  - pgtle.unregister_feature_if_exists

为 RDS for PostgreSQL 创建 TLE 扩展

您可以将使用 TLE 创建的任何扩展安装在安装了 `pg_tle` 扩展的任何 RDS for PostgreSQL 数据库实例中。`pg_tle` 扩展的范围限于安装该扩展的 PostgreSQL 数据库。使用 TLE 创建的扩展的范围限于同一个数据库。

使用各种 `pgtle` 函数安装构成 TLE 扩展的代码。以下可信语言扩展函数全部需要 `pgtle_admin` 角色。

- pgtle.install_extension
- pgtle.install_update_path
- pgtle.register_feature
- pgtle.register_feature_if_not_exists
- pgtle.set_default_version

- pgtle.uninstall_extension(name)
- pgtle.uninstall_extension(name, version)
- pgtle.uninstall_extension_if_exists
- pgtle.uninstall_update_path
- pgtle.uninstall_update_path_if_exists
- pgtle.unregister_feature
- pgtle.unregister_feature_if_exists

示例：使用 SQL 创建可信语言扩展

以下示例说明如何创建名为 pg_distance 的 TLE 扩展，该扩展包含一些 SQL 函数，用于使用不同的公式计算距离。在列表中，您可以找到用于计算曼哈顿距离的函数和用于计算欧几里得距离的函数。有关这些公式之间差异的更多信息，请参阅维基百科中的 Taxicab 几何和欧几里得几何。

如果您按照在 RDS for PostgreSQL 数据库实例中设置可信语言扩展中详述设置了 pg_tle 扩展，则可以在自己的 RDS for PostgreSQL 数据库实例中使用此示例。

> **Note**
> 您需要拥有 pgtle_admin 角色的权限才能执行此过程。

创建示例 TLE 扩展

以下步骤使用名为 labdb 的示例数据库。该数据库归 postgres 主用户所有。postgres 角色还具有 pgtle_admin 角色的权限。

1. 使用 psql 连接到 RDS for PostgreSQL 数据库实例。

```
psql --host=db-instance-123456789012.aws-region.rds.amazonaws.com
--port=5432 --username=postgres --password --dbname=labdb
```

2. 通过复制以下代码并将其粘贴到 psql 会话控制台中来创建名为 pg_distance 的 TLE 扩展。

```
SELECT pgtle.install_extension
(
 'pg_distance',
```

```
 '0.1',
  'Distance functions for two points',
$_pg_tle_$
    CREATE FUNCTION dist(x1 float8, y1 float8, x2 float8, y2 float8, norm int)
    RETURNS float8
    AS $$
      SELECT (abs(x2 - x1) ^ norm + abs(y2 - y1) ^ norm) ^ (1::float8 / norm);
    $$ LANGUAGE SQL;

    CREATE FUNCTION manhattan_dist(x1 float8, y1 float8, x2 float8, y2 float8)
    RETURNS float8
    AS $$
      SELECT dist(x1, y1, x2, y2, 1);
    $$ LANGUAGE SQL;

    CREATE FUNCTION euclidean_dist(x1 float8, y1 float8, x2 float8, y2 float8)
    RETURNS float8
    AS $$
      SELECT dist(x1, y1, x2, y2, 2);
    $$ LANGUAGE SQL;
$_pg_tle_$
);
```

您将看到如下输出。

```
install_extension
---------------
 t
(1 row)
```

构成 pg_distance 扩展的构件现在已安装在您的数据库中。这些构件包括扩展的控制文件和代码，这些项目需要存在，这样才能使用 CREATE EXTENSION 命令创建扩展。换句话说，您仍然需要创建扩展以使其函数可供数据库用户使用。

3. 要创建扩展，请像使用任何其他扩展一样使用 CREATE EXTENSION 命令。与其他扩展一样，数据库用户需要在数据库中拥有 CREATE 权限。

```
CREATE EXTENSION pg_distance;
```

4. 要测试 pg_distance TLE 扩展，可以使用它来计算四个点之间的曼哈顿距离。

```
labdb=> SELECT manhattan_dist(1, 1, 5, 5);
```

```
8
```

要计算同一组点之间的欧几里得距离，可以使用以下方法。

```
labdb=> SELECT euclidean_dist(1, 1, 5, 5);
5.656854249492381
```

pg_distance 扩展将函数加载到数据库中，并使对数据库具有权限的任何用户都可以使用它们。

修改您的 TLE 扩展

要提高此 TLE 扩展中打包的函数的查询性能，请在其规范中添加以下两个 PostgreSQL 属性。

- IMMUTABLE – IMMUTABLE 属性可确保查询优化程序可以使用优化措施来缩短查询响应时间。有关更多信息，请参阅 PostgreSQL 文档中的函数波动性类别。
- PARALLEL SAFE – PARALLEL SAFE 属性是允许 PostgreSQL 在并行模式下运行该函数的另一个属性。有关更多信息，请参阅 PostgreSQL 文档中的 CREATE FUNCTION。

在以下示例中，您可以看到如何使用 pgtle.install_update_path 函数向每个函数添加这些属性，以创建 pg_distance TLE 扩展的版本 0.2。有关此函数的更多信息，请参阅 pgtle.install_update_path。您需要拥有 pgtle_admin 角色才能执行此任务。

更新现有 TLE 扩展并指定默认版本

1. 使用 psql 或其他客户端工具（例如 pgAdmin）连接到 RDS for PostgreSQL 数据库实例。

```
psql --host=db-instance-123456789012.aws-region.rds.amazonaws.com
--port=5432 --username=postgres --password --dbname=labdb
```

2. 通过复制以下代码并将其粘贴到 psql 会话控制台中来修改现有 TLE 扩展。

```
SELECT pgtle.install_update_path
(
 'pg_distance',
 '0.1',
 '0.2',
$_pg_tle_$
    CREATE OR REPLACE FUNCTION dist(x1 float8, y1 float8, x2 float8, y2 float8,
 norm int)
```

```
    RETURNS float8
    AS $$
      SELECT (abs(x2 - x1) ^ norm + abs(y2 - y1) ^ norm) ^ (1::float8 / norm);
    $$ LANGUAGE SQL IMMUTABLE PARALLEL SAFE;

    CREATE OR REPLACE FUNCTION manhattan_dist(x1 float8, y1 float8, x2 float8, y2
 float8)
    RETURNS float8
    AS $$
      SELECT dist(x1, y1, x2, y2, 1);
    $$ LANGUAGE SQL IMMUTABLE PARALLEL SAFE;

    CREATE OR REPLACE FUNCTION euclidean_dist(x1 float8, y1 float8, x2 float8, y2
 float8)
    RETURNS float8
    AS $$
      SELECT dist(x1, y1, x2, y2, 2);
    $$ LANGUAGE SQL IMMUTABLE PARALLEL SAFE;
$_pg_tle_$
);
```

您看到的响应与以下内容类似。

```
install_update_path
---------------------
 t
(1 row)
```

您可以将此版本的扩展设置为默认版本，这样，数据库用户在其数据库中创建或更新扩展时就不必指定版本。

3. 要将 TLE 扩展的修改版本（版本 0.2）指定为默认版本，请使用以下示例所示的 `pgtle.set_default_version` 函数。

```
SELECT pgtle.set_default_version('pg_distance', '0.2');
```

有关此函数的更多信息，请参阅 pgtle.set_default_version。

4. 代码准备就绪后，您可以使用 `ALTER EXTENSION ... UPDATE` 命令以常规方式更新已安装的 TLE 扩展，如此处所示：

```
ALTER EXTENSION pg_distance UPDATE;
```

从数据库中删除 TLE 扩展

您可以使用 DROP EXTENSION 命令删除 TLE 扩展，方法与处理其他 PostgreSQL 扩展的方法相同。删除扩展并不会删除构成扩展的安装文件，这允许用户重新创建扩展。要删除扩展及其安装文件，请执行以下两步过程。

删除 TLE 扩展并删除其安装文件

1. 使用 psql 或其他客户端工具连接到 RDS for PostgreSQL 数据库实例。

```
psql --host=.111122223333.aws-region.rds.amazonaws.com --port=5432 --
username=postgres --password --dbname=dbname
```

2. 像删除任何 PostgreSQL 扩展一样删除此扩展。

```
DROP EXTENSION your-TLE-extension
```

例如，如果您按照示例：使用 SQL 创建可信语言扩展中的详细说明创建 pg_distance 扩展，则可以按如下方式删除此扩展。

```
DROP EXTENSION pg_distance;
```

您将看到确认扩展已删除的输出，如下所示。

```
DROP EXTENSION
```

此时，该扩展在数据库中不再处于活动状态。但是，它的安装文件和控制文件在数据库中仍然可用，因此数据库用户可以根据需要再次创建扩展。

- 如果您想让扩展文件保持完好，以便数据库用户可以创建您的 TLE 扩展，则可以在此处停止。
- 如果要删除构成扩展的所有文件，请继续执行下一步。

3. 要删除扩展的所有安装文件，请使用 pgtle.uninstall_extension 函数。此函数删除扩展的所有代码和控制文件。

```
SELECT pgtle.uninstall_extension('your-tle-extension-name');
```

例如，要删除所有 pg_distance 安装文件，请使用以下命令。

```
SELECT pgtle.uninstall_extension('pg_distance');
 uninstall_extension
---------------------
 t
(1 row)
```

卸载适用于 PostgreSQL 的可信语言扩展

如果您不想再使用 TLE 创建自己的 TLE 扩展，则可以删除 pg_tle 扩展并删除所有构件。此操作包括删除数据库中的所有 TLE 扩展和删除 pgtle 模式。

从数据库中删除 **pg_tle** 扩展及其模式

1. 使用 psql 或其他客户端工具连接到 RDS for PostgreSQL 数据库实例。

   ```
   psql --host=.111122223333.aws-region.rds.amazonaws.com --port=5432 --
   username=postgres --password --dbname=dbname
   ```

2. 从数据库中删除 pg_tle 扩展。如果数据库中仍在运行您自己的 TLE 扩展，则还需要删除这些扩展。为此，您可以使用 CASCADE 关键字，如以下所示。

   ```
   DROP EXTENSION pg_tle CASCADE;
   ```

   如果 pg_tle 扩展在数据库中仍未处于活动状态，则无需使用 CASCADE 关键字。

3. 删除 pgtle 模式。此操作将从数据库中删除所有管理函数。

   ```
   DROP SCHEMA pgtle CASCADE;
   ```

   该过程完成后，该命令将返回以下内容。

   ```
   DROP SCHEMA
   ```

   pg_tle 扩展、其模式和函数以及所有构件均已删除。要使用 TLE 创建新扩展，请再次完成设置过程。有关更多信息，请参阅 在 RDS for PostgreSQL 数据库实例中设置可信语言扩展。

在您的 TLE 扩展中使用 PostgreSQL 挂钩

挂钩是 PostgreSQL 中可用的一种回调机制，它允许开发人员在常规数据库操作期间调用自定义函数或其他例程。TLE 开发套件支持 PostgreSQL 挂钩，因此您可以在运行时将自定义函数与 PostgreSQL 行为集成在一起。例如，您可以使用挂钩将身份验证过程与您自己的自定义代码关联起来，或者根据您的特定需求修改查询规划和执行流程。

您的 TLE 扩展可以使用挂钩。如果挂钩在作用域方面是全局的，则它适用于所有数据库。因此，如果您的 TLE 扩展使用全局挂钩，则需要在用户可以访问的所有数据库中创建 TLE 扩展。

当您使用 `pg_tle` 扩展构建自己的可信语言扩展时，您可以使用 SQL API 中的可用挂钩来构建扩展的功能。您应该向 `pg_tle` 注册任何挂钩。对于某些挂钩，您可能还需要设置各种配置参数。例如，可以将 `passcode` 检查挂钩设置为 on、off 或 require。有关可用 `pg_tle` 挂钩的特定要求的更多信息，请参阅 适用于 PostgreSQL 的可信语言扩展的挂钩参考。

示例：创建使用 PostgreSQL 挂钩的扩展

本节讨论的示例使用 PostgreSQL 挂钩检查在特定 SQL 操作期间提供的密码，并防止数据库用户将其密码设置为 `password_check.bad_passwords` 表中包含的任何密码。该表包含十大最常用但易于破解的密码选择。

要在 RDS for PostgreSQL 数据库实例中设置此示例，您必须已经安装了可信语言扩展。有关更多信息，请参阅 在 RDS for PostgreSQL 数据库实例中设置可信语言扩展。

设置密码检查挂钩示例

1. 使用 `psql` 连接到 RDS for PostgreSQL 数据库实例。

```
psql --host=db-instance-123456789012.aws-region.rds.amazonaws.com
--port=5432 --username=postgres --password --dbname=labdb
```

2. 从密码检查挂钩代码列表中复制代码并将其粘贴到数据库中。

```
SELECT pgtle.install_extension (
  'my_password_check_rules',
  '1.0',
  'Do not let users use the 10 most commonly used passwords',
$_pgtle_$
  CREATE SCHEMA password_check;
  REVOKE ALL ON SCHEMA password_check FROM PUBLIC;
  GRANT USAGE ON SCHEMA password_check TO PUBLIC;
```

```
 CREATE TABLE password_check.bad_passwords (plaintext) AS
 VALUES
   ('123456'),
   ('password'),
   ('12345678'),
   ('qwerty'),
   ('123456789'),
   ('12345'),
   ('1234'),
   ('111111'),
   ('1234567'),
   ('dragon');
 CREATE UNIQUE INDEX ON password_check.bad_passwords (plaintext);

 CREATE FUNCTION password_check.passcheck_hook(username text, password text,
password_type pgtle.password_types, valid_until timestamptz, valid_null boolean)
 RETURNS void AS $$
   DECLARE
     invalid bool := false;
   BEGIN
     IF password_type = 'PASSWORD_TYPE_MD5' THEN
       SELECT EXISTS(
         SELECT 1
         FROM password_check.bad_passwords bp
         WHERE ('md5' || md5(bp.plaintext || username)) = password
       ) INTO invalid;
       IF invalid THEN
         RAISE EXCEPTION 'Cannot use passwords from the common password
dictionary';
       END IF;
     ELSIF password_type = 'PASSWORD_TYPE_PLAINTEXT' THEN
       SELECT EXISTS(
         SELECT 1
         FROM password_check.bad_passwords bp
         WHERE bp.plaintext = password
       ) INTO invalid;
       IF invalid THEN
         RAISE EXCEPTION 'Cannot use passwords from the common password
dictionary';
       END IF;
     END IF;
   END
 $$ LANGUAGE plpgsql SECURITY DEFINER;
```

```
  GRANT EXECUTE ON FUNCTION password_check.passcheck_hook TO PUBLIC;

  SELECT pgtle.register_feature('password_check.passcheck_hook', 'passcheck');
$_pgtle_$
);
```

将扩展加载到数据库后，您会看到如下输出。

```
 install_extension
-------------------
 t
(1 row)
```

3. 当仍然连接到数据库时，现在可以创建扩展了。

```
CREATE EXTENSION my_password_check_rules;
```

4. 您可以使用以下 psql 元命令确认已在数据库中创建扩展。

```
\dx
                        List of installed extensions
          Name           | Version |   Schema   |
 Description
-------------------------+---------+------------
+-------------------------------------------------------------
 my_password_check_rules | 1.0     | public     | Prevent use of any of the top-ten
 most common bad passwords
 pg_tle                  | 1.0.1   | pgtle      | Trusted-Language Extensions for
 PostgreSQL
 plpgsql                 | 1.0     | pg_catalog | PL/pgSQL procedural language
(3 rows)
```

5. 打开另一个终端会话来使用 Amazon CLI。您需要修改您的自定义数据库参数组才能开启密码检查挂钩。为此，请使用 modify-db-parameter-group CLI 命令，如以下示例中所示。

```
aws rds modify-db-parameter-group \
    --region aws-region \
    --db-parameter-group-name your-custom-parameter-group \
    --parameters
 "ParameterName=pgtle.enable_password_check,ParameterValue=on,ApplyMethod=immediate"
```

成功开启此参数后，您将看到如下输出。

```
{
    "DBParameterGroupName": "docs-lab-parameters-for-tle"
}
```

对参数组设置进行的更改可能需要几分钟才能生效。但是，此参数是动态的，因此您无需重新启动 RDS for PostgreSQL 数据库实例，即可使该设置生效。

6. 打开 `psql` 会话并查询数据库，以验证密码检查挂钩已开启。

```
labdb=> SHOW pgtle.enable_password_check;
pgtle.enable_password_check
-----------------------------
on
(1 row)
```

密码检查挂钩现处于活动状态。您可以通过创建新角色并使用其中一个错误密码来对其进行测试，如以下示例中所示。

```
CREATE ROLE test_role PASSWORD 'password';
ERROR:  Cannot use passwords from the common password dictionary
CONTEXT:  PL/pgSQL function
 password_check.passcheck_hook(text,text,pgtle.password_types,timestamp with time
 zone,boolean) line 21 at RAISE
SQL statement "SELECT password_check.passcheck_hook(
    $1::pg_catalog.text,
    $2::pg_catalog.text,
    $3::pgtle.password_types,
    $4::pg_catalog.timestamptz,
    $5::pg_catalog.bool)"
```

对输出设置了格式以便于阅读。

以下示例显示，`pgsql` 交互式元命令 `\password` 行为也受到密码检查挂钩的影响。

```
postgres=> SET password_encryption TO 'md5';
SET
postgres=> \password
Enter new password for user "postgres":*****
Enter it again:*****
ERROR:  Cannot use passwords from the common password dictionary
```

```
CONTEXT:  PL/pgSQL function
 password_check.passcheck_hook(text,text,pgtle.password_types,timestamp with time
 zone,boolean) line 12 at RAISE
SQL statement "SELECT password_check.passcheck_hook($1::pg_catalog.text,
 $2::pg_catalog.text, $3::pgtle.password_types, $4::pg_catalog.timestamptz,
 $5::pg_catalog.bool)"
```

如果需要，可以删除此 TLE 扩展并卸载其源文件。有关更多信息，请参阅 从数据库中删除 TLE 扩展。

密码检查挂钩代码列表

此处显示的示例代码定义了 my_password_check_rules TLE 扩展的规范。当您复制此代码并将其粘贴到数据库中时，my_password_check_rules 扩展的代码将加载到数据库中，并注册 password_check 挂钩以供扩展使用。

```
SELECT pgtle.install_extension (
  'my_password_check_rules',
  '1.0',
  'Do not let users use the 10 most commonly used passwords',
$_pgtle_$
  CREATE SCHEMA password_check;
  REVOKE ALL ON SCHEMA password_check FROM PUBLIC;
  GRANT USAGE ON SCHEMA password_check TO PUBLIC;

  CREATE TABLE password_check.bad_passwords (plaintext) AS
  VALUES
    ('123456'),
    ('password'),
    ('12345678'),
    ('qwerty'),
    ('123456789'),
    ('12345'),
    ('1234'),
    ('111111'),
    ('1234567'),
    ('dragon');
  CREATE UNIQUE INDEX ON password_check.bad_passwords (plaintext);

  CREATE FUNCTION password_check.passcheck_hook(username text, password text,
 password_type pgtle.password_types, valid_until timestamptz, valid_null boolean)
  RETURNS void AS $$
    DECLARE
```

```
      invalid bool := false;
    BEGIN
      IF password_type = 'PASSWORD_TYPE_MD5' THEN
        SELECT EXISTS(
          SELECT 1
          FROM password_check.bad_passwords bp
          WHERE ('md5' || md5(bp.plaintext || username)) = password
        ) INTO invalid;
        IF invalid THEN
          RAISE EXCEPTION 'Cannot use passwords from the common password dictionary';
        END IF;
      ELSIF password_type = 'PASSWORD_TYPE_PLAINTEXT' THEN
        SELECT EXISTS(
          SELECT 1
          FROM password_check.bad_passwords bp
          WHERE bp.plaintext = password
        ) INTO invalid;
        IF invalid THEN
          RAISE EXCEPTION 'Cannot use passwords from the common password dictionary';
        END IF;
      END IF;
    END
  $$ LANGUAGE plpgsql SECURITY DEFINER;

  GRANT EXECUTE ON FUNCTION password_check.passcheck_hook TO PUBLIC;

  SELECT pgtle.register_feature('password_check.passcheck_hook', 'passcheck');
$_pgtle_$
);
```

在 TLE 中使用自定义数据类型

PostgreSQL 支持注册新的基本类型（也称为标量类型）的命令，以便高效地处理数据库中的复杂数据结构。基本类型允许您自定义在内部存储数据的方式，以及如何将其与外部文本表示形式相互转换。在扩展 PostgreSQL 以支持功能域时，这些自定义数据类型非常有用，在这种情况下，诸如数字或文本之类的内置类型无法提供足够的搜索语义。

RDS for PostgreSQL 允许您在可信语言扩展中创建自定义数据类型，并为这些新的数据类型定义支持 SQL 和索引操作的函数。自定义数据类型适用于以下版本：

- RDS for PostgreSQL 15.4 和更高的 15 版本
- RDS for PostgreSQL 14.9 和更高的 14 版本

- RDS for PostgreSQL 13.12 和更高的 13 版本

有关更多信息，请参阅可信语言库类型。

适用于 PostgreSQL 的可信语言扩展的函数参考

查看以下有关适用于 PostgreSQL 的可信语言扩展中提供的函数的参考文档。使用这些函数安装、注册、更新和管理您的 TLE 扩展，即您使用可信语言扩展开发套件开发的 PostgreSQL 扩展。

函数

pgtle.available_extensions

`pgtle.available_extensions` 函数是一个集合返回函数。它返回数据库中所有可用的 TLE 扩展。返回的每一行都包含有关单个 TLE 扩展的信息。

函数原型

```
pgtle.available_extensions()
```

角色

无。

参数

无。

输出

- name – TLE 扩展的名称。
- default_version – 在未指定版本的情况下调用 CREATE EXTENSION 时要使用的 TLE 扩展的版本。
- description – 有关 TLE 扩展的更详细描述。

用法示例

```
SELECT * FROM pgtle.available_extensions();
```

pgtle.available_extension_versions

available_extension_versions 函数是一个集合返回函数。它返回所有可用 TLE 扩展及其版本的列表。每行都包含有关给定 TLE 扩展的特定版本的信息，包括它是否需要特定角色。

函数原型

```
pgtle.available_extension_versions()
```

角色

无。

参数

无。

输出

- name – TLE 扩展的名称。

- version – TLE 扩展的版本。
- superuser – 对于您的 TLE 扩展，此值始终为 false。创建 TLE 扩展或更新 TLE 扩展所需的权限与在给定数据库中创建其他对象所需的权限相同。
- trusted – 对于您的 TLE 扩展，此值始终为 false。
- relocatable – 对于您的 TLE 扩展，此值始终为 false。
- schema – 指定安装 TLE 扩展的模式的名称。
- requires – 包含此 TLE 扩展所需的其他扩展的名称的数组。
- description – TLE 扩展的详细描述。

有关输出值的更多信息，请参阅 PostgreSQL 文档中的将相关对象打包为扩展 > 扩展文件。

用法示例

```
SELECT * FROM pgtle.available_extension_versions();
```

pgtle.extension_update_paths

extension_update_paths 函数是一个集合返回函数。它返回 TLE 扩展的所有可能更新路径的列表。每行都包含该 TLE 扩展的可用升级或降级。

函数原型

```
pgtle.extension_update_paths(name)
```

角色

无。

参数

name – 从中获取升级路径的 TLE 扩展的名称。

输出

- source – 更新的源版本。
- target – 更新的目标版本。

- path – 用于将 TLE 扩展从 source 版本更新到 target 版本的升级路径，例如 0.1--0.2。

用法示例

```
SELECT * FROM pgtle.extension_update_paths('your-TLE');
```

pgtle.install_extension

install_extension 函数允许您在数据库中安装构成 TLE 扩展的构件，之后可以使用 CREATE EXTENSION 命令创建 TLE 扩展。

函数原型

```
pgtle.install_extension(name text, version text, description text, ext text, requires
 text[] DEFAULT NULL::text[])
```

角色

无。

参数

- name – TLE 扩展的名称。调用 CREATE EXTENSION 时使用此值。
- version – TLE 扩展的版本。
- description – 有关 TLE 扩展的详细描述。此描述显示在 pgtle.available_extensions() 的 comment 字段中。
- ext – TLE 扩展的内容。此值包含诸如函数之类的对象。
- requires – 一个可选参数，用于指定此 TLE 扩展的依赖项。pg_tle 扩展会自动添加为依赖项。

其中许多参数与扩展控制文件中包含的参数相同，用于在 PostgreSQL 实例的文件系统上安装 PostgreSQL 扩展。有关更多信息，请参阅 PostgreSQL 文档的将相关对象打包为扩展中的扩展文件。

输出

此函数在成功时返回 OK，在出现错误时返回 NULL。

- OK – TLE 扩展已成功安装在数据库中。

- NULL – TLE 扩展未成功安装在数据库中。

用法示例

```
SELECT pgtle.install_extension(
 'pg_tle_test',
 '0.1',
 'My first pg_tle extension',
$_pgtle_$
  CREATE FUNCTION my_test()
  RETURNS INT
  AS $$
    SELECT 42;
  $$ LANGUAGE SQL IMMUTABLE;
$_pgtle_$
);
```

pgtle.install_update_path

`install_update_path` 函数提供 TLE 扩展的两个不同版本之间的更新路径。此功能允许您的 TLE 扩展的用户使用 `ALTER EXTENSION ... UPDATE` 语法更新其版本。

函数原型

```
pgtle.install_update_path(name text, fromvers text, tovers text, ext text)
```

角色

`pgtle_admin`

参数

- `name` – TLE 扩展的名称。调用 `CREATE EXTENSION` 时使用此值。
- `fromvers` – 用于升级的 TLE 扩展的源版本。
- `tovers` – 用于升级的 TLE 扩展的目标版本。
- `ext` – 更新的内容。此值包含诸如函数之类的对象。

输出

无。

用法示例

```
SELECT pgtle.install_update_path('pg_tle_test', '0.1', '0.2',
  $_pgtle_$
    CREATE OR REPLACE FUNCTION my_test()
    RETURNS INT
    AS $$
      SELECT 21;
    $$ LANGUAGE SQL IMMUTABLE;
  $_pgtle_$
);
```

pgtle.register_feature

`register_feature` 函数向 `pgtle.feature_info` 表中添加了指定的内部 PostgreSQL 功能。PostgreSQL 挂钩是 PostgreSQL 内部功能的一个示例。可信语言扩展开发套件支持使用 PostgreSQL 挂钩。目前，此函数支持以下功能。

- `passcheck` – 将密码检查挂钩注册到自定义 PostgreSQL 的密码检查行为的过程或函数。

函数原型

```
pgtle.register_feature(proc regproc, feature pg_tle_feature)
```

角色

`pgtle_admin`

参数

- `proc` – 用于该功能的存储过程或函数的名称。
- `feature` – 要注册到该函数的 `pg_tle` 功能（例如 `passcheck`）的名称。

输出

无。

用法示例

```
SELECT pgtle.register_feature('pw_hook', 'passcheck');
```

pgtle.register_feature_if_not_exists

`pgtle.register_feature_if_not_exists` 函数将指定的 PostgreSQL 功能添加到 `pgtle.feature_info` 表中，并标识使用该功能的 TLE 扩展或其他过程或函数。有关挂钩和可信语言扩展的更多信息，请参阅 在您的 TLE 扩展中使用 PostgreSQL 挂钩。

函数原型

```
pgtle.register_feature_if_not_exists(proc regproc, feature pg_tle_feature)
```

角色

`pgtle_admin`

参数

- `proc` – 包含用作 TLE 扩展的功能的逻辑（代码）的存储过程或函数的名称。例如，pw_hook 代码。
- `feature` – 要为 TLE 函数注册的 PostgreSQL 功能的名称。目前，唯一可用的功能是 `passcheck` 挂钩。有关更多信息，请参阅 密码检查挂钩（passcheck）。

输出

为指定的扩展注册该功能后返回 `true`。如果该功能已经注册，则返回 `false`。

用法示例

```
SELECT pgtle.register_feature_if_not_exists('pw_hook', 'passcheck');
```

pgtle.set_default_version

`set_default_version` 函数可让您指定 TLE 扩展的 `default_version`。您可以使用此函数定义升级路径，并将该版本指定为 TLE 扩展的缺省版本。当数据库用户在 `CREATE EXTENSION` 和 `ALTER EXTENSION ... UPDATE` 命令中指定您的 TLE 扩展时，将在数据库中为该用户创建该版本的 TLE 扩展。

成功时，此函数返回 `true`。如果在 name 参数中指定的 TLE 扩展不存在，此函数将返回错误。同样，如果 TLE 扩展的 `version` 不存在，它会返回错误。

函数原型

```
pgtle.set_default_version(name text, version text)
```

角色

pgtle_admin

参数

- name – TLE 扩展的名称。调用 CREATE EXTENSION 时使用此值。
- version – 用于设置缺省设置的 TLE 扩展版本。

输出

- true – 成功设置缺省版本时，此函数返回 true。
- ERROR – 如果具有指定名称或版本的 TLE 扩展不存在，则返回错误消息。

用法示例

```
SELECT * FROM pgtle.set_default_version('my-extension', '1.1');
```

pgtle.uninstall_extension(name)

uninstall_extension 函数从数据库中删除 TLE 扩展的所有版本。此函数可防止 CREATE EXTENSION 的将来调用安装 TLE 扩展。如果数据库中不存在 TLE 扩展，则会引发错误。

uninstall_extension 函数不会删除数据库中当前处于活动状态的 TLE 扩展。要删除当前处于活动状态的 TLE 扩展，您需要显式调用 DROP EXTENSION 以将其删除。

函数原型

```
pgtle.uninstall_extension(extname text)
```

角色

pgtle_admin

参数

- `extname` – 要卸载的 TLE 扩展的名称。此名称与用于 `CREATE EXTENSION` 的名称相同，旨在加载 TLE 扩展以便在给定数据库中使用。

输出

无。

用法示例

```
SELECT * FROM pgtle.uninstall_extension('pg_tle_test');
```

pgtle.uninstall_extension(name, version)

`uninstall_extension(name, version)` 函数从数据库中删除指定版本的 TLE 扩展。此功能可防止 `CREATE EXTENSION` 和 `ALTER EXTENSION` 将 TLE 扩展安装或更新到指定版本。此函数还删除指定版本的 TLE 扩展的所有更新路径。如果 TLE 扩展当前在数据库中处于活动状态，则此函数不会将其卸载。必须显式调用 `DROP EXTENSION` 才能删除 TLE 扩展。要卸载 TLE 扩展的所有版本，请参阅 pgtle.uninstall_extension(name)。

函数原型

```
pgtle.uninstall_extension(extname text, version text)
```

角色

`pgtle_admin`

参数

- `extname` – TLE 扩展的名称。调用 `CREATE EXTENSION` 时使用此值。
- `version` – 要从数据库中卸载的 TLE 扩展的版本。

输出

无。

用法示例

```
SELECT * FROM pgtle.uninstall_extension('pg_tle_test', '0.2');
```

pgtle.uninstall_extension_if_exists

uninstall_extension_if_exists 函数从给定的数据库中删除 TLE 扩展的所有版本。如果 TLE 扩展不存在，则该函数会静默返回（不会引发任何错误消息）。如果指定的扩展当前在数据库中处于活动状态，则此函数不会将其删除。必须先显式调用 DROP EXTENSION 以删除 TLE 扩展，然后才能使用此函数卸载其构件。

函数原型

```
pgtle.uninstall_extension_if_exists(extname text)
```

角色

pgtle_admin

参数

- extname – TLE 扩展的名称。调用 CREATE EXTENSION 时使用此值。

输出

在卸载指定扩展后，uninstall_extension_if_exists 函数返回 true。如果指定的扩展不存在，则此函数将返回 false。

- true – 卸载 TLE 扩展后返回 true。
- false – 当数据库中不存在 TLE 扩展时返回 false。

用法示例

```
SELECT * FROM pgtle.uninstall_extension_if_exists('pg_tle_test');
```

pgtle.uninstall_update_path

uninstall_update_path 函数从 TLE 扩展中删除特定的更新路径。这可以防止 ALTER EXTENSION ... UPDATE TO 将其用作更新路径。

如果 TLE 扩展当前正由此更新路径上的其中一个版本使用，则该扩展保留在数据库中。

如果指定的更新路径不存在，则此函数会引发错误。

函数原型

```
pgtle.uninstall_update_path(extname text, fromvers text, tovers text)
```

角色

pgtle_admin

参数

- extname – TLE 扩展的名称。调用 CREATE EXTENSION 时使用此值。
- fromvers – 更新路径上使用的 TLE 扩展的源版本。
- tovers – 更新路径上使用的 TLE 扩展的目标版本。

输出

无。

用法示例

```
SELECT * FROM pgtle.uninstall_update_path('pg_tle_test', '0.1', '0.2');
```

pgtle.uninstall_update_path_if_exists

uninstall_update_path_if_exists 函数与 uninstall_update_path 类似，它从 TLE 扩展中删除指定的更新路径。但是，如果更新路径不存在，则此函数不会引发错误消息。相反，该函数返回 false。

函数原型

```
pgtle.uninstall_update_path_if_exists(extname text, fromvers text, tovers text)
```

角色

pgtle_admin

参数

- extname – TLE 扩展的名称。调用 CREATE EXTENSION 时使用此值。
- fromvers – 更新路径上使用的 TLE 扩展的源版本。
- tovers – 更新路径上使用的 TLE 扩展的目标版本。

输出

- true – 该函数已成功更新 TLE 扩展的路径。
- false – 该函数无法更新 TLE 扩展的路径。

用法示例

```
SELECT * FROM pgtle.uninstall_update_path_if_exists('pg_tle_test', '0.1', '0.2');
```

pgtle.unregister_feature

unregister_feature 函数提供了一种方法，用于删除注册为使用 pg_tle 功能（如挂钩）的函数。有关注册功能的信息，请参阅 pgtle.register_feature。

函数原型

```
pgtle.unregister_feature(proc regproc, feature pg_tle_features)
```

角色

pgtle_admin

参数

- proc – 要向 pg_tle 功能注册的存储函数的名称。
- feature – 要向函数注册的 pg_tle 功能的名称。例如，passcheck 是一项可以注册以供您开发的可信语言扩展使用的功能。有关更多信息，请参阅 密码检查挂钩（passcheck）。

输出

无。

用法示例

```
SELECT * FROM pgtle.unregister_feature('pw_hook', 'passcheck');
```

pgtle.unregister_feature_if_exists

unregister_feature 函数提供了一种方法，用于删除注册为使用 pg_tle 功能（如挂钩）的函数。有关更多信息，请参阅 在您的 TLE 扩展中使用 PostgreSQL 挂钩。成功取消注册该功能后，返回 true。如果未注册该功能，则返回 false。

有关为 TLE 扩展注册 pg_tle 功能的信息，请参阅 pgtle.register_feature。

函数原型

```
pgtle.unregister_feature_if_exists('proc regproc', 'feature pg_tle_features')
```

角色

pgtle_admin

参数

- proc – 为包含 pg_tle 功能而注册的存储函数的名称。
- feature – 向可信语言扩展注册的 pg_tle 功能的名称。

输出

返回 true 或 false，如下所示。

- true – 该函数已成功将该功能从扩展中取消注册。
- false – 该函数无法从 TLE 扩展中取消注册该功能。

用法示例

```
SELECT * FROM pgtle.unregister_feature_if_exists('pw_hook', 'passcheck');
```

适用于 PostgreSQL 的可信语言扩展的挂钩参考

适用于 PostgreSQL 的可信语言扩展支持 PostgreSQL 挂钩。挂钩是一种内部回调机制，可供开发人员扩展 PostgreSQL 的核心功能。通过使用挂钩，开发人员可以实现自己的函数或过程以在各种数据

库操作中使用，从而以某种方式修改 PostgreSQL 的行为。例如，您可以使用 passcheck 挂钩自定义 PostgreSQL 如何处理在为用户（角色）创建或更改密码时提供的密码。

查看以下文档，了解可用于 TLE 扩展的 passcheck 钩子。要了解有关可用钩子（包括客户端身份验证钩子）的更多信息，请参阅 Trusted Language Extensions hooks。

密码检查挂钩（passcheck）

passcheck 挂钩用于在以下 SQL 命令和 psql 元命令的密码检查过程中自定义 PostgreSQL 行为。

- `CREATE ROLE` *username* `...PASSWORD` – 有关更多信息，请参阅 PostgreSQL 文档中的 CREATE ROLE。
- `ALTER ROLE` *username*`...PASSWORD` – 有关更多信息，请参阅 PostgreSQL 文档中的 ALTER ROLE。
- `\password` *username* – 此交互式 psql 元命令在透明地使用 `ALTER ROLE ... PASSWORD` 语法之前，通过对密码进行哈希处理来安全地更改指定用户的密码。元命令是 `ALTER ROLE ... PASSWORD` 命令的安全包装器，因此挂钩适用于 psql 元命令的行为。

有关示例，请参阅密码检查挂钩代码列表。

目录

函数原型

```
passcheck_hook(username text, password text, password_type pgtle.password_types,
 valid_until timestamptz, valid_null boolean)
```

参数

passcheck 挂钩函数采用以下参数。

- username – 设置密码的角色（用户名）的名称（文本）。

- password – 纯文本或哈希处理的密码。输入的密码应与在 password_type 中指定的类型相匹配。
- password_type – 指定密码的 pgtle.password_type 格式。此格式可能是以下选项之一。
  - PASSWORD_TYPE_PLAINTEXT – 纯文本密码。
  - PASSWORD_TYPE_MD5 – 已使用 MD5（消息摘要 5）算法进行哈希处理的密码。
  - PASSWORD_TYPE_SCRAM_SHA_256 – 已使用 SCRAM-SHA-256 算法进行哈希处理的密码。
- valid_until – 指定密码变为失效的时间。此参数是可选的。如果使用此参数，请将时间指定为 timestamptz 值。
- valid_null – 如果此布尔值设置为 true，则 valid_until 选项设置为 NULL。

配置

函数 pgtle.enable_password_check 控制 passcheck 挂钩是否处于活动状态。passcheck 挂钩有三种可能的设置。

- off – 关闭 passcheck 密码检查挂钩。这是默认值。
- on – 打开 passcode 密码检查挂钩，以便对照表检查密码。
- require – 需要定义密码检查挂钩。

使用说明

要打开或关闭 passcheck 挂钩，您需要修改 RDS for PostgreSQL 数据库实例的自定义数据库参数组。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-parameter-group \
    --region aws-region \
    --db-parameter-group-name your-custom-parameter-group \
    --parameters 
 "ParameterName=pgtle.enable_password_check,ParameterValue=on,ApplyMethod=immediate"
```

对于 Windows：

```
aws rds modify-db-parameter-group ^
    --region aws-region ^
    --db-parameter-group-name your-custom-parameter-group ^
```

```
    --parameters 
"ParameterName=pgtle.enable_password_check,ParameterValue=on,ApplyMethod=immediate"
```

适用于使用 Amazon SDK 的 Amazon RDS 的代码示例

以下代码示例展示如何将 Amazon RDS 与 Amazon 软件开发工具包（SDK）一起使用。

基础知识是向您展示如何在服务中执行基本操作的代码示例。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

场景是向您展示如何通过在一个服务中调用多个函数或与其他 Amazon Web Services 服务 服务结合来完成特定任务的代码示例。

有关 Amazon SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 Amazon SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

开始使用

开始使用 Amazon RDS

以下代码示例演示了如何开始使用 Amazon RDS。

.NET

适用于 .NET 的 Amazon SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
using System;
using System.Threading.Tasks;
using Amazon.RDS;
using Amazon.RDS.Model;

namespace RDSActions;

public static class HelloRds
{
    static async Task Main(string[] args)
    {
```

```
        var rdsClient = new AmazonRDSClient();

        Console.WriteLine($"Hello Amazon RDS! Following are some of your DB
 instances:");
        Console.WriteLine();

        // You can use await and any of the async methods to get a response.
        // Let's get the first twenty DB instances.
        var response = await rdsClient.DescribeDBInstancesAsync(
            new DescribeDBInstancesRequest()
            {
                MaxRecords = 20 // Must be between 20 and 100.
            });

        foreach (var instance in response.DBInstances)
        {
            Console.WriteLine($"\tDB name: {instance.DBName}");
            Console.WriteLine($"\tArn: {instance.DBInstanceArn}");
            Console.WriteLine($"\tIdentifier: {instance.DBInstanceIdentifier}");
            Console.WriteLine();
        }
    }
}
```

- 有关 API 详细信息，请参阅《适用于 .NET 的 Amazon SDK API 参考》中的 DescribeDBInstances。

C++

SDK for C++

Note

查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

CMakeLists.txt CMake 文件的代码。

```
# Set the minimum required version of CMake for this project.
cmake_minimum_required(VERSION 3.13)
```

```
# Set the AWS service components used by this project.
set(SERVICE_COMPONENTS rds)

# Set this project's name.
project("hello_rds")

# Set the C++ standard to use to build this target.
# At least C++ 11 is required for the AWS SDK for C++.
set(CMAKE_CXX_STANDARD 11)

# Use the MSVC variable to determine if this is a Windows build.
set(WINDOWS_BUILD ${MSVC})

if (WINDOWS_BUILD) # Set the location where CMake can find the installed
 libraries for the AWS SDK.
    string(REPLACE ";" "/aws-cpp-sdk-all;" SYSTEM_MODULE_PATH
 "${CMAKE_SYSTEM_PREFIX_PATH}/aws-cpp-sdk-all")
    list(APPEND CMAKE_PREFIX_PATH ${SYSTEM_MODULE_PATH})
endif ()

# Find the AWS SDK for C++ package.
find_package(AWSSDK REQUIRED COMPONENTS ${SERVICE_COMPONENTS})

if (WINDOWS_BUILD AND AWSSDK_INSTALL_AS_SHARED_LIBS)
     # Copy relevant AWS SDK for C++ libraries into the current binary directory
 for running and debugging.

     # set(BIN_SUB_DIR "/Debug") # If you are building from the command line, you
 may need to uncomment this
                                    # and set the proper subdirectory to the
 executables' location.

     AWSSDK_CPY_DYN_LIBS(SERVICE_COMPONENTS ""
 ${CMAKE_CURRENT_BINARY_DIR}${BIN_SUB_DIR})
endif ()

add_executable(${PROJECT_NAME}
        hello_rds.cpp)

target_link_libraries(${PROJECT_NAME}
        ${AWSSDK_LINK_LIBRARIES})
```

hello_rds.cpp 源文件的代码。

```
#include <aws/core/Aws.h>
#include <aws/rds/RDSClient.h>
#include <aws/rds/model/DescribeDBInstancesRequest.h>
#include <iostream>

/*
 *  A "Hello Rds" starter application which initializes an Amazon Relational
 Database Service (Amazon RDS) client and
 *  describes the Amazon RDS instances.
 *
 *  main function
 *
 *  Usage: 'hello_rds'
 *
 */

int main(int argc, char **argv) {
    Aws::SDKOptions options;
    // Optionally change the log level for debugging.
//   options.loggingOptions.logLevel = Utils::Logging::LogLevel::Debug;
    Aws::InitAPI(options); // Should only be called once.
    int result = 0;
    {
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

        Aws::RDS::RDSClient rdsClient(clientConfig);
        Aws::String marker;
        std::vector<Aws::String> instanceDBIDs;

        do {
            Aws::RDS::Model::DescribeDBInstancesRequest request;

            if (!marker.empty()) {
                request.SetMarker(marker);
            }

            Aws::RDS::Model::DescribeDBInstancesOutcome outcome =
                    rdsClient.DescribeDBInstances(request);

            if (outcome.IsSuccess()) {
```

```
                for (auto &instance: outcome.GetResult().GetDBInstances()) {
                    instanceDBIDs.push_back(instance.GetDBInstanceIdentifier());
                }
                marker = outcome.GetResult().GetMarker();
            } else {
                result = 1;
                std::cerr << "Error with RDS::DescribeDBInstances. "
                          << outcome.GetError().GetMessage()
                          << std::endl;
                break;
            }
        } while (!marker.empty());

        std::cout << instanceDBIDs.size() << " RDS instances found." <<
 std::endl;
        for (auto &instanceDBID: instanceDBIDs) {
            std::cout << "   Instance: " << instanceDBID << std::endl;
        }
    }

    Aws::ShutdownAPI(options); // Should only be called once.
    return result;
}
```

- 有关 API 详细信息，请参阅《适用于 C++ 的 Amazon SDK API 参考》中的 DescribeDBInstances。

Go

适用于 Go V2 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
package main
```

```
import (
 "context"
 "fmt"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/service/rds"
)

// main uses the AWS SDK for Go V2 to create an Amazon Relational Database
 Service (Amazon RDS)
// client and list up to 20 DB instances in your account.
// This example uses the default settings specified in your shared credentials
// and config files.
func main() {
 ctx := context.Background()
 sdkConfig, err := config.LoadDefaultConfig(ctx)
 if err != nil {
  fmt.Println("Couldn't load default configuration. Have you set up your AWS
 account?")
  fmt.Println(err)
  return
 }
 rdsClient := rds.NewFromConfig(sdkConfig)
 const maxInstances = 20
 fmt.Printf("Let's list up to %v DB instances.\n", maxInstances)
 output, err := rdsClient.DescribeDBInstances(ctx,
  &rds.DescribeDBInstancesInput{MaxRecords: aws.Int32(maxInstances)})
 if err != nil {
  fmt.Printf("Couldn't list DB instances: %v\n", err)
  return
 }
 if len(output.DBInstances) == 0 {
  fmt.Println("No DB instances found.")
 } else {
  for _, instance := range output.DBInstances {
   fmt.Printf("DB instance %v has database %v.\n",
 *instance.DBInstanceIdentifier,
    *instance.DBName)
  }
 }
}
```

- 有关 API 详细信息，请参阅《适用于 Go 的 Amazon SDK API 参考》中的 DescribeDBInstances。

Java

适用于 Java 的 SDK 2.x

Note

查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.rds.RdsClient;
import software.amazon.awssdk.services.rds.model.DescribeDbInstancesResponse;
import software.amazon.awssdk.services.rds.model.DBInstance;
import software.amazon.awssdk.services.rds.model.RdsException;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DescribeDBInstances {

    public static void main(String[] args) {
        Region region = Region.US_EAST_1;
        RdsClient rdsClient = RdsClient.builder()
                .region(region)
                .build();

        describeInstances(rdsClient);
        rdsClient.close();
```

```
    }

    public static void describeInstances(RdsClient rdsClient) {
        try {
            DescribeDbInstancesResponse response =
 rdsClient.describeDBInstances();
            List<DBInstance> instanceList = response.dbInstances();
            for (DBInstance instance : instanceList) {
                System.out.println("Instance ARN is: " +
 instance.dbInstanceArn());
                System.out.println("The Engine is " + instance.engine());
                System.out.println("Connection endpoint is" +
 instance.endpoint().address());
            }

        } catch (RdsException e) {
            System.out.println(e.getLocalizedMessage());
            System.exit(1);
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon SDK for Java 2.x API 参考》中的 DescribeDBInstances。

Python

适用于 Python 的 SDK（Boto3）

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
"""
Purpose

Shows how to use the AWS SDK for Python (Boto3) with the Amazon Relational
 Database Service
```

```
(Amazon RDS) to list the databases in your account.
"""

import boto3
from botocore.exceptions import ClientError

# Create an RDS client
rds_client = boto3.client("rds")

# Create a paginator for the describe_db_instances operation
paginator = rds_client.get_paginator("describe_db_instances")

try:
    # Use the paginator to get a list of DB instances
    response_iterator = paginator.paginate(
        PaginationConfig={
            "MaxItems": 123,
            "PageSize": 50,  # Adjust PageSize as needed
            "StartingToken": None,
        }
    )

    # Iterate through the pages of the response
    instances_found = False
    for page in response_iterator:
        if "DBInstances" in page and page["DBInstances"]:
            instances_found = True
            print("Your RDS instances are:")
            for db in page["DBInstances"]:
                print(db["DBInstanceIdentifier"])

    if not instances_found:
        print("No RDS instances found!")

except ClientError as e:
    print(f"Couldn't list RDS instances. Here's why: {e.response['Error']
['Message']}")
```

- 有关 API 详细信息，请参阅《适用于 Python 的 Amazon SDK（Boto3）API 参考》中的 DescribeDBInstances。

Ruby

适用于 Ruby 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 [Amazon 代码示例存储库](https://github.com/awsdocs/aws-doc-sdk-examples)中查找完整示例，了解如何进行设置和运行。

```
require 'aws-sdk-rds'
require 'logger'

# RDSManager is a class responsible for managing RDS operations
# such as listing all RDS DB instances in the current AWS account.
class RDSManager
  def initialize(client)
    @client = client
    @logger = Logger.new($stdout)
  end

  # Lists and prints all RDS DB instances in the current AWS account.
  def list_db_instances
    @logger.info('Listing RDS DB instances')

    paginator = @client.describe_db_instances
    instances = []

    paginator.each_page do |page|
      instances.concat(page.db_instances)
    end

    if instances.empty?
      @logger.info('No instances found.')
    else
      @logger.info("Found #{instances.count} instance(s):")
      instances.each do |instance|
        @logger.info(" * #{instance.db_instance_identifier}
 (#{instance.db_instance_status})")
      end
    end
```

```
  end
end

if $PROGRAM_NAME == __FILE__
  rds_client = Aws::RDS::Client.new(region: 'us-west-2')
  manager = RDSManager.new(rds_client)
  manager.list_db_instances
end
```

- 有关 API 详细信息，请参阅《适用于 Ruby 的 Amazon SDK API 参考》中的 DescribeDBInstances。

代码示例

- 使用 Amazon SDK 的 Amazon RDS 的基本示例
  - 开始使用 Amazon RDS
  - 了解使用 Amazon SDK 的 Amazon RDS 的基本功能
  - 适用于使用 Amazon SDK 的 Amazon RDS 的操作
    - 将 CreateDBInstance 与 Amazon SDK 或 CLI 配合使用
    - 将 CreateDBParameterGroup 与 Amazon SDK 或 CLI 配合使用
    - 将 CreateDBSnapshot 与 Amazon SDK 或 CLI 配合使用
    - 将 DeleteDBInstance 与 Amazon SDK 或 CLI 配合使用
    - 将 DeleteDBParameterGroup 与 Amazon SDK 或 CLI 配合使用
    - 将 DescribeAccountAttributes 与 Amazon SDK 或 CLI 配合使用
    - 将 DescribeDBEngineVersions 与 Amazon SDK 或 CLI 配合使用
    - 将 DescribeDBInstances 与 Amazon SDK 或 CLI 配合使用
    - 将 DescribeDBParameterGroups 与 Amazon SDK 或 CLI 配合使用
    - 将 DescribeDBParameters 与 Amazon SDK 或 CLI 配合使用
    - 将 DescribeDBSnapshots 与 Amazon SDK 或 CLI 配合使用
    - 将 DescribeOrderableDBInstanceOptions 与 Amazon SDK 或 CLI 配合使用
    - 将 GenerateRDSAuthToken 和 Amazon SDK 搭配使用
    - 将 ModifyDBInstance 与 Amazon SDK 或 CLI 配合使用
    - 将 ModifyDBParameterGroup 与 Amazon SDK 或 CLI 配合使用

使用 Amazon SDK 的 Amazon RDS 的基本示例

以下代码示例演示了如何将 Amazon Relational Database Service 的基本功能与 Amazon SDK 结合使用。

示例

开始使用 Amazon RDS

以下代码示例演示了如何开始使用 Amazon RDS。

.NET

适用于 .NET 的 Amazon SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 [Amazon 代码示例存储库](#)中查找完整示例，了解如何进行设置和运行。

```
using System;
using System.Threading.Tasks;
using Amazon.RDS;
using Amazon.RDS.Model;

namespace RDSActions;

public static class HelloRds
{
    static async Task Main(string[] args)
    {
        var rdsClient = new AmazonRDSClient();

        Console.WriteLine($"Hello Amazon RDS! Following are some of your DB
 instances:");
        Console.WriteLine();

        // You can use await and any of the async methods to get a response.
        // Let's get the first twenty DB instances.
        var response = await rdsClient.DescribeDBInstancesAsync(
            new DescribeDBInstancesRequest()
            {
                MaxRecords = 20 // Must be between 20 and 100.
            });

        foreach (var instance in response.DBInstances)
        {
            Console.WriteLine($"\tDB name: {instance.DBName}");
```

```
                Console.WriteLine($"\tArn: {instance.DBInstanceArn}");
                Console.WriteLine($"\tIdentifier: {instance.DBInstanceIdentifier}");
                Console.WriteLine();
            }
        }
}
```

- 有关 API 详细信息，请参阅《适用于 .NET 的 Amazon SDK API 参考》中的 DescribeDBInstances。

C++

SDK for C++

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

CMakeLists.txt CMake 文件的代码。

```
# Set the minimum required version of CMake for this project.
cmake_minimum_required(VERSION 3.13)

# Set the AWS service components used by this project.
set(SERVICE_COMPONENTS rds)

# Set this project's name.
project("hello_rds")

# Set the C++ standard to use to build this target.
# At least C++ 11 is required for the AWS SDK for C++.
set(CMAKE_CXX_STANDARD 11)

# Use the MSVC variable to determine if this is a Windows build.
set(WINDOWS_BUILD ${MSVC})

if (WINDOWS_BUILD) # Set the location where CMake can find the installed
 libraries for the AWS SDK.
```

```
    string(REPLACE ";" "/aws-cpp-sdk-all;" SYSTEM_MODULE_PATH
 "${CMAKE_SYSTEM_PREFIX_PATH}/aws-cpp-sdk-all")
    list(APPEND CMAKE_PREFIX_PATH ${SYSTEM_MODULE_PATH})
endif ()

# Find the AWS SDK for C++ package.
find_package(AWSSDK REQUIRED COMPONENTS ${SERVICE_COMPONENTS})

if (WINDOWS_BUILD AND AWSSDK_INSTALL_AS_SHARED_LIBS)
     # Copy relevant AWS SDK for C++ libraries into the current binary directory
 for running and debugging.

     # set(BIN_SUB_DIR "/Debug") # If you are building from the command line, you
 may need to uncomment this
                                    # and set the proper subdirectory to the
 executables' location.

     AWSSDK_CPY_DYN_LIBS(SERVICE_COMPONENTS ""
 ${CMAKE_CURRENT_BINARY_DIR}${BIN_SUB_DIR})
endif ()

add_executable(${PROJECT_NAME}
        hello_rds.cpp)

target_link_libraries(${PROJECT_NAME}
        ${AWSSDK_LINK_LIBRARIES})
```

hello_rds.cpp 源文件的代码。

```
#include <aws/core/Aws.h>
#include <aws/rds/RDSClient.h>
#include <aws/rds/model/DescribeDBInstancesRequest.h>
#include <iostream>

/*
 *  A "Hello Rds" starter application which initializes an Amazon Relational
 Database Service (Amazon RDS) client and
 *  describes the Amazon RDS instances.
 *
 *  main function
 *
 *  Usage: 'hello_rds'
```

```
 *
 */

int main(int argc, char **argv) {
    Aws::SDKOptions options;
    // Optionally change the log level for debugging.
//   options.loggingOptions.logLevel = Utils::Logging::LogLevel::Debug;
    Aws::InitAPI(options); // Should only be called once.
    int result = 0;
    {
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

        Aws::RDS::RDSClient rdsClient(clientConfig);
        Aws::String marker;
        std::vector<Aws::String> instanceDBIDs;

        do {
            Aws::RDS::Model::DescribeDBInstancesRequest request;

            if (!marker.empty()) {
                request.SetMarker(marker);
            }

            Aws::RDS::Model::DescribeDBInstancesOutcome outcome =
                    rdsClient.DescribeDBInstances(request);

            if (outcome.IsSuccess()) {
                for (auto &instance: outcome.GetResult().GetDBInstances()) {
                    instanceDBIDs.push_back(instance.GetDBInstanceIdentifier());
                }
                marker = outcome.GetResult().GetMarker();
            } else {
                result = 1;
                std::cerr << "Error with RDS::DescribeDBInstances. "
                          << outcome.GetError().GetMessage()
                          << std::endl;
                break;
            }
        } while (!marker.empty());

        std::cout << instanceDBIDs.size() << " RDS instances found." <<
 std::endl;
```

```
        for (auto &instanceDBID: instanceDBIDs) {
            std::cout << "   Instance: " << instanceDBID << std::endl;
        }
    }

    Aws::ShutdownAPI(options); // Should only be called once.
    return result;
}
```

- 有关 API 详细信息，请参阅《适用于 C++ 的 Amazon SDK API 参考》中的 DescribeDBInstances。

Go

适用于 Go V2 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
package main

import (
 "context"
 "fmt"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/service/rds"
)

// main uses the AWS SDK for Go V2 to create an Amazon Relational Database
 Service (Amazon RDS)
// client and list up to 20 DB instances in your account.
// This example uses the default settings specified in your shared credentials
// and config files.
func main() {
```

```
 ctx := context.Background()
 sdkConfig, err := config.LoadDefaultConfig(ctx)
 if err != nil {
  fmt.Println("Couldn't load default configuration. Have you set up your AWS
 account?")
  fmt.Println(err)
  return
 }
 rdsClient := rds.NewFromConfig(sdkConfig)
 const maxInstances = 20
 fmt.Printf("Let's list up to %v DB instances.\n", maxInstances)
 output, err := rdsClient.DescribeDBInstances(ctx,
  &rds.DescribeDBInstancesInput{MaxRecords: aws.Int32(maxInstances)})
 if err != nil {
  fmt.Printf("Couldn't list DB instances: %v\n", err)
  return
 }
 if len(output.DBInstances) == 0 {
  fmt.Println("No DB instances found.")
 } else {
  for _, instance := range output.DBInstances {
   fmt.Printf("DB instance %v has database %v.\n",
 *instance.DBInstanceIdentifier,
    *instance.DBName)
  }
 }
}
```

- 有关 API 详细信息，请参阅《适用于 Go 的 Amazon SDK API 参考》中的 [DescribeDBInstances](#)。

Java

适用于 Java 的 SDK 2.x

> **Note**
>
> 查看 GitHub，了解更多信息。在 [Amazon 代码示例存储库](#)中查找完整示例，了解如何进行设置和运行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.rds.RdsClient;
import software.amazon.awssdk.services.rds.model.DescribeDbInstancesResponse;
import software.amazon.awssdk.services.rds.model.DBInstance;
import software.amazon.awssdk.services.rds.model.RdsException;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DescribeDBInstances {

    public static void main(String[] args) {
        Region region = Region.US_EAST_1;
        RdsClient rdsClient = RdsClient.builder()
                .region(region)
                .build();

        describeInstances(rdsClient);
        rdsClient.close();
    }

    public static void describeInstances(RdsClient rdsClient) {
        try {
            DescribeDbInstancesResponse response =
 rdsClient.describeDBInstances();
            List<DBInstance> instanceList = response.dbInstances();
            for (DBInstance instance : instanceList) {
                System.out.println("Instance ARN is: " +
 instance.dbInstanceArn());
                System.out.println("The Engine is " + instance.engine());
                System.out.println("Connection endpoint is" +
 instance.endpoint().address());
            }

        } catch (RdsException e) {
            System.out.println(e.getLocalizedMessage());
```

```
                System.exit(1);
            }
        }
    }
```

- 有关 API 详细信息，请参阅《Amazon SDK for Java 2.x API 参考》中的 DescribeDBInstances。

Python

适用于 Python 的 SDK（Boto3）

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
"""
Purpose

Shows how to use the AWS SDK for Python (Boto3) with the Amazon Relational
 Database Service
(Amazon RDS) to list the databases in your account.
"""

import boto3
from botocore.exceptions import ClientError

# Create an RDS client
rds_client = boto3.client("rds")

# Create a paginator for the describe_db_instances operation
paginator = rds_client.get_paginator("describe_db_instances")

try:
    # Use the paginator to get a list of DB instances
    response_iterator = paginator.paginate(
        PaginationConfig={
            "MaxItems": 123,
```

```
            "PageSize": 50,  # Adjust PageSize as needed
            "StartingToken": None,
        }
    )

    # Iterate through the pages of the response
    instances_found = False
    for page in response_iterator:
        if "DBInstances" in page and page["DBInstances"]:
            instances_found = True
            print("Your RDS instances are:")
            for db in page["DBInstances"]:
                print(db["DBInstanceIdentifier"])

    if not instances_found:
        print("No RDS instances found!")

except ClientError as e:
    print(f"Couldn't list RDS instances. Here's why: {e.response['Error']
['Message']}")
```

- 有关 API 详细信息，请参阅《适用于 Python 的 Amazon SDK（Boto3）API 参考》中的 DescribeDBInstances。

Ruby

适用于 Ruby 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
require 'aws-sdk-rds'
require 'logger'

# RDSManager is a class responsible for managing RDS operations
```

```
# such as listing all RDS DB instances in the current AWS account.
class RDSManager
  def initialize(client)
    @client = client
    @logger = Logger.new($stdout)
  end

  # Lists and prints all RDS DB instances in the current AWS account.
  def list_db_instances
    @logger.info('Listing RDS DB instances')

    paginator = @client.describe_db_instances
    instances = []

    paginator.each_page do |page|
      instances.concat(page.db_instances)
    end

    if instances.empty?
      @logger.info('No instances found.')
    else
      @logger.info("Found #{instances.count} instance(s):")
      instances.each do |instance|
        @logger.info(" * #{instance.db_instance_identifier}
 (#{instance.db_instance_status})")
      end
    end
  end
end

if $PROGRAM_NAME == __FILE__
  rds_client = Aws::RDS::Client.new(region: 'us-west-2')
  manager = RDSManager.new(rds_client)
  manager.list_db_instances
end
```

- 有关 API 详细信息，请参阅《适用于 Ruby 的 Amazon SDK API 参考》中的 DescribeDBInstances。

有关 Amazon SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 Amazon SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

了解使用 Amazon SDK 的 Amazon RDS 的基本功能

以下代码示例演示了如何：

- 创建自定义数据库参数组并设置参数值。
- 创建一个配置为使用参数组的数据库实例。数据库实例还包含一个数据库。
- 拍摄实例的快照。
- 删除实例和参数组。

.NET

适用于 .NET 的 Amazon SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

在命令提示符中运行交互式场景。

```
/// <summary>
/// Scenario for RDS DB instance example.
/// </summary>
public class RDSInstanceScenario
{
    /*
    Before running this .NET code example, set up your development environment,
 including your credentials.

    This .NET example performs the following tasks:
    1.  Returns a list of the available DB engine families using the
 DescribeDBEngineVersionsAsync method.
    2.  Selects an engine family and creates a custom DB parameter group using
 the CreateDBParameterGroupAsync method.
```

```
    3.  Gets the parameter groups using the DescribeDBParameterGroupsAsync
method.
    4.  Gets parameters in the group using the DescribeDBParameters method.
    5.  Parses and displays parameters in the group.
    6.  Modifies both the auto_increment_offset and auto_increment_increment
parameters
        using the ModifyDBParameterGroupAsync method.
    7.  Gets and displays the updated parameters using the DescribeDBParameters
method with a source of "user".
    8.  Gets a list of allowed engine versions using the
DescribeDBEngineVersionsAsync method.
    9.  Displays and selects from a list of micro instance classes available for
the selected engine and version.
    10. Creates an RDS DB instance that contains a MySql database and uses the
parameter group
        using the CreateDBInstanceAsync method.
    11. Waits for DB instance to be ready using the DescribeDBInstancesAsync
method.
    12. Prints out the connection endpoint string for the new DB instance.
    13. Creates a snapshot of the DB instance using the CreateDBSnapshotAsync
method.
    14. Waits for DB snapshot to be ready using the DescribeDBSnapshots method.
    15. Deletes the DB instance using the DeleteDBInstanceAsync method.
    16. Waits for DB instance to be deleted using the DescribeDbInstances method.
    17. Deletes the parameter group using the DeleteDBParameterGroupAsync.
    */

    private static readonly string sepBar = new('-', 80);
    private static RDSWrapper rdsWrapper = null!;
    private static ILogger logger = null!;
    private static readonly string engine = "mysql";
    static async Task Main(string[] args)
    {
        // Set up dependency injection for the Amazon RDS service.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
LogLevel.Trace))
            .ConfigureServices((_, services) =>
                services.AddAWSService<IAmazonRDS>()
                    .AddTransient<RDSWrapper>()
```

```
            )
            .Build();

        logger = LoggerFactory.Create(builder =>
        {
            builder.AddConsole();
        }).CreateLogger<RDSInstanceScenario>();

        rdsWrapper = host.Services.GetRequiredService<RDSWrapper>();

        Console.WriteLine(sepBar);
        Console.WriteLine(
            "Welcome to the Amazon Relational Database Service (Amazon RDS) DB
 instance scenario example.");
        Console.WriteLine(sepBar);

        try
        {
            var parameterGroupFamily = await ChooseParameterGroupFamily();

            var parameterGroup = await
 CreateDbParameterGroup(parameterGroupFamily);

            var parameters = await
 DescribeParametersInGroup(parameterGroup.DBParameterGroupName,
                new List<string> { "auto_increment_offset",
 "auto_increment_increment" });

            await ModifyParameters(parameterGroup.DBParameterGroupName,
 parameters);

            await
 DescribeUserSourceParameters(parameterGroup.DBParameterGroupName);

            var engineVersionChoice = await
 ChooseDbEngineVersion(parameterGroupFamily);

            var instanceChoice = await ChooseDbInstanceClass(engine,
 engineVersionChoice.EngineVersion);

            var newInstanceIdentifier = "Example-Instance-" + DateTime.Now.Ticks;

            var newInstance = await CreateRdsNewInstance(parameterGroup, engine,
 engineVersionChoice.EngineVersion,
```

```
                instanceChoice.DBInstanceClass, newInstanceIdentifier);
            if (newInstance != null)
            {
                DisplayConnectionString(newInstance);

                await CreateSnapshot(newInstance);

                await DeleteRdsInstance(newInstance);
            }

            await DeleteParameterGroup(parameterGroup);

            Console.WriteLine("Scenario complete.");
            Console.WriteLine(sepBar);
        }
        catch (Exception ex)
        {
            logger.LogError(ex, "There was a problem executing the scenario.");
        }
    }

    /// <summary>
    /// Choose the RDS DB parameter group family from a list of available
 options.
    /// </summary>
    /// <returns>The selected parameter group family.</returns>
    public static async Task<string> ChooseParameterGroupFamily()
    {
        Console.WriteLine(sepBar);
        // 1. Get a list of available engines.
        var engines = await rdsWrapper.DescribeDBEngineVersions(engine);

        Console.WriteLine("1. The following is a list of available DB parameter
 group families:");
        int i = 1;
        var parameterGroupFamilies = engines.GroupBy(e =>
 e.DBParameterGroupFamily).ToList();
        foreach (var parameterGroupFamily in parameterGroupFamilies)
        {
            // List the available parameter group families.
            Console.WriteLine(
                $"\t{i}. Family: {parameterGroupFamily.Key}");
            i++;
        }
```

```
        var choiceNumber = 0;
        while (choiceNumber < 1 || choiceNumber > parameterGroupFamilies.Count)
        {
            Console.WriteLine("Select an available DB parameter group family by
 entering a number from the list above:");
            var choice = Console.ReadLine();
            Int32.TryParse(choice, out choiceNumber);
        }
        var parameterGroupFamilyChoice = parameterGroupFamilies[choiceNumber -
 1];
        Console.WriteLine(sepBar);
        return parameterGroupFamilyChoice.Key;
    }

    /// <summary>
    /// Create and get information on a DB parameter group.
    /// </summary>
    /// <param name="dbParameterGroupFamily">The DBParameterGroupFamily for the
 new DB parameter group.</param>
    /// <returns>The new DBParameterGroup.</returns>
    public static async Task<DBParameterGroup> CreateDbParameterGroup(string
 dbParameterGroupFamily)
    {
        Console.WriteLine(sepBar);
        Console.WriteLine($"2. Create new DB parameter group with family
 {dbParameterGroupFamily}:");

        var parameterGroup = await rdsWrapper.CreateDBParameterGroup(
            "ExampleParameterGroup-" + DateTime.Now.Ticks,
            dbParameterGroupFamily, "New example parameter group");

        var groupInfo =
            await rdsWrapper.DescribeDBParameterGroups(parameterGroup
                .DBParameterGroupName);

        Console.WriteLine(
            $"3. New DB parameter group: \n\t{groupInfo[0].Description}, \n\tARN
 {groupInfo[0].DBParameterGroupArn}");
        Console.WriteLine(sepBar);
        return parameterGroup;
    }

    /// <summary>
```

```
    /// Get and describe parameters from a DBParameterGroup.
    /// </summary>
    /// <param name="parameterGroupName">Name of the DBParameterGroup.</param>
    /// <param name="parameterNames">Optional specific names of parameters to
 describe.</param>
    /// <returns>The list of requested parameters.</returns>
    public static async Task<List<Parameter>> DescribeParametersInGroup(string
 parameterGroupName, List<string>? parameterNames = null)
    {
        Console.WriteLine(sepBar);
        Console.WriteLine("4. Get some parameters from the group.");
        Console.WriteLine(sepBar);

        var parameters =
            await rdsWrapper.DescribeDBParameters(parameterGroupName);

        var matchingParameters =
            parameters.Where(p => parameterNames == null ||
 parameterNames.Contains(p.ParameterName)).ToList();

        Console.WriteLine("5. Parameter information:");
        matchingParameters.ForEach(p =>
            Console.WriteLine(
                $"\n\tParameter: {p.ParameterName}." +
                $"\n\tDescription: {p.Description}." +
                $"\n\tAllowed Values: {p.AllowedValues}." +
                $"\n\tValue: {p.ParameterValue}."));

        Console.WriteLine(sepBar);

        return matchingParameters;
    }

    /// <summary>
    /// Modify a parameter from a DBParameterGroup.
    /// </summary>
    /// <param name="parameterGroupName">Name of the DBParameterGroup.</param>
    /// <param name="parameters">The parameters to modify.</param>
    /// <returns>Async task.</returns>
    public static async Task ModifyParameters(string parameterGroupName,
 List<Parameter> parameters)
    {
        Console.WriteLine(sepBar);
        Console.WriteLine("6. Modify some parameters in the group.");
```

```
        foreach (var p in parameters)
        {
            if (p.IsModifiable && p.DataType == "integer")
            {
                int newValue = 0;
                while (newValue == 0)
                {
                    Console.WriteLine(
                        $"Enter a new value for {p.ParameterName} from the
 allowed values {p.AllowedValues} ");

                    var choice = Console.ReadLine();
                    Int32.TryParse(choice, out newValue);
                }

                p.ParameterValue = newValue.ToString();
            }
        }

        await rdsWrapper.ModifyDBParameterGroup(parameterGroupName, parameters);

        Console.WriteLine(sepBar);
    }

    /// <summary>
    /// Describe the user source parameters in the group.
    /// </summary>
    /// <param name="parameterGroupName">Name of the DBParameterGroup.</param>
    /// <returns>Async task.</returns>
    public static async Task DescribeUserSourceParameters(string
 parameterGroupName)
    {
        Console.WriteLine(sepBar);
        Console.WriteLine("7. Describe user source parameters in the group.");

        var parameters =
            await rdsWrapper.DescribeDBParameters(parameterGroupName, "user");


        parameters.ForEach(p =>
            Console.WriteLine(
                $"\n\tParameter: {p.ParameterName}." +
                $"\n\tDescription: {p.Description}." +
```

```
                $"\n\tAllowed Values: {p.AllowedValues}." +
                $"\n\tValue: {p.ParameterValue}."));

        Console.WriteLine(sepBar);
    }



    /// <summary>
    /// Choose a DB engine version.
    /// </summary>
    /// <param name="dbParameterGroupFamily">DB parameter group family for engine
 choice.</param>
    /// <returns>The selected engine version.</returns>
    public static async Task<DBEngineVersion> ChooseDbEngineVersion(string
 dbParameterGroupFamily)
    {
        Console.WriteLine(sepBar);
        // Get a list of allowed engines.
        var allowedEngines =
            await rdsWrapper.DescribeDBEngineVersions(engine,
 dbParameterGroupFamily);

        Console.WriteLine($"Available DB engine versions for parameter group
 family {dbParameterGroupFamily}:");
        int i = 1;
        foreach (var version in allowedEngines)
        {
            Console.WriteLine(
                $"\t{i}. Engine: {version.Engine} Version
 {version.EngineVersion}.");
            i++;
        }

        var choiceNumber = 0;
        while (choiceNumber < 1 || choiceNumber > allowedEngines.Count)
        {
            Console.WriteLine("8. Select an available DB engine version by
 entering a number from the list above:");
            var choice = Console.ReadLine();
            Int32.TryParse(choice, out choiceNumber);
        }

        var engineChoice = allowedEngines[choiceNumber - 1];
        Console.WriteLine(sepBar);
```

```
        return engineChoice;
    }

    /// <summary>
    /// Choose a DB instance class for a particular engine and engine version.
    /// </summary>
    /// <param name="engine">DB engine for DB instance choice.</param>
    /// <param name="engineVersion">DB engine version for DB instance choice.</
param>
    /// <returns>The selected orderable DB instance option.</returns>
    public static async Task<OrderableDBInstanceOption>
 ChooseDbInstanceClass(string engine, string engineVersion)
    {
        Console.WriteLine(sepBar);
        // Get a list of allowed DB instance classes.
        var allowedInstances =
            await rdsWrapper.DescribeOrderableDBInstanceOptions(engine,
 engineVersion);

        Console.WriteLine($"8. Available micro DB instance classes for engine
 {engine} and version {engineVersion}:");
        int i = 1;

        // Filter to micro instances for this example.
        allowedInstances = allowedInstances
            .Where(i => i.DBInstanceClass.Contains("micro")).ToList();

        foreach (var instance in allowedInstances)
        {
            Console.WriteLine(
                $"\t{i}. Instance class: {instance.DBInstanceClass} (storage type
 {instance.StorageType})");
            i++;
        }

        var choiceNumber = 0;
        while (choiceNumber < 1 || choiceNumber > allowedInstances.Count)
        {
            Console.WriteLine("9. Select an available DB instance class by
 entering a number from the list above:");
            var choice = Console.ReadLine();
            Int32.TryParse(choice, out choiceNumber);
        }
```

```
        var instanceChoice = allowedInstances[choiceNumber - 1];
        Console.WriteLine(sepBar);
        return instanceChoice;
    }

    /// <summary>
    /// Create a new RDS DB instance.
    /// </summary>
    /// <param name="parameterGroup">Parameter group to use for the DB
 instance.</param>
    /// <param name="engineName">Engine to use for the DB instance.</param>
    /// <param name="engineVersion">Engine version to use for the DB instance.</
param>
    /// <param name="instanceClass">Instance class to use for the DB instance.</
param>
    /// <param name="instanceIdentifier">Instance identifier to use for the DB
 instance.</param>
    /// <returns>The new DB instance.</returns>
    public static async Task<DBInstance?> CreateRdsNewInstance(DBParameterGroup
 parameterGroup,
        string engineName, string engineVersion, string instanceClass, string
 instanceIdentifier)
    {
        Console.WriteLine(sepBar);
        Console.WriteLine($"10. Create a new DB instance with identifier
 {instanceIdentifier}.");
        bool isInstanceReady = false;
        DBInstance newInstance;
        var instances = await rdsWrapper.DescribeDBInstances();
        isInstanceReady = instances.FirstOrDefault(i =>
            i.DBInstanceIdentifier == instanceIdentifier)?.DBInstanceStatus ==
 "available";

        if (isInstanceReady)
        {
            Console.WriteLine("Instance already created.");
            newInstance = instances.First(i => i.DBInstanceIdentifier ==
 instanceIdentifier);
        }
        else
        {
            Console.WriteLine("Please enter an admin user name:");
            var username = Console.ReadLine();
```

```
            Console.WriteLine("Please enter an admin password:");
            var password = Console.ReadLine();

            newInstance = await rdsWrapper.CreateDBInstance(
                "ExampleInstance",
                instanceIdentifier,
                parameterGroup.DBParameterGroupName,
                engineName,
                engineVersion,
                instanceClass,
                20,
                username,
                password
            );

            // 11. Wait for the DB instance to be ready.

            Console.WriteLine("11. Waiting for DB instance to be ready...");
            while (!isInstanceReady)
            {
                instances = await
rdsWrapper.DescribeDBInstances(instanceIdentifier);
                isInstanceReady = instances.FirstOrDefault()?.DBInstanceStatus ==
"available";
                newInstance = instances.First();
                Thread.Sleep(30000);
            }
        }

        Console.WriteLine(sepBar);
        return newInstance;
    }

    /// <summary>
    /// Display a connection string for an RDS DB instance.
    /// </summary>
    /// <param name="instance">The DB instance to use to get a connection
string.</param>
    public static void DisplayConnectionString(DBInstance instance)
    {
        Console.WriteLine(sepBar);
        // Display the connection string.
        Console.WriteLine("12. New DB instance connection string: ");
        Console.WriteLine(
```

```
            $"\n{engine} -h {instance.Endpoint.Address} -P 
 {instance.Endpoint.Port} "
            + $"-u {instance.MasterUsername} -p [YOUR PASSWORD]\n");

        Console.WriteLine(sepBar);
    }

    /// <summary>
    /// Create a snapshot from an RDS DB instance.
    /// </summary>
    /// <param name="instance">DB instance to use when creating a snapshot.</
param>
    /// <returns>The snapshot object.</returns>
    public static async Task<DBSnapshot> CreateSnapshot(DBInstance instance)
    {
        Console.WriteLine(sepBar);
        // Create a snapshot.
        Console.WriteLine($"13. Creating snapshot from DB instance 
 {instance.DBInstanceIdentifier}.");
        var snapshot = await 
 rdsWrapper.CreateDBSnapshot(instance.DBInstanceIdentifier, "ExampleSnapshot-" + 
 DateTime.Now.Ticks);

        // Wait for the snapshot to be available
        bool isSnapshotReady = false;

        Console.WriteLine($"14. Waiting for snapshot to be ready...");
        while (!isSnapshotReady)
        {
            var snapshots = await 
 rdsWrapper.DescribeDBSnapshots(instance.DBInstanceIdentifier);
            isSnapshotReady = snapshots.FirstOrDefault()?.Status == "available";
            snapshot = snapshots.First();
            Thread.Sleep(30000);
        }

        Console.WriteLine(
            $"Snapshot {snapshot.DBSnapshotIdentifier} status is 
 {snapshot.Status}.");
        Console.WriteLine(sepBar);
        return snapshot;
    }

    /// <summary>
```

```
    /// Delete an RDS DB instance.
    /// </summary>
    /// <param name="instance">The DB instance to delete.</param>
    /// <returns>Async task.</returns>
    public static async Task DeleteRdsInstance(DBInstance newInstance)
    {
        Console.WriteLine(sepBar);
        // Delete the DB instance.
        Console.WriteLine($"15. Delete the DB instance
 {newInstance.DBInstanceIdentifier}.");
        await rdsWrapper.DeleteDBInstance(newInstance.DBInstanceIdentifier);

        // Wait for the DB instance to delete.
        Console.WriteLine($"16. Waiting for the DB instance to delete...");
        bool isInstanceDeleted = false;

        while (!isInstanceDeleted)
        {
            var instance = await rdsWrapper.DescribeDBInstances();
            isInstanceDeleted = instance.All(i => i.DBInstanceIdentifier !=
 newInstance.DBInstanceIdentifier);
            Thread.Sleep(30000);
        }

        Console.WriteLine("DB instance deleted.");
        Console.WriteLine(sepBar);
    }

    /// <summary>
    /// Delete a DB parameter group.
    /// </summary>
    /// <param name="parameterGroup">The parameter group to delete.</param>
    /// <returns>Async task.</returns>
    public static async Task DeleteParameterGroup(DBParameterGroup
 parameterGroup)
    {
        Console.WriteLine(sepBar);
        // Delete the parameter group.
        Console.WriteLine($"17. Delete the DB parameter group
 {parameterGroup.DBParameterGroupName}.");
        await
 rdsWrapper.DeleteDBParameterGroup(parameterGroup.DBParameterGroupName);

        Console.WriteLine(sepBar);
```

```
    }
```

场景用于数据库实例操作的包装程序方法。

```
/// <summary>
/// Wrapper methods to use Amazon Relational Database Service (Amazon RDS) with
 DB instance operations.
/// </summary>
public partial class RDSWrapper
{
    private readonly IAmazonRDS _amazonRDS;
    public RDSWrapper(IAmazonRDS amazonRDS)
    {
        _amazonRDS = amazonRDS;
    }


    /// <summary>
    /// Get a list of DB engine versions for a particular DB engine.
    /// </summary>
    /// <param name="engine">Name of the engine.</param>
    /// <param name="dbParameterGroupFamily">Optional parameter group family
 name.</param>
    /// <returns>List of DBEngineVersions.</returns>
    public async Task<List<DBEngineVersion>> DescribeDBEngineVersions(string
 engine,
        string dbParameterGroupFamily = null)
    {
        var response = await _amazonRDS.DescribeDBEngineVersionsAsync(
            new DescribeDBEngineVersionsRequest()
            {
                Engine = engine,
                DBParameterGroupFamily = dbParameterGroupFamily
            });
        return response.DBEngineVersions;
    }



    /// <summary>
    /// Get a list of orderable DB instance options for a specific
```

```
    /// engine and engine version.
    /// </summary>
    /// <param name="engine">Name of the engine.</param>
    /// <param name="engineVersion">Version of the engine.</param>
    /// <returns>List of OrderableDBInstanceOptions.</returns>
    public async Task<List<OrderableDBInstanceOption>>
 DescribeOrderableDBInstanceOptions(string engine, string engineVersion)
    {
        // Use a paginator to get a list of DB instance options.
        var results = new List<OrderableDBInstanceOption>();
        var paginateInstanceOptions =
 _amazonRDS.Paginators.DescribeOrderableDBInstanceOptions(
            new DescribeOrderableDBInstanceOptionsRequest()
            {
                Engine = engine,
                EngineVersion = engineVersion,
            });
        // Get the entire list using the paginator.
        await foreach (var instanceOptions in
 paginateInstanceOptions.OrderableDBInstanceOptions)
        {
            results.Add(instanceOptions);
        }
        return results;
    }



    /// <summary>
    /// Returns a list of DB instances.
    /// </summary>
    /// <param name="dbInstanceIdentifier">Optional name of a specific DB
 instance.</param>
    /// <returns>List of DB instances.</returns>
    public async Task<List<DBInstance>> DescribeDBInstances(string
 dbInstanceIdentifier = null)
    {
        var results = new List<DBInstance>();
        var instancesPaginator = _amazonRDS.Paginators.DescribeDBInstances(
            new DescribeDBInstancesRequest
            {
                DBInstanceIdentifier = dbInstanceIdentifier
            });
        // Get the entire list using the paginator.
```

```
        await foreach (var instances in instancesPaginator.DBInstances)
        {
            results.Add(instances);
        }
        return results;
    }



    /// <summary>
    /// Create an RDS DB instance with a particular set of properties. Use the
 action DescribeDBInstancesAsync
    /// to determine when the DB instance is ready to use.
    /// </summary>
    /// <param name="dbName">Name for the DB instance.</param>
    /// <param name="dbInstanceIdentifier">DB instance identifier.</param>
    /// <param name="parameterGroupName">DB parameter group to associate with the
 instance.</param>
    /// <param name="dbEngine">The engine for the DB instance.</param>
    /// <param name="dbEngineVersion">Version for the DB instance.</param>
    /// <param name="instanceClass">Class for the DB instance.</param>
    /// <param name="allocatedStorage">The amount of storage in gibibytes (GiB)
 to allocate to the DB instance.</param>
    /// <param name="adminName">Admin user name.</param>
    /// <param name="adminPassword">Admin user password.</param>
    /// <returns>DB instance object.</returns>
    public async Task<DBInstance> CreateDBInstance(string dbName, string
 dbInstanceIdentifier,
        string parameterGroupName, string dbEngine, string dbEngineVersion,
        string instanceClass, int allocatedStorage, string adminName, string
 adminPassword)
    {
        var response = await _amazonRDS.CreateDBInstanceAsync(
            new CreateDBInstanceRequest()
            {
                DBName = dbName,
                DBInstanceIdentifier = dbInstanceIdentifier,
                DBParameterGroupName = parameterGroupName,
                Engine = dbEngine,
                EngineVersion = dbEngineVersion,
                DBInstanceClass = instanceClass,
                AllocatedStorage = allocatedStorage,
                MasterUsername = adminName,
                MasterUserPassword = adminPassword
```

```
            });

        return response.DBInstance;
    }



    /// <summary>
    /// Delete a particular DB instance.
    /// </summary>
    /// <param name="dbInstanceIdentifier">DB instance identifier.</param>
    /// <returns>DB instance object.</returns>
    public async Task<DBInstance> DeleteDBInstance(string dbInstanceIdentifier)
    {
        var response = await _amazonRDS.DeleteDBInstanceAsync(
            new DeleteDBInstanceRequest()
            {
                DBInstanceIdentifier = dbInstanceIdentifier,
                SkipFinalSnapshot = true,
                DeleteAutomatedBackups = true
            });

        return response.DBInstance;
    }
```

场景用于数据库参数组的包装程序方法。

```
/// <summary>
/// Wrapper methods to use Amazon Relational Database Service (Amazon RDS) with
 parameter groups.
/// </summary>
public partial class RDSWrapper
{

    /// <summary>
    /// Get descriptions of DB parameter groups.
    /// </summary>
    /// <param name="name">Optional name of the DB parameter group to describe.</
param>
    /// <returns>The list of DB parameter group descriptions.</returns>
```

```
    public async Task<List<DBParameterGroup>> DescribeDBParameterGroups(string
 name = null)
    {
        var response = await _amazonRDS.DescribeDBParameterGroupsAsync(
            new DescribeDBParameterGroupsRequest()
            {
                DBParameterGroupName = name
            });
        return response.DBParameterGroups;
    }



    /// <summary>
    /// Create a new DB parameter group. Use the action
 DescribeDBParameterGroupsAsync
    /// to determine when the DB parameter group is ready to use.
    /// </summary>
    /// <param name="name">Name of the DB parameter group.</param>
    /// <param name="family">Family of the DB parameter group.</param>
    /// <param name="description">Description of the DB parameter group.</param>
    /// <returns>The new DB parameter group.</returns>
    public async Task<DBParameterGroup> CreateDBParameterGroup(
        string name, string family, string description)
    {
        var response = await _amazonRDS.CreateDBParameterGroupAsync(
            new CreateDBParameterGroupRequest()
            {
                DBParameterGroupName = name,
                DBParameterGroupFamily = family,
                Description = description
            });
        return response.DBParameterGroup;
    }



    /// <summary>
    /// Update a DB parameter group. Use the action
 DescribeDBParameterGroupsAsync
    /// to determine when the DB parameter group is ready to use.
    /// </summary>
    /// <param name="name">Name of the DB parameter group.</param>
```

```
    /// <param name="parameters">List of parameters. Maximum of 20 per request.</
param>
    /// <returns>The updated DB parameter group name.</returns>
    public async Task<string> ModifyDBParameterGroup(
        string name, List<Parameter> parameters)
    {
        var response = await _amazonRDS.ModifyDBParameterGroupAsync(
            new ModifyDBParameterGroupRequest()
            {
                DBParameterGroupName = name,
                Parameters = parameters,
            });
        return response.DBParameterGroupName;
    }



    /// <summary>
    /// Delete a DB parameter group. The group cannot be a default DB parameter
 group
    /// or be associated with any DB instances.
    /// </summary>
    /// <param name="name">Name of the DB parameter group.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> DeleteDBParameterGroup(string name)
    {
        var response = await _amazonRDS.DeleteDBParameterGroupAsync(
            new DeleteDBParameterGroupRequest()
            {
                DBParameterGroupName = name,
            });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }



    /// <summary>
    /// Get a list of DB parameters from a specific parameter group.
    /// </summary>
    /// <param name="dbParameterGroupName">Name of a specific DB parameter
 group.</param>
    /// <param name="source">Optional source for selecting parameters.</param>
    /// <returns>List of parameter values.</returns>
```

```
    public async Task<List<Parameter>> DescribeDBParameters(string
 dbParameterGroupName, string source = null)
    {
        var results = new List<Parameter>();
        var paginateParameters = _amazonRDS.Paginators.DescribeDBParameters(
            new DescribeDBParametersRequest()
            {
                DBParameterGroupName = dbParameterGroupName,
                Source = source
            });
        // Get the entire list using the paginator.
        await foreach (var parameters in paginateParameters.Parameters)
        {
            results.Add(parameters);
        }
        return results;
    }
```

场景用于数据库快照操作的包装程序方法。

```
/// <summary>
/// Wrapper methods to use Amazon Relational Database Service (Amazon RDS) with
 snapshots.
/// </summary>
public partial class RDSWrapper
{

    /// <summary>
    /// Create a snapshot of a DB instance.
    /// </summary>
    /// <param name="dbInstanceIdentifier">DB instance identifier.</param>
    /// <param name="snapshotIdentifier">Identifier for the snapshot.</param>
    /// <returns>DB snapshot object.</returns>
    public async Task<DBSnapshot> CreateDBSnapshot(string dbInstanceIdentifier,
 string snapshotIdentifier)
    {
        var response = await _amazonRDS.CreateDBSnapshotAsync(
            new CreateDBSnapshotRequest()
            {
                DBSnapshotIdentifier = snapshotIdentifier,
```

```
                DBInstanceIdentifier = dbInstanceIdentifier
            });

        return response.DBSnapshot;
    }



    /// <summary>
    /// Return a list of DB snapshots for a particular DB instance.
    /// </summary>
    /// <param name="dbInstanceIdentifier">DB instance identifier.</param>
    /// <returns>List of DB snapshots.</returns>
    public async Task<List<DBSnapshot>> DescribeDBSnapshots(string
 dbInstanceIdentifier)
    {
        var results = new List<DBSnapshot>();
        var snapshotsPaginator = _amazonRDS.Paginators.DescribeDBSnapshots(
            new DescribeDBSnapshotsRequest()
            {
                DBInstanceIdentifier = dbInstanceIdentifier
            });

        // Get the entire list using the paginator.
        await foreach (var snapshots in snapshotsPaginator.DBSnapshots)
        {
            results.Add(snapshots);
        }
        return results;
    }
```

- 有关 API 详细信息，请参阅《适用于 .NET 的 Amazon SDK API 参考》中的以下主题。
  - CreateDBInstance
  - CreateDBParameterGroup
  - CreateDBSnapshot
  - DeleteDBInstance
  - DeleteDBParameterGroup
  - DescribeDBEngineVersions

- DescribeDBInstances
- DescribeDBParameterGroups
- DescribeDBParameters
- DescribeDBSnapshots
- DescribeOrderableDBInstanceOptions
- ModifyDBParameterGroup

C++

SDK for C++

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库 中查找完整示例，了解如何进行设置和运行。

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

//! Routine which creates an Amazon RDS instance and demonstrates several
 operations
//! on that instance.
/*!
 \sa gettingStartedWithDBInstances()
 \param clientConfiguration: AWS client configuration.
 \return bool: Successful completion.
 */
bool AwsDoc::RDS::gettingStartedWithDBInstances(
        const Aws::Client::ClientConfiguration &clientConfig) {
    Aws::RDS::RDSClient client(clientConfig);

    printAsterisksLine();
    std::cout << "Welcome to the Amazon Relational Database Service (Amazon RDS)"
              << std::endl;
    std::cout << "get started with DB instances demo." << std::endl;
    printAsterisksLine();
```

```
    std::cout << "Checking for an existing DB parameter group named '" <<
              PARAMETER_GROUP_NAME << "'." << std::endl;
    Aws::String dbParameterGroupFamily("Undefined");
    bool parameterGroupFound = true;
    {
        // 1. Check if the DB parameter group already exists.
        Aws::RDS::Model::DescribeDBParameterGroupsRequest request;
        request.SetDBParameterGroupName(PARAMETER_GROUP_NAME);

        Aws::RDS::Model::DescribeDBParameterGroupsOutcome outcome =
                client.DescribeDBParameterGroups(request);

        if (outcome.IsSuccess()) {
            std::cout << "DB parameter group named '" <<
                      PARAMETER_GROUP_NAME << "' already exists." << std::endl;
            dbParameterGroupFamily = outcome.GetResult().GetDBParameterGroups()
[0].GetDBParameterGroupFamily();
        }
        else if (outcome.GetError().GetErrorType() ==
                 Aws::RDS::RDSErrors::D_B_PARAMETER_GROUP_NOT_FOUND_FAULT) {
            std::cout << "DB parameter group named '" <<
                      PARAMETER_GROUP_NAME << "' does not exist." << std::endl;
            parameterGroupFound = false;
        }
        else {
            std::cerr << "Error with RDS::DescribeDBParameterGroups. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            return false;
        }
    }

    if (!parameterGroupFound) {
        Aws::Vector<Aws::RDS::Model::DBEngineVersion> engineVersions;

        // 2. Get available engine versions for the specified engine.
        if (!getDBEngineVersions(DB_ENGINE, NO_PARAMETER_GROUP_FAMILY,
                                 engineVersions, client)) {
            return false;
        }

        std::cout << "Getting available database engine versions for " <<
 DB_ENGINE
                  << "."
```

```
                  << std::endl;
        std::vector<Aws::String> families;
        for (const Aws::RDS::Model::DBEngineVersion &version: engineVersions) {
            Aws::String family = version.GetDBParameterGroupFamily();
            if (std::find(families.begin(), families.end(), family) ==
                families.end()) {
                families.push_back(family);
                std::cout << "  " << families.size() << ": " << family <<
std::endl;
            }
        }

        int choice = askQuestionForIntRange("Which family do you want to use? ",
1,
                                            static_cast<int>(families.size()));
        dbParameterGroupFamily = families[choice - 1];
    }
    if (!parameterGroupFound) {
        // 3.  Create a DB parameter group.
        Aws::RDS::Model::CreateDBParameterGroupRequest request;
        request.SetDBParameterGroupName(PARAMETER_GROUP_NAME);
        request.SetDBParameterGroupFamily(dbParameterGroupFamily);
        request.SetDescription("Example parameter group.");

        Aws::RDS::Model::CreateDBParameterGroupOutcome outcome =
                client.CreateDBParameterGroup(request);

        if (outcome.IsSuccess()) {
            std::cout << "The DB parameter group was successfully created."
                      << std::endl;
        }
        else {
            std::cerr << "Error with RDS::CreateDBParameterGroup. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            return false;
        }
    }

    printAsterisksLine();
    std::cout << "Let's set some parameter values in your parameter group."
              << std::endl;

    Aws::String marker;
```

```
    Aws::Vector<Aws::RDS::Model::Parameter> autoIncrementParameters;
    // 4.  Get the parameters in the DB parameter group.
    if (!getDBParameters(PARAMETER_GROUP_NAME, AUTO_INCREMENT_PREFIX, NO_SOURCE,
                         autoIncrementParameters,
                         client)) {
        cleanUpResources(PARAMETER_GROUP_NAME, "", client);
        return false;
    }

    Aws::Vector<Aws::RDS::Model::Parameter> updateParameters;

    for (Aws::RDS::Model::Parameter &autoIncParameter: autoIncrementParameters) {
        if (autoIncParameter.GetIsModifiable() &&
            (autoIncParameter.GetDataType() == "integer")) {
            std::cout << "The " << autoIncParameter.GetParameterName()
                      << " is described as: " <<
                      autoIncParameter.GetDescription() << "." << std::endl;
            if (autoIncParameter.ParameterValueHasBeenSet()) {
                std::cout << "The current value is "
                          << autoIncParameter.GetParameterValue()
                          << "." << std::endl;
            }
            std::vector<int> splitValues = splitToInts(
                    autoIncParameter.GetAllowedValues(), '-');
            if (splitValues.size() == 2) {
                int newValue = askQuestionForIntRange(
                        Aws::String("Enter a new value in the range ") +
                        autoIncParameter.GetAllowedValues() + ": ",
                        splitValues[0], splitValues[1]);
                autoIncParameter.SetParameterValue(std::to_string(newValue));
                updateParameters.push_back(autoIncParameter);

            }
            else {
                std::cerr << "Error parsing " <<
autoIncParameter.GetAllowedValues()
                          << std::endl;
            }
        }
    }

    {
        // 5.  Modify the auto increment parameters in the group.
        Aws::RDS::Model::ModifyDBParameterGroupRequest request;
```

```
        request.SetDBParameterGroupName(PARAMETER_GROUP_NAME);
        request.SetParameters(updateParameters);

        Aws::RDS::Model::ModifyDBParameterGroupOutcome outcome =
                client.ModifyDBParameterGroup(request);

        if (outcome.IsSuccess()) {
            std::cout << "The DB parameter group was successfully modified."
                      << std::endl;
        }
        else {
            std::cerr << "Error with RDS::ModifyDBParameterGroup. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
        }
    }

    std::cout
            << "You can get a list of parameters you've set by specifying a
 source of 'user'."
            << std::endl;

    Aws::Vector<Aws::RDS::Model::Parameter> userParameters;
    // 6.  Display the modified parameters in the group.
    if (!getDBParameters(PARAMETER_GROUP_NAME, NO_NAME_PREFIX, "user",
 userParameters,
                         client)) {
        cleanUpResources(PARAMETER_GROUP_NAME, "", client);
        return false;
    }

    for (const auto &userParameter: userParameters) {
        std::cout << "  " << userParameter.GetParameterName() << ", " <<
                  userParameter.GetDescription() << ", parameter value - "
                  << userParameter.GetParameterValue() << std::endl;
    }

    printAsterisksLine();
    std::cout << "Checking for an existing DB instance." << std::endl;

    Aws::RDS::Model::DBInstance dbInstance;
    // 7.  Check if the DB instance already exists.
    if (!describeDBInstance(DB_INSTANCE_IDENTIFIER, dbInstance, client)) {
        cleanUpResources(PARAMETER_GROUP_NAME, "", client);
```

```
        return false;
    }

    if (dbInstance.DbInstancePortHasBeenSet()) {
        std::cout << "The DB instance already exists." << std::endl;
    }
    else {
        std::cout << "Let's create a DB instance." << std::endl;
        const Aws::String administratorName = askQuestion(
                "Enter an administrator username for the database: ");
        const Aws::String administratorPassword = askQuestion(
                "Enter a password for the administrator (at least 8 characters):
");
        Aws::Vector<Aws::RDS::Model::DBEngineVersion> engineVersions;

        // 8.  Get a list of available engine versions.
        if (!getDBEngineVersions(DB_ENGINE, dbParameterGroupFamily,
engineVersions,
                                client)) {
            cleanUpResources(PARAMETER_GROUP_NAME, "", client);
            return false;
        }

        std::cout << "The available engines for your parameter group are:" <<
std::endl;

        int index = 1;
        for (const Aws::RDS::Model::DBEngineVersion &engineVersion:
engineVersions) {
            std::cout << "  " << index << ": " <<
engineVersion.GetEngineVersion()
                      << std::endl;
            ++index;
        }
        int choice = askQuestionForIntRange("Which engine do you want to use? ",
1,

static_cast<int>(engineVersions.size()));
        const Aws::RDS::Model::DBEngineVersion engineVersion =
engineVersions[choice -
                                                                              1];

        Aws::String dbInstanceClass;
        // 9.  Get a list of micro instance classes.
```

```
        if (!chooseMicroDBInstanceClass(engineVersion.GetEngine(),
                                        engineVersion.GetEngineVersion(),
                                        dbInstanceClass,
                                        client)) {
            cleanUpResources(PARAMETER_GROUP_NAME, "", client);
            return false;
        }

        std::cout << "Creating a DB instance named '" << DB_INSTANCE_IDENTIFIER
                  << "' and database '" << DB_NAME << "'.\n"
                  << "The DB instance is configured to use your custom parameter
group '"
                  << PARAMETER_GROUP_NAME << "',\n"
                  << "selected engine version " <<
engineVersion.GetEngineVersion()
                  << ",\n"
                  << "selected DB instance class '" << dbInstanceClass << "',"
                  << " and " << DB_ALLOCATED_STORAGE << " GiB of " <<
DB_STORAGE_TYPE
                  << " storage.\nThis typically takes several minutes." <<
std::endl;

        Aws::RDS::Model::CreateDBInstanceRequest request;
        request.SetDBName(DB_NAME);
        request.SetDBInstanceIdentifier(DB_INSTANCE_IDENTIFIER);
        request.SetDBParameterGroupName(PARAMETER_GROUP_NAME);
        request.SetEngine(engineVersion.GetEngine());
        request.SetEngineVersion(engineVersion.GetEngineVersion());
        request.SetDBInstanceClass(dbInstanceClass);
        request.SetStorageType(DB_STORAGE_TYPE);
        request.SetAllocatedStorage(DB_ALLOCATED_STORAGE);
        request.SetMasterUsername(administratorName);
        request.SetMasterUserPassword(administratorPassword);

        Aws::RDS::Model::CreateDBInstanceOutcome outcome =
                client.CreateDBInstance(request);

        if (outcome.IsSuccess()) {
            std::cout << "The DB instance creation has started."
                      << std::endl;
        }
        else {
            std::cerr << "Error with RDS::CreateDBInstance. "
                      << outcome.GetError().GetMessage()
```

```
                     << std::endl;
            cleanUpResources(PARAMETER_GROUP_NAME, "", client);
            return false;
        }
    }

    std::cout << "Waiting for the DB instance to become available." << std::endl;

    int counter = 0;
    // 11. Wait for the DB instance to become available.
    do {
        std::this_thread::sleep_for(std::chrono::seconds(1));
        ++counter;
        if (counter > 900) {
            std::cerr << "Wait for instance to become available timed out ofter "
                      << counter
                      << " seconds." << std::endl;
            cleanUpResources(PARAMETER_GROUP_NAME, DB_INSTANCE_IDENTIFIER,
 client);
            return false;
        }

        dbInstance = Aws::RDS::Model::DBInstance();
        if (!describeDBInstance(DB_INSTANCE_IDENTIFIER, dbInstance, client)) {
            cleanUpResources(PARAMETER_GROUP_NAME, DB_INSTANCE_IDENTIFIER,
 client);
            return false;
        }

        if ((counter % 20) == 0) {
            std::cout << "Current DB instance status is '"
                      << dbInstance.GetDBInstanceStatus()
                      << "' after " << counter << " seconds." << std::endl;
        }
    } while (dbInstance.GetDBInstanceStatus() != "available");

    if (dbInstance.GetDBInstanceStatus() == "available") {
        std::cout << "The DB instance has been created." << std::endl;
    }

    printAsterisksLine();

    // 12. Display the connection string that can be used to connect a 'mysql'
 shell to the database.
```

```
    displayConnection(dbInstance);

    printAsterisksLine();

    if (askYesNoQuestion(
            "Do you want to create a snapshot of your DB instance (y/n)? ")) {
        Aws::String snapshotID(DB_INSTANCE_IDENTIFIER + "-" +
                               Aws::String(Aws::Utils::UUID::RandomUUID()));
        {
            std::cout << "Creating a snapshot named " << snapshotID << "." <<
 std::endl;
            std::cout << "This typically takes a few minutes." << std::endl;

            // 13. Create a snapshot of the DB instance.
            Aws::RDS::Model::CreateDBSnapshotRequest request;
            request.SetDBInstanceIdentifier(DB_INSTANCE_IDENTIFIER);
            request.SetDBSnapshotIdentifier(snapshotID);

            Aws::RDS::Model::CreateDBSnapshotOutcome outcome =
                    client.CreateDBSnapshot(request);

            if (outcome.IsSuccess()) {
                std::cout << "Snapshot creation has started."
                          << std::endl;
            }
            else {
                std::cerr << "Error with RDS::CreateDBSnapshot. "
                          << outcome.GetError().GetMessage()
                          << std::endl;
                cleanUpResources(PARAMETER_GROUP_NAME, DB_INSTANCE_IDENTIFIER,
 client);
                return false;
            }
        }

        std::cout << "Waiting for snapshot to become available." << std::endl;

        Aws::RDS::Model::DBSnapshot snapshot;
        counter = 0;
        do {
            std::this_thread::sleep_for(std::chrono::seconds(1));
            ++counter;
            if (counter > 600) {
                std::cerr << "Wait for snapshot to be available timed out ofter "
```

```
                            << counter
                            << " seconds." << std::endl;
                cleanUpResources(PARAMETER_GROUP_NAME, DB_INSTANCE_IDENTIFIER,
client);
                return false;
            }

            // 14. Wait for the snapshot to become available.
            Aws::RDS::Model::DescribeDBSnapshotsRequest request;
            request.SetDBSnapshotIdentifier(snapshotID);

            Aws::RDS::Model::DescribeDBSnapshotsOutcome outcome =
                    client.DescribeDBSnapshots(request);

            if (outcome.IsSuccess()) {
                snapshot = outcome.GetResult().GetDBSnapshots()[0];
            }
            else {
                std::cerr << "Error with RDS::DescribeDBSnapshots. "
                          << outcome.GetError().GetMessage()
                          << std::endl;
                cleanUpResources(PARAMETER_GROUP_NAME, DB_INSTANCE_IDENTIFIER,
client);
                return false;
            }

            if ((counter % 20) == 0) {
                std::cout << "Current snapshot status is '"
                          << snapshot.GetStatus()
                          << "' after " << counter << " seconds." << std::endl;
            }
        } while (snapshot.GetStatus() != "available");

        if (snapshot.GetStatus() != "available") {
            std::cout << "A snapshot has been created." << std::endl;
        }
    }

    printAsterisksLine();

    bool result = true;
    if (askYesNoQuestion(
            "Do you want to delete the DB instance and parameter group (y/n)? "))
{
```

```
        result = cleanUpResources(PARAMETER_GROUP_NAME, DB_INSTANCE_IDENTIFIER,
 client);
    }

    return result;
}


//! Routine which gets DB parameters using the 'DescribeDBParameters' api.
/*!
 \sa getDBParameters()
 \param parameterGroupName: The name of the parameter group.
 \param namePrefix: Prefix string to filter results by parameter name.
 \param source: A source such as 'user', ignored if empty.
 \param parametersResult: Vector of 'Parameter' objects returned by the routine.
 \param client: 'RDSClient' instance.
 \return bool: Successful completion.
 */
bool AwsDoc::RDS::getDBParameters(const Aws::String &parameterGroupName,
                                  const Aws::String &namePrefix,
                                  const Aws::String &source,
                                  Aws::Vector<Aws::RDS::Model::Parameter>
 &parametersResult,
                                  const Aws::RDS::RDSClient &client) {
    Aws::String marker;
    do {
        Aws::RDS::Model::DescribeDBParametersRequest request;
        request.SetDBParameterGroupName(PARAMETER_GROUP_NAME);
        if (!marker.empty()) {
            request.SetMarker(marker);
        }
        if (!source.empty()) {
            request.SetSource(source);
        }

        Aws::RDS::Model::DescribeDBParametersOutcome outcome =
                client.DescribeDBParameters(request);

        if (outcome.IsSuccess()) {
            const Aws::Vector<Aws::RDS::Model::Parameter> &parameters =
                    outcome.GetResult().GetParameters();
            for (const Aws::RDS::Model::Parameter &parameter: parameters) {
                if (!namePrefix.empty()) {
                    if (parameter.GetParameterName().find(namePrefix) == 0) {
```

```
                        parametersResult.push_back(parameter);
                    }
                }
                else {
                    parametersResult.push_back(parameter);
                }
            }

            marker = outcome.GetResult().GetMarker();
        }
        else {
            std::cerr << "Error with RDS::DescribeDBParameters. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            return false;
        }
    } while (!marker.empty());

    return true;
}


//! Routine which gets available DB engine versions for an engine name and
//! an optional parameter group family.
/*!
 \sa getDBEngineVersions()
 \param engineName: A DB engine name.
 \param parameterGroupFamily: A parameter group family name, ignored if empty.
 \param engineVersionsResult: Vector of 'DBEngineVersion' objects returned by the
 routine.
 \param client: 'RDSClient' instance.
 \return bool: Successful completion.
 */
bool AwsDoc::RDS::getDBEngineVersions(const Aws::String &engineName,
                                      const Aws::String &parameterGroupFamily,

 Aws::Vector<Aws::RDS::Model::DBEngineVersion> &engineVersionsResult,
                                      const Aws::RDS::RDSClient &client) {
    Aws::RDS::Model::DescribeDBEngineVersionsRequest request;
    request.SetEngine(engineName);
    if (!parameterGroupFamily.empty()) {
        request.SetDBParameterGroupFamily(parameterGroupFamily);
    }
```

```
    engineVersionsResult.clear();
    Aws::String marker; // Used for pagination.

    do {
        if (!marker.empty()) {
            request.SetMarker(marker);
        }


        Aws::RDS::Model::DescribeDBEngineVersionsOutcome outcome =
                client.DescribeDBEngineVersions(request);

        if (outcome.IsSuccess()) {
            auto &engineVersions = outcome.GetResult().GetDBEngineVersions();
            engineVersionsResult.insert(engineVersionsResult.end(),
 engineVersions.begin(),
                                        engineVersions.end());
            marker = outcome.GetResult().GetMarker();
        }
        else {
            std::cerr << "Error with RDS::DescribeDBEngineVersionsRequest. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            return false;
        }

    } while (!marker.empty());


    return true;
}


//! Routine which gets a DB instance description.
/*!
 \sa describeDBInstance()
 \param dbInstanceIdentifier: A DB instance identifier.
 \param instanceResult: The 'DBInstance' object containing the description.
 \param client: 'RDSClient' instance.
 \return bool: Successful completion.
 */
bool AwsDoc::RDS::describeDBInstance(const Aws::String &dbInstanceIdentifier,
                                     Aws::RDS::Model::DBInstance &instanceResult,
                                     const Aws::RDS::RDSClient &client) {
```

```
    Aws::RDS::Model::DescribeDBInstancesRequest request;
    request.SetDBInstanceIdentifier(dbInstanceIdentifier);

    Aws::RDS::Model::DescribeDBInstancesOutcome outcome =
            client.DescribeDBInstances(request);

    bool result = true;
    if (outcome.IsSuccess()) {
        instanceResult = outcome.GetResult().GetDBInstances()[0];
    }
    else if (outcome.GetError().GetErrorType() !=
             Aws::RDS::RDSErrors::D_B_INSTANCE_NOT_FOUND_FAULT) {
        result = false;
        std::cerr << "Error with RDS::DescribeDBInstances. "
                  << outcome.GetError().GetMessage()
                  << std::endl;
    }
        // This example does not log an error if the DB instance does not exist.
        // Instead, instanceResult is set to empty.
    else {
        instanceResult = Aws::RDS::Model::DBInstance();
    }

    return result;
}


//! Routine which gets available 'micro' DB instance classes, displays the list
//! to the user, and returns the user selection.
/*!
 \sa chooseMicroDBInstanceClass()
 \param engineName: The DB engine name.
 \param engineVersion: The DB engine version.
 \param dbInstanceClass: String for DB instance class chosen by the user.
 \param client: 'RDSClient' instance.
 \return bool: Successful completion.
 */
bool AwsDoc::RDS::chooseMicroDBInstanceClass(const Aws::String &engine,
                                             const Aws::String &engineVersion,
                                             Aws::String &dbInstanceClass,
                                             const Aws::RDS::RDSClient &client) {
    std::vector<Aws::String> instanceClasses;
    Aws::String marker;
    do {
```

```
        Aws::RDS::Model::DescribeOrderableDBInstanceOptionsRequest request;
        request.SetEngine(engine);
        request.SetEngineVersion(engineVersion);
        if (!marker.empty()) {
            request.SetMarker(marker);
        }

        Aws::RDS::Model::DescribeOrderableDBInstanceOptionsOutcome outcome =
                client.DescribeOrderableDBInstanceOptions(request);

        if (outcome.IsSuccess()) {
            const Aws::Vector<Aws::RDS::Model::OrderableDBInstanceOption>
&options =
                    outcome.GetResult().GetOrderableDBInstanceOptions();
            for (const Aws::RDS::Model::OrderableDBInstanceOption &option:
options) {
                const Aws::String &instanceClass = option.GetDBInstanceClass();
                if (instanceClass.find("micro") != std::string::npos) {
                    if (std::find(instanceClasses.begin(), instanceClasses.end(),
                                  instanceClass) ==
                        instanceClasses.end()) {
                        instanceClasses.push_back(instanceClass);
                    }
                }
            }
            marker = outcome.GetResult().GetMarker();
        }
        else {
            std::cerr << "Error with RDS::DescribeOrderableDBInstanceOptions. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            return false;
        }
    } while (!marker.empty());

    std::cout << "The available micro DB instance classes for your database
engine are:"
             << std::endl;
    for (int i = 0; i < instanceClasses.size(); ++i) {
        std::cout << "   " << i + 1 << ": " << instanceClasses[i] << std::endl;
    }

    int choice = askQuestionForIntRange(
            "Which micro DB instance class do you want to use? ",
```

```
            1, static_cast<int>(instanceClasses.size()));
    dbInstanceClass = instanceClasses[choice - 1];
    return true;
}

//! Routine which deletes resources created by the scenario.
/*!
\sa cleanUpResources()
\param parameterGroupName: A parameter group name, this may be empty.
\param dbInstanceIdentifier: A DB instance identifier, this may be empty.
\param client: 'RDSClient' instance.
\return bool: Successful completion.
*/
bool AwsDoc::RDS::cleanUpResources(const Aws::String &parameterGroupName,
                                   const Aws::String &dbInstanceIdentifier,
                                   const Aws::RDS::RDSClient &client) {
    bool result = true;
    if (!dbInstanceIdentifier.empty()) {
        {
            // 15. Delete the DB instance.
            Aws::RDS::Model::DeleteDBInstanceRequest request;
            request.SetDBInstanceIdentifier(dbInstanceIdentifier);
            request.SetSkipFinalSnapshot(true);
            request.SetDeleteAutomatedBackups(true);

            Aws::RDS::Model::DeleteDBInstanceOutcome outcome =
                    client.DeleteDBInstance(request);

            if (outcome.IsSuccess()) {
                std::cout << "DB instance deletion has started."
                          << std::endl;
            }
            else {
                std::cerr << "Error with RDS::DeleteDBInstance. "
                          << outcome.GetError().GetMessage()
                          << std::endl;
                result = false;
            }
        }

        std::cout
                << "Waiting for DB instance to delete before deleting the
 parameter group."
                << std::endl;
```

```
        std::cout << "This may take a while." << std::endl;

        int counter = 0;
        Aws::RDS::Model::DBInstance dbInstance;
        do {
            std::this_thread::sleep_for(std::chrono::seconds(1));
            ++counter;
            if (counter > 800) {
                std::cerr << "Wait for instance to delete timed out ofter " <<
counter
                          << " seconds." << std::endl;
                return false;
            }

            dbInstance = Aws::RDS::Model::DBInstance();
            // 16. Wait for the DB instance to be deleted.
            if (!describeDBInstance(dbInstanceIdentifier, dbInstance, client)) {
                return false;
            }

            if (dbInstance.DBInstanceIdentifierHasBeenSet() && (counter % 20) ==
0) {
                std::cout << "Current DB instance status is '"
                          << dbInstance.GetDBInstanceStatus()
                          << "' after " << counter << " seconds." << std::endl;
            }
        } while (dbInstance.DBInstanceIdentifierHasBeenSet());
    }

    if (!parameterGroupName.empty()) {
        // 17. Delete the parameter group.
        Aws::RDS::Model::DeleteDBParameterGroupRequest request;
        request.SetDBParameterGroupName(parameterGroupName);

        Aws::RDS::Model::DeleteDBParameterGroupOutcome outcome =
                client.DeleteDBParameterGroup(request);

        if (outcome.IsSuccess()) {
            std::cout << "The DB parameter group was successfully deleted."
                      << std::endl;
        }
        else {
            std::cerr << "Error with RDS::DeleteDBParameterGroup. "
                      << outcome.GetError().GetMessage()
```

```
                        << std::endl;
                result = false;
            }
        }

        return result;
}
```

- 有关 API 详细信息，请参阅《适用于 C++ 的 Amazon SDK API 参考》中的以下主题。
  - CreateDBInstance
  - CreateDBParameterGroup
  - CreateDBSnapshot
  - DeleteDBInstance
  - DeleteDBParameterGroup
  - DescribeDBEngineVersions
  - DescribeDBInstances
  - DescribeDBParameterGroups
  - DescribeDBParameters
  - DescribeDBSnapshots
  - DescribeOrderableDBInstanceOptions
  - ModifyDBParameterGroup

Go

适用于 Go V2 的 SDK

> **Note**
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

在命令提示符中运行交互式场景。

```
import (
```

```
 "context"
 "fmt"
 "log"
 "sort"
 "strconv"
 "strings"
 "time"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/rds"
 "github.com/aws/aws-sdk-go-v2/service/rds/types"
 "github.com/awsdocs/aws-doc-sdk-examples/gov2/demotools"
 "github.com/awsdocs/aws-doc-sdk-examples/gov2/rds/actions"
 "github.com/google/uuid"
)

// GetStartedInstances is an interactive example that shows you how to use the
 AWS SDK for Go
// with Amazon Relation Database Service (Amazon RDS) to do the following:
//
//  1. Create a custom DB parameter group and set parameter values.
//  2. Create a DB instance that is configured to use the parameter group. The DB
 instance
//     also contains a database.
//  3. Take a snapshot of the DB instance.
//  4. Delete the DB instance and parameter group.
type GetStartedInstances struct {
 sdkConfig  aws.Config
 instances  actions.DbInstances
 questioner demotools.IQuestioner
 helper     IScenarioHelper
 isTestRun  bool
}

// NewGetStartedInstances constructs a GetStartedInstances instance from a
 configuration.
// It uses the specified config to get an Amazon RDS
// client and create wrappers for the actions used in the scenario.
func NewGetStartedInstances(sdkConfig aws.Config, questioner
 demotools.IQuestioner,
 helper IScenarioHelper) GetStartedInstances {
 rdsClient := rds.NewFromConfig(sdkConfig)
 return GetStartedInstances{
  sdkConfig:  sdkConfig,
```

```
  instances:  actions.DbInstances{RdsClient: rdsClient},
  questioner: questioner,
  helper:     helper,
 }
}

// Run runs the interactive scenario.
func (scenario GetStartedInstances) Run(ctx context.Context, dbEngine string,
 parameterGroupName string,
 instanceName string, dbName string) {
 defer func() {
  if r := recover(); r != nil {
   log.Println("Something went wrong with the demo.")
  }
 }()

 log.Println(strings.Repeat("-", 88))
 log.Println("Welcome to the Amazon Relational Database Service (Amazon RDS) DB
 Instance demo.")
 log.Println(strings.Repeat("-", 88))

 parameterGroup := scenario.CreateParameterGroup(ctx, dbEngine,
 parameterGroupName)
 scenario.SetUserParameters(ctx, parameterGroupName)
 instance := scenario.CreateInstance(ctx, instanceName, dbEngine, dbName,
 parameterGroup)
 scenario.DisplayConnection(instance)
 scenario.CreateSnapshot(ctx, instance)
 scenario.Cleanup(ctx, instance, parameterGroup)

 log.Println(strings.Repeat("-", 88))
 log.Println("Thanks for watching!")
 log.Println(strings.Repeat("-", 88))
}

// CreateParameterGroup shows how to get available engine versions for a
 specified
// database engine and create a DB parameter group that is compatible with a
// selected engine family.
func (scenario GetStartedInstances) CreateParameterGroup(ctx context.Context,
 dbEngine string,
 parameterGroupName string) *types.DBParameterGroup {

 log.Printf("Checking for an existing DB parameter group named %v.\n",
```

```
  parameterGroupName)
 parameterGroup, err := scenario.instances.GetParameterGroup(ctx,
 parameterGroupName)
 if err != nil {
  panic(err)
 }
 if parameterGroup == nil {
  log.Printf("Getting available database engine versions for %v.\n", dbEngine)
  engineVersions, err := scenario.instances.GetEngineVersions(ctx, dbEngine, "")
  if err != nil {
   panic(err)
  }

  familySet := map[string]struct{}{}
  for _, family := range engineVersions {
   familySet[*family.DBParameterGroupFamily] = struct{}{}
  }
  var families []string
  for family := range familySet {
   families = append(families, family)
  }
  sort.Strings(families)
  familyIndex := scenario.questioner.AskChoice("Which family do you want to use?
\n", families)
  log.Println("Creating a DB parameter group.")
  _, err = scenario.instances.CreateParameterGroup(
   ctx, parameterGroupName, families[familyIndex], "Example parameter group.")
  if err != nil {
   panic(err)
  }
  parameterGroup, err = scenario.instances.GetParameterGroup(ctx,
 parameterGroupName)
  if err != nil {
   panic(err)
  }
 }
 log.Printf("Parameter group %v:\n", *parameterGroup.DBParameterGroupFamily)
 log.Printf("\tName: %v\n", *parameterGroup.DBParameterGroupName)
 log.Printf("\tARN: %v\n", *parameterGroup.DBParameterGroupArn)
 log.Printf("\tFamily: %v\n", *parameterGroup.DBParameterGroupFamily)
 log.Printf("\tDescription: %v\n", *parameterGroup.Description)
 log.Println(strings.Repeat("-", 88))
 return parameterGroup
}
```

```
// SetUserParameters shows how to get the parameters contained in a custom
 parameter
// group and update some of the parameter values in the group.
func (scenario GetStartedInstances) SetUserParameters(ctx context.Context,
 parameterGroupName string) {
 log.Println("Let's set some parameter values in your parameter group.")
 dbParameters, err := scenario.instances.GetParameters(ctx, parameterGroupName,
 "")
 if err != nil {
  panic(err)
 }
 var updateParams []types.Parameter
 for _, dbParam := range dbParameters {
  if strings.HasPrefix(*dbParam.ParameterName, "auto_increment") &&
   *dbParam.IsModifiable && *dbParam.DataType == "integer" {
   log.Printf("The %v parameter is described as:\n\t%v",
    *dbParam.ParameterName, *dbParam.Description)
   rangeSplit := strings.Split(*dbParam.AllowedValues, "-")
   lower, _ := strconv.Atoi(rangeSplit[0])
   upper, _ := strconv.Atoi(rangeSplit[1])
   newValue := scenario.questioner.AskInt(
    fmt.Sprintf("Enter a value between %v and %v:", lower, upper),
    demotools.InIntRange{Lower: lower, Upper: upper})
   dbParam.ParameterValue = aws.String(strconv.Itoa(newValue))
   updateParams = append(updateParams, dbParam)
  }
 }
 err = scenario.instances.UpdateParameters(ctx, parameterGroupName, updateParams)
 if err != nil {
  panic(err)
 }
 log.Println("To get a list of parameters that you set previously, specify a
 source of 'user'.")
 userParameters, err := scenario.instances.GetParameters(ctx, parameterGroupName,
 "user")
 if err != nil {
  panic(err)
 }
 log.Println("Here are the parameters you set:")
 for _, param := range userParameters {
  log.Printf("\t%v: %v\n", *param.ParameterName, *param.ParameterValue)
 }
 log.Println(strings.Repeat("-", 88))
```

```
}

// CreateInstance shows how to create a DB instance that contains a database of a
// specified type. The database is also configured to use a custom DB parameter
 group.
func (scenario GetStartedInstances) CreateInstance(ctx context.Context,
 instanceName string, dbEngine string,
 dbName string, parameterGroup *types.DBParameterGroup) *types.DBInstance {

 log.Println("Checking for an existing DB instance.")
 instance, err := scenario.instances.GetInstance(ctx, instanceName)
 if err != nil {
  panic(err)
 }
 if instance == nil {
  adminUsername := scenario.questioner.Ask(
   "Enter an administrator username for the database: ", demotools.NotEmpty{})
  adminPassword := scenario.questioner.AskPassword(
   "Enter a password for the administrator (at least 8 characters): ", 7)
  engineVersions, err := scenario.instances.GetEngineVersions(ctx, dbEngine,
   *parameterGroup.DBParameterGroupFamily)
  if err != nil {
   panic(err)
  }
  var engineChoices []string
  for _, engine := range engineVersions {
   engineChoices = append(engineChoices, *engine.EngineVersion)
  }
  engineIndex := scenario.questioner.AskChoice(
   "The available engines for your parameter group are:\n", engineChoices)
  engineSelection := engineVersions[engineIndex]
  instOpts, err := scenario.instances.GetOrderableInstances(ctx,
 *engineSelection.Engine,
   *engineSelection.EngineVersion)
  if err != nil {
   panic(err)
  }
  optSet := map[string]struct{}{}
  for _, opt := range instOpts {
   if strings.Contains(*opt.DBInstanceClass, "micro") {
    optSet[*opt.DBInstanceClass] = struct{}{}
   }
  }
  var optChoices []string
```

```
  for opt := range optSet {
   optChoices = append(optChoices, opt)
  }
  sort.Strings(optChoices)
  optIndex := scenario.questioner.AskChoice(
   "The available micro DB instance classes for your database engine are:\n",
 optChoices)
  storageType := "standard"
  allocatedStorage := int32(5)
  log.Printf("Creating a DB instance named %v and database %v.\n"+
   "The DB instance is configured to use your custom parameter group %v,\n"+
   "selected engine %v,\n"+
   "selected DB instance class %v,"+
   "and %v GiB of %v storage.\n"+
   "This typically takes several minutes.",
   instanceName, dbName, *parameterGroup.DBParameterGroupName,
 *engineSelection.EngineVersion,
   optChoices[optIndex], allocatedStorage, storageType)
  instance, err = scenario.instances.CreateInstance(
   ctx, instanceName, dbName, *engineSelection.Engine,
 *engineSelection.EngineVersion,
   *parameterGroup.DBParameterGroupName, optChoices[optIndex], storageType,
   allocatedStorage, adminUsername, adminPassword)
  if err != nil {
   panic(err)
  }
  for *instance.DBInstanceStatus != "available" {
   scenario.helper.Pause(30)
   instance, err = scenario.instances.GetInstance(ctx, instanceName)
   if err != nil {
    panic(err)
   }
  }
  log.Println("Instance created and available.")
 }
 log.Println("Instance data:")
 log.Printf("\tDBInstanceIdentifier: %v\n", *instance.DBInstanceIdentifier)
 log.Printf("\tARN: %v\n", *instance.DBInstanceArn)
 log.Printf("\tStatus: %v\n", *instance.DBInstanceStatus)
 log.Printf("\tEngine: %v\n", *instance.Engine)
 log.Printf("\tEngine version: %v\n", *instance.EngineVersion)
 log.Println(strings.Repeat("-", 88))
 return instance
}
```

```
// DisplayConnection displays connection information about a DB instance and tips
// on how to connect to it.
func (scenario GetStartedInstances) DisplayConnection(instance *types.DBInstance)
 {
 log.Println(
  "You can now connect to your database by using your favorite MySQL client.\n" +
   "One way to connect is by using the 'mysql' shell on an Amazon EC2 instance\n"
 +
   "that is running in the same VPC as your DB instance. Pass the endpoint,\n" +
   "port, and administrator username to 'mysql'. Then, enter your password\n" +
   "when prompted:")
 log.Printf("\n\tmysql -h %v -P %v -u %v -p\n",
  *instance.Endpoint.Address, instance.Endpoint.Port, *instance.MasterUsername)
 log.Println("For more information, see the User Guide for RDS:\n" +
  "\thttps://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/
CHAP_GettingStarted.CreatingConnecting.MySQL.html#CHAP_GettingStarted.Connecting.MySQL")
 log.Println(strings.Repeat("-", 88))
}

// CreateSnapshot shows how to create a DB instance snapshot and wait until it's
 available.
func (scenario GetStartedInstances) CreateSnapshot(ctx context.Context, instance
 *types.DBInstance) {
 if scenario.questioner.AskBool(
  "Do you want to create a snapshot of your DB instance (y/n)? ", "y") {
  snapshotId := fmt.Sprintf("%v-%v", *instance.DBInstanceIdentifier,
 scenario.helper.UniqueId())
  log.Printf("Creating a snapshot named %v. This typically takes a few minutes.
\n", snapshotId)
  snapshot, err := scenario.instances.CreateSnapshot(ctx,
 *instance.DBInstanceIdentifier, snapshotId)
  if err != nil {
   panic(err)
  }
  for *snapshot.Status != "available" {
   scenario.helper.Pause(30)
   snapshot, err = scenario.instances.GetSnapshot(ctx, snapshotId)
   if err != nil {
    panic(err)
   }
  }
  log.Println("Snapshot data:")
  log.Printf("\tDBSnapshotIdentifier: %v\n", *snapshot.DBSnapshotIdentifier)
```

```
  log.Printf("\tARN: %v\n", *snapshot.DBSnapshotArn)
  log.Printf("\tStatus: %v\n", *snapshot.Status)
  log.Printf("\tEngine: %v\n", *snapshot.Engine)
  log.Printf("\tEngine version: %v\n", *snapshot.EngineVersion)
  log.Printf("\tDBInstanceIdentifier: %v\n", *snapshot.DBInstanceIdentifier)
  log.Printf("\tSnapshotCreateTime: %v\n", *snapshot.SnapshotCreateTime)
  log.Println(strings.Repeat("-", 88))
 }
}

// Cleanup shows how to clean up a DB instance and DB parameter group.
// Before the DB parameter group can be deleted, all associated DB instances must
 first be deleted.
func (scenario GetStartedInstances) Cleanup(
 ctx context.Context, instance *types.DBInstance, parameterGroup
 *types.DBParameterGroup) {

 if scenario.questioner.AskBool(
  "\nDo you want to delete the database instance and parameter group (y/n)? ",
 "y") {
  log.Printf("Deleting database instance %v.\n", *instance.DBInstanceIdentifier)
  err := scenario.instances.DeleteInstance(ctx, *instance.DBInstanceIdentifier)
  if err != nil {
   panic(err)
  }
  log.Println(
   "Waiting for the DB instance to delete. This typically takes several
 minutes.")
  for instance != nil {
   scenario.helper.Pause(30)
   instance, err = scenario.instances.GetInstance(ctx,
 *instance.DBInstanceIdentifier)
   if err != nil {
    panic(err)
   }
  }
  log.Printf("Deleting parameter group %v.",
 *parameterGroup.DBParameterGroupName)
  err = scenario.instances.DeleteParameterGroup(ctx,
 *parameterGroup.DBParameterGroupName)
  if err != nil {
   panic(err)
  }
 }
```

```
}

// IScenarioHelper abstracts the function from a scenario so that it
// can be mocked for unit testing.
type IScenarioHelper interface {
 Pause(secs int)
 UniqueId() string
}
type ScenarioHelper struct{}

// Pause waits for the specified number of seconds.
func (helper ScenarioHelper) Pause(secs int) {
 time.Sleep(time.Duration(secs) * time.Second)
}

// UniqueId returns a new UUID.
func (helper ScenarioHelper) UniqueId() string {
 return uuid.New().String()
}
```

定义场景调用以管理 Amazon RDS 操作的函数。

```
import (
 "context"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/rds"
 "github.com/aws/aws-sdk-go-v2/service/rds/types"
)

type DbInstances struct {
 RdsClient *rds.Client
}


// GetParameterGroup gets a DB parameter group by name.
func (instances *DbInstances) GetParameterGroup(ctx context.Context,
 parameterGroupName string) (
```

```
 *types.DBParameterGroup, error) {
 output, err := instances.RdsClient.DescribeDBParameterGroups(
  ctx, &rds.DescribeDBParameterGroupsInput{
   DBParameterGroupName: aws.String(parameterGroupName),
  })
 if err != nil {
  var notFoundError *types.DBParameterGroupNotFoundFault
  if errors.As(err, &notFoundError) {
   log.Printf("Parameter group %v does not exist.\n", parameterGroupName)
   err = nil
  } else {
   log.Printf("Error getting parameter group %v: %v\n", parameterGroupName, err)
  }
  return nil, err
 } else {
  return &output.DBParameterGroups[0], err
 }
}



// CreateParameterGroup creates a DB parameter group that is based on the
 specified
// parameter group family.
func (instances *DbInstances) CreateParameterGroup(
 ctx context.Context, parameterGroupName string, parameterGroupFamily string,
 description string) (
 *types.DBParameterGroup, error) {

 output, err := instances.RdsClient.CreateDBParameterGroup(ctx,
  &rds.CreateDBParameterGroupInput{
   DBParameterGroupName:   aws.String(parameterGroupName),
   DBParameterGroupFamily: aws.String(parameterGroupFamily),
   Description:            aws.String(description),
  })
 if err != nil {
  log.Printf("Couldn't create parameter group %v: %v\n", parameterGroupName, err)
  return nil, err
 } else {
  return output.DBParameterGroup, err
 }
}
```

```
// DeleteParameterGroup deletes the named DB parameter group.
func (instances *DbInstances) DeleteParameterGroup(ctx context.Context,
 parameterGroupName string) error {
 _, err := instances.RdsClient.DeleteDBParameterGroup(ctx,
  &rds.DeleteDBParameterGroupInput{
   DBParameterGroupName: aws.String(parameterGroupName),
  })
 if err != nil {
  log.Printf("Couldn't delete parameter group %v: %v\n", parameterGroupName, err)
  return err
 } else {
  return nil
 }
}



// GetParameters gets the parameters that are contained in a DB parameter group.
func (instances *DbInstances) GetParameters(ctx context.Context,
 parameterGroupName string, source string) (
 []types.Parameter, error) {

 var output *rds.DescribeDBParametersOutput
 var params []types.Parameter
 var err error
 parameterPaginator := rds.NewDescribeDBParametersPaginator(instances.RdsClient,
  &rds.DescribeDBParametersInput{
   DBParameterGroupName: aws.String(parameterGroupName),
   Source:               aws.String(source),
  })
 for parameterPaginator.HasMorePages() {
  output, err = parameterPaginator.NextPage(ctx)
  if err != nil {
   log.Printf("Couldn't get parameters for %v: %v\n", parameterGroupName, err)
   break
  } else {
   params = append(params, output.Parameters...)
  }
 }
 return params, err
}
```

```
// UpdateParameters updates parameters in a named DB parameter group.
func (instances *DbInstances) UpdateParameters(ctx context.Context,
 parameterGroupName string, params []types.Parameter) error {
 _, err := instances.RdsClient.ModifyDBParameterGroup(ctx,
  &rds.ModifyDBParameterGroupInput{
   DBParameterGroupName: aws.String(parameterGroupName),
   Parameters:           params,
  })
 if err != nil {
  log.Printf("Couldn't update parameters in %v: %v\n", parameterGroupName, err)
  return err
 } else {
  return nil
 }
}



// CreateSnapshot creates a snapshot of a DB instance.
func (instances *DbInstances) CreateSnapshot(ctx context.Context, instanceName
 string, snapshotName string) (
 *types.DBSnapshot, error) {
 output, err := instances.RdsClient.CreateDBSnapshot(ctx,
 &rds.CreateDBSnapshotInput{
  DBInstanceIdentifier: aws.String(instanceName),
  DBSnapshotIdentifier: aws.String(snapshotName),
 })
 if err != nil {
  log.Printf("Couldn't create snapshot %v: %v\n", snapshotName, err)
  return nil, err
 } else {
  return output.DBSnapshot, nil
 }
}



// GetSnapshot gets a DB instance snapshot.
func (instances *DbInstances) GetSnapshot(ctx context.Context, snapshotName
 string) (*types.DBSnapshot, error) {
 output, err := instances.RdsClient.DescribeDBSnapshots(ctx,
  &rds.DescribeDBSnapshotsInput{
   DBSnapshotIdentifier: aws.String(snapshotName),
```

```
  })
 if err != nil {
  log.Printf("Couldn't get snapshot %v: %v\n", snapshotName, err)
  return nil, err
 } else {
  return &output.DBSnapshots[0], nil
 }
}



// CreateInstance creates a DB instance.
func (instances *DbInstances) CreateInstance(ctx context.Context, instanceName
 string, dbName string,
 dbEngine string, dbEngineVersion string, parameterGroupName string,
 dbInstanceClass string,
 storageType string, allocatedStorage int32, adminName string, adminPassword
 string) (
 *types.DBInstance, error) {
 output, err := instances.RdsClient.CreateDBInstance(ctx,
 &rds.CreateDBInstanceInput{
  DBInstanceIdentifier: aws.String(instanceName),
  DBName:               aws.String(dbName),
  DBParameterGroupName: aws.String(parameterGroupName),
  Engine:               aws.String(dbEngine),
  EngineVersion:        aws.String(dbEngineVersion),
  DBInstanceClass:      aws.String(dbInstanceClass),
  StorageType:          aws.String(storageType),
  AllocatedStorage:     aws.Int32(allocatedStorage),
  MasterUsername:       aws.String(adminName),
  MasterUserPassword:   aws.String(adminPassword),
 })
 if err != nil {
  log.Printf("Couldn't create instance %v: %v\n", instanceName, err)
  return nil, err
 } else {
  return output.DBInstance, nil
 }
}



// GetInstance gets data about a DB instance.
```

```
func (instances *DbInstances) GetInstance(ctx context.Context, instanceName
 string) (
 *types.DBInstance, error) {
 output, err := instances.RdsClient.DescribeDBInstances(ctx,
  &rds.DescribeDBInstancesInput{
   DBInstanceIdentifier: aws.String(instanceName),
  })
 if err != nil {
  var notFoundError *types.DBInstanceNotFoundFault
  if errors.As(err, &notFoundError) {
   log.Printf("DB instance %v does not exist.\n", instanceName)
   err = nil
  } else {
   log.Printf("Couldn't get instance %v: %v\n", instanceName, err)
  }
  return nil, err
 } else {
  return &output.DBInstances[0], nil
 }
}



// DeleteInstance deletes a DB instance.
func (instances *DbInstances) DeleteInstance(ctx context.Context, instanceName
 string) error {
 _, err := instances.RdsClient.DeleteDBInstance(ctx, &rds.DeleteDBInstanceInput{
  DBInstanceIdentifier:   aws.String(instanceName),
  SkipFinalSnapshot:      aws.Bool(true),
  DeleteAutomatedBackups: aws.Bool(true),
 })
 if err != nil {
  log.Printf("Couldn't delete instance %v: %v\n", instanceName, err)
  return err
 } else {
  return nil
 }
}



// GetEngineVersions gets database engine versions that are available for the
 specified engine
// and parameter group family.
```

```
func (instances *DbInstances) GetEngineVersions(ctx context.Context, engine
 string, parameterGroupFamily string) (
 []types.DBEngineVersion, error) {
 output, err := instances.RdsClient.DescribeDBEngineVersions(ctx,
  &rds.DescribeDBEngineVersionsInput{
   Engine:                 aws.String(engine),
   DBParameterGroupFamily: aws.String(parameterGroupFamily),
  })
 if err != nil {
  log.Printf("Couldn't get engine versions for %v: %v\n", engine, err)
  return nil, err
 } else {
  return output.DBEngineVersions, nil
 }
}



// GetOrderableInstances uses a paginator to get DB instance options that can be
 used to create DB instances that are
// compatible with a set of specifications.
func (instances *DbInstances) GetOrderableInstances(ctx context.Context, engine
 string, engineVersion string) (
 []types.OrderableDBInstanceOption, error) {

 var output *rds.DescribeOrderableDBInstanceOptionsOutput
 var instanceOptions []types.OrderableDBInstanceOption
 var err error
 orderablePaginator :=
 rds.NewDescribeOrderableDBInstanceOptionsPaginator(instances.RdsClient,
  &rds.DescribeOrderableDBInstanceOptionsInput{
   Engine:        aws.String(engine),
   EngineVersion: aws.String(engineVersion),
  })
 for orderablePaginator.HasMorePages() {
  output, err = orderablePaginator.NextPage(ctx)
  if err != nil {
   log.Printf("Couldn't get orderable DB instance options: %v\n", err)
   break
  } else {
   instanceOptions = append(instanceOptions,
 output.OrderableDBInstanceOptions...)
  }
 }
```

```
  return instanceOptions, err
}
```

- 有关 API 详细信息，请参阅《适用于 Go 的 Amazon SDK API 参考》中的以下主题。
  - CreateDBInstance
  - CreateDBParameterGroup
  - CreateDBSnapshot
  - DeleteDBInstance
  - DeleteDBParameterGroup
  - DescribeDBEngineVersions
  - DescribeDBInstances
  - DescribeDBParameterGroups
  - DescribeDBParameters
  - DescribeDBSnapshots
  - DescribeOrderableDBInstanceOptions
  - ModifyDBParameterGroup

Java

适用于 Java 的 SDK 2.x

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

运行多个操作。

```
import com.google.gson.Gson;
import
 software.amazon.awssdk.auth.credentials.EnvironmentVariableCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.rds.RdsClient;
import software.amazon.awssdk.services.rds.model.CreateDbInstanceRequest;
```

```
import software.amazon.awssdk.services.rds.model.CreateDbInstanceResponse;
import software.amazon.awssdk.services.rds.model.CreateDbParameterGroupResponse;
import software.amazon.awssdk.services.rds.model.CreateDbSnapshotRequest;
import software.amazon.awssdk.services.rds.model.CreateDbSnapshotResponse;
import software.amazon.awssdk.services.rds.model.DBEngineVersion;
import software.amazon.awssdk.services.rds.model.DBInstance;
import software.amazon.awssdk.services.rds.model.DBParameterGroup;
import software.amazon.awssdk.services.rds.model.DBSnapshot;
import software.amazon.awssdk.services.rds.model.DeleteDbInstanceRequest;
import software.amazon.awssdk.services.rds.model.DeleteDbInstanceResponse;
import software.amazon.awssdk.services.rds.model.DescribeDbEngineVersionsRequest;
import
 software.amazon.awssdk.services.rds.model.DescribeDbEngineVersionsResponse;
import software.amazon.awssdk.services.rds.model.DescribeDbInstancesRequest;
import software.amazon.awssdk.services.rds.model.DescribeDbInstancesResponse;
import
 software.amazon.awssdk.services.rds.model.DescribeDbParameterGroupsResponse;
import software.amazon.awssdk.services.rds.model.DescribeDbParametersResponse;
import software.amazon.awssdk.services.rds.model.DescribeDbSnapshotsRequest;
import software.amazon.awssdk.services.rds.model.DescribeDbSnapshotsResponse;
import
 software.amazon.awssdk.services.rds.model.DescribeOrderableDbInstanceOptionsResponse;
import software.amazon.awssdk.services.rds.model.ModifyDbParameterGroupResponse;
import software.amazon.awssdk.services.rds.model.OrderableDBInstanceOption;
import software.amazon.awssdk.services.rds.model.Parameter;
import software.amazon.awssdk.services.rds.model.RdsException;
import software.amazon.awssdk.services.rds.model.CreateDbParameterGroupRequest;
import
 software.amazon.awssdk.services.rds.model.DescribeDbParameterGroupsRequest;
import software.amazon.awssdk.services.rds.model.DescribeDbParametersRequest;
import software.amazon.awssdk.services.rds.model.ModifyDbParameterGroupRequest;
import
 software.amazon.awssdk.services.rds.model.DescribeOrderableDbInstanceOptionsRequest;
import software.amazon.awssdk.services.rds.model.DeleteDbParameterGroupRequest;
import software.amazon.awssdk.services.secretsmanager.SecretsManagerClient;
import
 software.amazon.awssdk.services.secretsmanager.model.GetSecretValueRequest;
import
 software.amazon.awssdk.services.secretsmanager.model.GetSecretValueResponse;
import java.util.ArrayList;
import java.util.List;

/**
 * Before running this Java (v2) code example, set up your development
```

```
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 *
 * This example requires an AWS Secrets Manager secret that contains the
 * database credentials. If you do not create a
 * secret, this example will not work. For details, see:
 *
 * https://docs.aws.amazon.com/secretsmanager/latest/userguide/integrating_how-
services-use-secrets_RS.html
 *
 * This Java example performs these tasks:
 *
 * 1. Returns a list of the available DB engines.
 * 2. Selects an engine family and create a custom DB parameter group.
 * 3. Gets the parameter groups.
 * 4. Gets parameters in the group.
 * 5. Modifies the auto_increment_offset parameter.
 * 6. Gets and displays the updated parameters.
 * 7. Gets a list of allowed engine versions.
 * 8. Gets a list of micro instance classes available for the selected engine.
 * 9. Creates an RDS database instance that contains a MySql database and uses
 * the parameter group.
 * 10. Waits for the DB instance to be ready and prints out the connection
 * endpoint value.
 * 11. Creates a snapshot of the DB instance.
 * 12. Waits for an RDS DB snapshot to be ready.
 * 13. Deletes the RDS DB instance.
 * 14. Deletes the parameter group.
 */
public class RDSScenario {
    public static long sleepTime = 20;
    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");

    public static void main(String[] args) throws InterruptedException {
        final String usage = """

                Usage:
                    <dbGroupName> <dbParameterGroupFamily> <dbInstanceIdentifier>
 <dbName> <dbSnapshotIdentifier> <secretName>
```

```
                Where:
                    dbGroupName - The database group name.\s
                    dbParameterGroupFamily - The database parameter group name
(for example, mysql8.0).
                    dbInstanceIdentifier - The database instance identifier\s
                    dbName - The database name.\s
                    dbSnapshotIdentifier - The snapshot identifier.\s
                    secretName - The name of the AWS Secrets Manager secret that
contains the database credentials"
                """;

        if (args.length != 6) {
            System.out.println(usage);
            System.exit(1);
        }

        String dbGroupName = args[0];
        String dbParameterGroupFamily = args[1];
        String dbInstanceIdentifier = args[2];
        String dbName = args[3];
        String dbSnapshotIdentifier = args[4];
        String secretName = args[5];

        Gson gson = new Gson();
        User user = gson.fromJson(String.valueOf(getSecretValues(secretName)),
User.class);
        String masterUsername = user.getUsername();
        String masterUserPassword = user.getPassword();

        Region region = Region.US_WEST_2;
        RdsClient rdsClient = RdsClient.builder()
                .region(region)
                .build();
        System.out.println(DASHES);
        System.out.println("Welcome to the Amazon RDS example scenario.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("1. Return a list of the available DB engines");
        describeDBEngines(rdsClient);
        System.out.println(DASHES);

        System.out.println(DASHES);
```

```
        System.out.println("2. Create a custom parameter group");
        createDBParameterGroup(rdsClient, dbGroupName, dbParameterGroupFamily);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("3. Get the parameter group");
        describeDbParameterGroups(rdsClient, dbGroupName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("4. Get the parameters in the group");
        describeDbParameters(rdsClient, dbGroupName, 0);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("5. Modify the auto_increment_offset parameter");
        modifyDBParas(rdsClient, dbGroupName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("6. Display the updated value");
        describeDbParameters(rdsClient, dbGroupName, -1);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("7. Get a list of allowed engine versions");
        getAllowedEngines(rdsClient, dbParameterGroupFamily);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("8. Get a list of micro instance classes available for
 the selected engine");
        getMicroInstances(rdsClient);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println(
                "9. Create an RDS database instance that contains a MySql
 database and uses the parameter group");
        String dbARN = createDatabaseInstance(rdsClient, dbGroupName,
 dbInstanceIdentifier, dbName, masterUsername,
                masterUserPassword);
        System.out.println("The ARN of the new database is " + dbARN);
        System.out.println(DASHES);
```

```
        System.out.println(DASHES);
        System.out.println("10. Wait for DB instance to be ready");
        waitForInstanceReady(rdsClient, dbInstanceIdentifier);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("11. Create a snapshot of the DB instance");
        createSnapshot(rdsClient, dbInstanceIdentifier, dbSnapshotIdentifier);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("12. Wait for DB snapshot to be ready");
        waitForSnapshotReady(rdsClient, dbInstanceIdentifier,
 dbSnapshotIdentifier);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("13. Delete the DB instance");
        deleteDatabaseInstance(rdsClient, dbInstanceIdentifier);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("14. Delete the parameter group");
        deleteParaGroup(rdsClient, dbGroupName, dbARN);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("The Scenario has successfully completed.");
        System.out.println(DASHES);

        rdsClient.close();
    }

    private static SecretsManagerClient getSecretClient() {
        Region region = Region.US_WEST_2;
        return SecretsManagerClient.builder()
            .region(region)
            .build();
    }

    public static String getSecretValues(String secretName) {
        SecretsManagerClient secretClient = getSecretClient();
        GetSecretValueRequest valueRequest = GetSecretValueRequest.builder()
```

```
                .secretId(secretName)
                .build();

        GetSecretValueResponse valueResponse =
 secretClient.getSecretValue(valueRequest);
        return valueResponse.secretString();
    }

    // Delete the parameter group after database has been deleted.
    // An exception is thrown if you attempt to delete the para group while
 database
    // exists.
    public static void deleteParaGroup(RdsClient rdsClient, String dbGroupName,
 String dbARN)
            throws InterruptedException {
        try {
            boolean isDataDel = false;
            boolean didFind;
            String instanceARN;

            // Make sure that the database has been deleted.
            while (!isDataDel) {
                DescribeDbInstancesResponse response =
 rdsClient.describeDBInstances();
                List<DBInstance> instanceList = response.dbInstances();
                int listSize = instanceList.size();
                didFind = false;
                int index = 1;
                for (DBInstance instance : instanceList) {
                    instanceARN = instance.dbInstanceArn();
                    if (instanceARN.compareTo(dbARN) == 0) {
                        System.out.println(dbARN + " still exists");
                        didFind = true;
                    }
                    if ((index == listSize) && (!didFind)) {
                        // Went through the entire list and did not find the
 database ARN.
                        isDataDel = true;
                    }
                    Thread.sleep(sleepTime * 1000);
                    index++;
                }
            }
```

```
            // Delete the para group.
            DeleteDbParameterGroupRequest parameterGroupRequest =
DeleteDbParameterGroupRequest.builder()
                    .dbParameterGroupName(dbGroupName)
                    .build();

            rdsClient.deleteDBParameterGroup(parameterGroupRequest);
            System.out.println(dbGroupName + " was deleted.");

        } catch (RdsException e) {
            System.out.println(e.getLocalizedMessage());
            System.exit(1);
        }
    }

    // Delete the DB instance.
    public static void deleteDatabaseInstance(RdsClient rdsClient, String
dbInstanceIdentifier) {
        try {
            DeleteDbInstanceRequest deleteDbInstanceRequest =
DeleteDbInstanceRequest.builder()
                    .dbInstanceIdentifier(dbInstanceIdentifier)
                    .deleteAutomatedBackups(true)
                    .skipFinalSnapshot(true)
                    .build();

            DeleteDbInstanceResponse response =
rdsClient.deleteDBInstance(deleteDbInstanceRequest);
            System.out.print("The status of the database is " +
response.dbInstance().dbInstanceStatus());

        } catch (RdsException e) {
            System.out.println(e.getLocalizedMessage());
            System.exit(1);
        }
    }

    // Waits until the snapshot instance is available.
    public static void waitForSnapshotReady(RdsClient rdsClient, String
dbInstanceIdentifier,
            String dbSnapshotIdentifier) {
        try {
            boolean snapshotReady = false;
            String snapshotReadyStr;
```

```
            System.out.println("Waiting for the snapshot to become available.");

            DescribeDbSnapshotsRequest snapshotsRequest =
DescribeDbSnapshotsRequest.builder()
                    .dbSnapshotIdentifier(dbSnapshotIdentifier)
                    .dbInstanceIdentifier(dbInstanceIdentifier)
                    .build();

            while (!snapshotReady) {
                DescribeDbSnapshotsResponse response =
rdsClient.describeDBSnapshots(snapshotsRequest);
                List<DBSnapshot> snapshotList = response.dbSnapshots();
                for (DBSnapshot snapshot : snapshotList) {
                    snapshotReadyStr = snapshot.status();
                    if (snapshotReadyStr.contains("available")) {
                        snapshotReady = true;
                    } else {
                        System.out.print(".");
                        Thread.sleep(sleepTime * 1000);
                    }
                }
            }

            System.out.println("The Snapshot is available!");
        } catch (RdsException | InterruptedException e) {
            System.out.println(e.getLocalizedMessage());
            System.exit(1);
        }
    }

    // Create an Amazon RDS snapshot.
    public static void createSnapshot(RdsClient rdsClient, String
dbInstanceIdentifier, String dbSnapshotIdentifier) {
        try {
            CreateDbSnapshotRequest snapshotRequest =
CreateDbSnapshotRequest.builder()
                    .dbInstanceIdentifier(dbInstanceIdentifier)
                    .dbSnapshotIdentifier(dbSnapshotIdentifier)
                    .build();

            CreateDbSnapshotResponse response =
rdsClient.createDBSnapshot(snapshotRequest);
            System.out.println("The Snapshot id is " +
response.dbSnapshot().dbiResourceId());
```

```
        } catch (RdsException e) {
            System.out.println(e.getLocalizedMessage());
            System.exit(1);
        }
    }

    // Waits until the database instance is available.
    public static void waitForInstanceReady(RdsClient rdsClient, String
 dbInstanceIdentifier) {
        boolean instanceReady = false;
        String instanceReadyStr;
        System.out.println("Waiting for instance to become available.");
        try {
            DescribeDbInstancesRequest instanceRequest =
 DescribeDbInstancesRequest.builder()
                    .dbInstanceIdentifier(dbInstanceIdentifier)
                    .build();

            String endpoint = "";
            while (!instanceReady) {
                DescribeDbInstancesResponse response =
 rdsClient.describeDBInstances(instanceRequest);
                List<DBInstance> instanceList = response.dbInstances();
                for (DBInstance instance : instanceList) {
                    instanceReadyStr = instance.dbInstanceStatus();
                    if (instanceReadyStr.contains("available")) {
                        endpoint = instance.endpoint().address();
                        instanceReady = true;
                    } else {
                        System.out.print(".");
                        Thread.sleep(sleepTime * 1000);
                    }
                }
            }
            System.out.println("Database instance is available! The connection
 endpoint is " + endpoint);

        } catch (RdsException | InterruptedException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
```

```
    // Create a database instance and return the ARN of the database.
    public static String createDatabaseInstance(RdsClient rdsClient,
            String dbGroupName,
            String dbInstanceIdentifier,
            String dbName,
            String userName,
            String userPassword) {

        try {
            CreateDbInstanceRequest instanceRequest =
 CreateDbInstanceRequest.builder()
                .dbInstanceIdentifier(dbInstanceIdentifier)
                .allocatedStorage(100)
                .dbName(dbName)
                .engine("mysql")
                .dbInstanceClass("db.t3.medium") // Updated to a supported class
                .engineVersion("8.0.32")         // Updated to a supported
 version
                .storageType("gp2")             // Changed to General Purpose SSD
 (gp2)
                .masterUsername(userName)
                .masterUserPassword(userPassword)
                .build();

            CreateDbInstanceResponse response =
 rdsClient.createDBInstance(instanceRequest);
            System.out.print("The status is " +
 response.dbInstance().dbInstanceStatus());
            return response.dbInstance().dbInstanceArn();

        } catch (RdsException e) {
            System.out.println(e.getLocalizedMessage());
            System.exit(1);
        }

        return "";
    }

    // Get a list of micro instances.
    public static void getMicroInstances(RdsClient rdsClient) {
        try {
            DescribeOrderableDbInstanceOptionsRequest dbInstanceOptionsRequest =
 DescribeOrderableDbInstanceOptionsRequest
                    .builder()
```

```
                    .engine("mysql")
                    .build();

            DescribeOrderableDbInstanceOptionsResponse response = rdsClient

.describeOrderableDBInstanceOptions(dbInstanceOptionsRequest);
            List<OrderableDBInstanceOption> orderableDBInstances =
response.orderableDBInstanceOptions();
            for (OrderableDBInstanceOption dbInstanceOption :
orderableDBInstances) {
                System.out.println("The engine version is " +
dbInstanceOption.engineVersion());
                System.out.println("The engine description is " +
dbInstanceOption.engine());
            }

        } catch (RdsException e) {
            System.out.println(e.getLocalizedMessage());
            System.exit(1);
        }
    }

    // Get a list of allowed engine versions.
    public static void getAllowedEngines(RdsClient rdsClient, String
dbParameterGroupFamily) {
        try {
            DescribeDbEngineVersionsRequest versionsRequest =
DescribeDbEngineVersionsRequest.builder()
                    .dbParameterGroupFamily(dbParameterGroupFamily)
                    .engine("mysql")
                    .build();

            DescribeDbEngineVersionsResponse response =
rdsClient.describeDBEngineVersions(versionsRequest);
            List<DBEngineVersion> dbEngines = response.dbEngineVersions();
            for (DBEngineVersion dbEngine : dbEngines) {
                System.out.println("The engine version is " +
dbEngine.engineVersion());
                System.out.println("The engine description is " +
dbEngine.dbEngineDescription());
            }

        } catch (RdsException e) {
            System.out.println(e.getLocalizedMessage());
```

```
            System.exit(1);
        }
    }

    // Modify auto_increment_offset and auto_increment_increment parameters.
    public static void modifyDBParas(RdsClient rdsClient, String dbGroupName) {
        try {
            Parameter parameter1 = Parameter.builder()
                    .parameterName("auto_increment_offset")
                    .applyMethod("immediate")
                    .parameterValue("5")
                    .build();

            List<Parameter> paraList = new ArrayList<>();
            paraList.add(parameter1);
            ModifyDbParameterGroupRequest groupRequest =
 ModifyDbParameterGroupRequest.builder()
                    .dbParameterGroupName(dbGroupName)
                    .parameters(paraList)
                    .build();

            ModifyDbParameterGroupResponse response =
 rdsClient.modifyDBParameterGroup(groupRequest);
            System.out.println("The parameter group " +
 response.dbParameterGroupName() + " was successfully modified");

        } catch (RdsException e) {
            System.out.println(e.getLocalizedMessage());
            System.exit(1);
        }
    }

    // Retrieve parameters in the group.
    public static void describeDbParameters(RdsClient rdsClient, String
 dbGroupName, int flag) {
        try {
            DescribeDbParametersRequest dbParameterGroupsRequest;
            if (flag == 0) {
                dbParameterGroupsRequest = DescribeDbParametersRequest.builder()
                        .dbParameterGroupName(dbGroupName)
                        .build();
            } else {
                dbParameterGroupsRequest = DescribeDbParametersRequest.builder()
                        .dbParameterGroupName(dbGroupName)
```

```
                        .source("user")
                        .build();
            }

            DescribeDbParametersResponse response =
rdsClient.describeDBParameters(dbParameterGroupsRequest);
            List<Parameter> dbParameters = response.parameters();
            String paraName;
            for (Parameter para : dbParameters) {
                // Only print out information about either auto_increment_offset
or
                // auto_increment_increment.
                paraName = para.parameterName();
                if ((paraName.compareTo("auto_increment_offset") == 0)
                        || (paraName.compareTo("auto_increment_increment ") ==
0)) {
                    System.out.println("*** The parameter name is  " + paraName);
                    System.out.println("*** The parameter value is  " +
para.parameterValue());
                    System.out.println("*** The parameter data type is " +
para.dataType());
                    System.out.println("*** The parameter description is " +
para.description());
                    System.out.println("*** The parameter allowed values  is " +
para.allowedValues());
                }
            }

        } catch (RdsException e) {
            System.out.println(e.getLocalizedMessage());
            System.exit(1);
        }
    }

    public static void describeDbParameterGroups(RdsClient rdsClient, String
dbGroupName) {
        try {
            DescribeDbParameterGroupsRequest groupsRequest =
DescribeDbParameterGroupsRequest.builder()
                    .dbParameterGroupName(dbGroupName)
                    .maxRecords(20)
                    .build();
```

```
            DescribeDbParameterGroupsResponse response =
rdsClient.describeDBParameterGroups(groupsRequest);
            List<DBParameterGroup> groups = response.dbParameterGroups();
            for (DBParameterGroup group : groups) {
                System.out.println("The group name is " +
group.dbParameterGroupName());
                System.out.println("The group description is " +
group.description());
            }

        } catch (RdsException e) {
            System.out.println(e.getLocalizedMessage());
            System.exit(1);
        }
    }

    public static void createDBParameterGroup(RdsClient rdsClient, String
dbGroupName, String dbParameterGroupFamily) {
        try {
            CreateDbParameterGroupRequest groupRequest =
CreateDbParameterGroupRequest.builder()
                    .dbParameterGroupName(dbGroupName)
                    .dbParameterGroupFamily(dbParameterGroupFamily)
                    .description("Created by using the AWS SDK for Java")
                    .build();

            CreateDbParameterGroupResponse response =
rdsClient.createDBParameterGroup(groupRequest);
            System.out.println("The group name is " +
response.dbParameterGroup().dbParameterGroupName());

        } catch (RdsException e) {
            System.out.println(e.getLocalizedMessage());
            System.exit(1);
        }
    }

    public static void describeDBEngines(RdsClient rdsClient) {
        try {
            DescribeDbEngineVersionsRequest engineVersionsRequest =
DescribeDbEngineVersionsRequest.builder()
                    .defaultOnly(true)
                    .engine("mysql")
                    .maxRecords(20)
```

```
                    .build();

            DescribeDbEngineVersionsResponse response =
 rdsClient.describeDBEngineVersions(engineVersionsRequest);
            List<DBEngineVersion> engines = response.dbEngineVersions();

            // Get all DBEngineVersion objects.
            for (DBEngineVersion engineOb : engines) {
                System.out.println("The name of the DB parameter group family for
 the database engine is "
                        + engineOb.dbParameterGroupFamily());
                System.out.println("The name of the database engine " +
 engineOb.engine());
                System.out.println("The version number of the database engine " +
 engineOb.engineVersion());
            }

        } catch (RdsException e) {
            System.out.println(e.getLocalizedMessage());
            System.exit(1);
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon SDK for Java 2.x API 参考》中的以下主题。
  - CreateDBInstance
  - CreateDBParameterGroup
  - CreateDBSnapshot
  - DeleteDBInstance
  - DeleteDBParameterGroup
  - DescribeDBEngineVersions
  - DescribeDBInstances
  - DescribeDBParameterGroups
  - DescribeDBParameters
  - DescribeDBSnapshots
  - DescribeOrderableDBInstanceOptions
  - ModifyDBParameterGroup

Kotlin

适用于 Kotlin 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
/**
Before running this code example, set up your development environment, including
 your credentials.

For more information, see the following documentation topic:

https://docs.aws.amazon.com/sdk-for-kotlin/latest/developer-guide/setup.html

This example requires an AWS Secrets Manager secret that contains the database
 credentials. If you do not create a
secret, this example will not work. For more details, see:

https://docs.aws.amazon.com/secretsmanager/latest/userguide/integrating_how-
services-use-secrets_RS.html

This example performs the following tasks:

1. Returns a list of the available DB engines by invoking the
 DescribeDbEngineVersions method.
2. Selects an engine family and create a custom DB parameter group by invoking
 the createDBParameterGroup method.
3. Gets the parameter groups by invoking the DescribeDbParameterGroups method.
4. Gets parameters in the group by invoking the DescribeDbParameters method.
5. Modifies both the auto_increment_offset and auto_increment_increment
 parameters by invoking the modifyDbParameterGroup method.
6. Gets and displays the updated parameters.
7. Gets a list of allowed engine versions by invoking the
 describeDbEngineVersions method.
8. Gets a list of micro instance classes available for the selected engine.
9. Creates an Amazon Relational Database Service (Amazon RDS) database instance
 that contains a MySQL database and uses the parameter group.
```

```
10. Waits for DB instance to be ready and prints out the connection endpoint
 value.
11. Creates a snapshot of the DB instance.
12. Waits for the DB snapshot to be ready.
13. Deletes the DB instance.
14. Deletes the parameter group.
 */

var sleepTime: Long = 20

suspend fun main(args: Array<String>) {
    val usage = """
        Usage:
            <dbGroupName> <dbParameterGroupFamily> <dbInstanceIdentifier>
 <dbName> <dbSnapshotIdentifier><secretName>

        Where:
            dbGroupName - The database group name.
            dbParameterGroupFamily - The database parameter group name.
            dbInstanceIdentifier - The database instance identifier.
            dbName -  The database name.
            dbSnapshotIdentifier - The snapshot identifier.
            secretName - The name of the AWS Secrets Manager secret that contains
 the database credentials.
    """

    if (args.size != 6) {
        println(usage)
        exitProcess(1)
    }

    val dbGroupName = args[0]
    val dbParameterGroupFamily = args[1]
    val dbInstanceIdentifier = args[2]
    val dbName = args[3]
    val dbSnapshotIdentifier = args[4]
    val secretName = args[5]

    val gson = Gson()
    val user = gson.fromJson(getSecretValues(secretName).toString(),
 User::class.java)
    val username = user.username
    val userPassword = user.password
```

```
    println("1. Return a list of the available DB engines")
    describeDBEngines()

    println("2. Create a custom parameter group")
    createDBParameterGroup(dbGroupName, dbParameterGroupFamily)

    println("3. Get the parameter groups")
    describeDbParameterGroups(dbGroupName)

    println("4. Get the parameters in the group")
    describeDbParameters(dbGroupName, 0)

    println("5. Modify the auto_increment_offset parameter")
    modifyDBParas(dbGroupName)

    println("6. Display the updated value")
    describeDbParameters(dbGroupName, -1)

    println("7. Get a list of allowed engine versions")
    getAllowedEngines(dbParameterGroupFamily)

    println("8. Get a list of micro instance classes available for the selected
 engine")
    getMicroInstances()

    println("9. Create an RDS database instance that contains a MySql database
 and uses the parameter group")
    val dbARN = createDatabaseInstance(dbGroupName, dbInstanceIdentifier, dbName,
 username, userPassword)
    println("The ARN of the new database is $dbARN")

    println("10. Wait for DB instance to be ready")
    waitForDbInstanceReady(dbInstanceIdentifier)

    println("11. Create a snapshot of the DB instance")
    createDbSnapshot(dbInstanceIdentifier, dbSnapshotIdentifier)

    println("12. Wait for DB snapshot to be ready")
    waitForSnapshotReady(dbInstanceIdentifier, dbSnapshotIdentifier)

    println("13. Delete the DB instance")
    deleteDbInstance(dbInstanceIdentifier)

    println("14. Delete the parameter group")
```

```
    if (dbARN != null) {
        deleteParaGroup(dbGroupName, dbARN)
    }

    println("The Scenario has successfully completed.")
}

suspend fun deleteParaGroup(
    dbGroupName: String,
    dbARN: String,
) {
    var isDataDel = false
    var didFind: Boolean
    var instanceARN: String

    RdsClient { region = "us-west-2" }.use { rdsClient ->
        // Make sure that the database has been deleted.
        while (!isDataDel) {
            val response = rdsClient.describeDbInstances()
            val instanceList = response.dbInstances
            val listSize = instanceList?.size
            isDataDel = false // Reset this value.
            didFind = false // Reset this value.
            var index = 1
            if (instanceList != null) {
                for (instance in instanceList) {
                    instanceARN = instance.dbInstanceArn.toString()
                    if (instanceARN.compareTo(dbARN) == 0) {
                        println("$dbARN still exists")
                        didFind = true
                    }
                    if (index == listSize && !didFind) {
                        // Went through the entire list and did not find the
 database name.
                        isDataDel = true
                    }
                    index++
                }
            }
        }

        // Delete the para group.
        val parameterGroupRequest =
            DeleteDbParameterGroupRequest {
```

```
                dbParameterGroupName = dbGroupName
            }
        rdsClient.deleteDbParameterGroup(parameterGroupRequest)
        println("$dbGroupName was deleted.")
    }
}

suspend fun deleteDbInstance(dbInstanceIdentifierVal: String) {
    val deleteDbInstanceRequest =
        DeleteDbInstanceRequest {
            dbInstanceIdentifier = dbInstanceIdentifierVal
            deleteAutomatedBackups = true
            skipFinalSnapshot = true
        }

    RdsClient { region = "us-west-2" }.use { rdsClient ->
        val response = rdsClient.deleteDbInstance(deleteDbInstanceRequest)
        print("The status of the database is
 ${response.dbInstance?.dbInstanceStatus}")
    }
}

// Waits until the snapshot instance is available.
suspend fun waitForSnapshotReady(
    dbInstanceIdentifierVal: String?,
    dbSnapshotIdentifierVal: String?,
) {
    var snapshotReady = false
    var snapshotReadyStr: String
    println("Waiting for the snapshot to become available.")

    val snapshotsRequest =
        DescribeDbSnapshotsRequest {
            dbSnapshotIdentifier = dbSnapshotIdentifierVal
            dbInstanceIdentifier = dbInstanceIdentifierVal
        }

    while (!snapshotReady) {
        RdsClient { region = "us-west-2" }.use { rdsClient ->
            val response = rdsClient.describeDbSnapshots(snapshotsRequest)
            val snapshotList: List<DbSnapshot>? = response.dbSnapshots
            if (snapshotList != null) {
                for (snapshot in snapshotList) {
                    snapshotReadyStr = snapshot.status.toString()
```

```
                    if (snapshotReadyStr.contains("available")) {
                        snapshotReady = true
                    } else {
                        print(".")
                        delay(sleepTime * 1000)
                    }
                }
            }
        }
    }
    println("The Snapshot is available!")
}

// Create an Amazon RDS snapshot.
suspend fun createDbSnapshot(
    dbInstanceIdentifierVal: String?,
    dbSnapshotIdentifierVal: String?,
) {
    val snapshotRequest =
        CreateDbSnapshotRequest {
            dbInstanceIdentifier = dbInstanceIdentifierVal
            dbSnapshotIdentifier = dbSnapshotIdentifierVal
        }

    RdsClient { region = "us-west-2" }.use { rdsClient ->
        val response = rdsClient.createDbSnapshot(snapshotRequest)
        print("The Snapshot id is ${response.dbSnapshot?.dbiResourceId}")
    }
}

// Waits until the database instance is available.
suspend fun waitForDbInstanceReady(dbInstanceIdentifierVal: String?) {
    var instanceReady = false
    var instanceReadyStr: String
    println("Waiting for instance to become available.")

    val instanceRequest =
        DescribeDbInstancesRequest {
            dbInstanceIdentifier = dbInstanceIdentifierVal
        }
    var endpoint = ""
    while (!instanceReady) {
        RdsClient { region = "us-west-2" }.use { rdsClient ->
            val response = rdsClient.describeDbInstances(instanceRequest)
```

```
            val instanceList = response.dbInstances
            if (instanceList != null) {
                for (instance in instanceList) {
                    instanceReadyStr = instance.dbInstanceStatus.toString()
                    if (instanceReadyStr.contains("available")) {
                        endpoint = instance.endpoint?.address.toString()
                        instanceReady = true
                    } else {
                        print(".")
                        delay(sleepTime * 1000)
                    }
                }
            }
        }
    }
    println("Database instance is available! The connection endpoint is
 $endpoint")
}

// Create a database instance and return the ARN of the database.
suspend fun createDatabaseInstance(
    dbGroupNameVal: String?,
    dbInstanceIdentifierVal: String?,
    dbNameVal: String?,
    masterUsernameVal: String?,
    masterUserPasswordVal: String?,
): String? {
    val instanceRequest =
        CreateDbInstanceRequest {
            dbInstanceIdentifier = dbInstanceIdentifierVal
            allocatedStorage = 100
            dbName = dbNameVal
            dbParameterGroupName = dbGroupNameVal
            engine = "mysql"
            dbInstanceClass = "db.t3.micro"
            engineVersion = "8.0.35"
            storageType = "gp2"
            masterUsername = masterUsernameVal
            masterUserPassword = masterUserPasswordVal
        }

    RdsClient { region = "us-west-2" }.use { rdsClient ->
        val response = rdsClient.createDbInstance(instanceRequest)
        print("The status is ${response.dbInstance?.dbInstanceStatus}")
```

```
        return response.dbInstance?.dbInstanceArn
    }
}

// Get a list of micro instances.
suspend fun getMicroInstances() {
    val dbInstanceOptionsRequest =
        DescribeOrderableDbInstanceOptionsRequest {
            engine = "mysql"
        }
    RdsClient { region = "us-west-2" }.use { rdsClient ->
        val response =
 rdsClient.describeOrderableDbInstanceOptions(dbInstanceOptionsRequest)
        val orderableDBInstances = response.orderableDbInstanceOptions
        if (orderableDBInstances != null) {
            for (dbInstanceOption in orderableDBInstances) {
                println("The engine version is
 ${dbInstanceOption.engineVersion}")
                println("The engine description is ${dbInstanceOption.engine}")
            }
        }
    }
}

// Get a list of allowed engine versions.
suspend fun getAllowedEngines(dbParameterGroupFamilyVal: String?) {
    val versionsRequest =
        DescribeDbEngineVersionsRequest {
            dbParameterGroupFamily = dbParameterGroupFamilyVal
            engine = "mysql"
        }
    RdsClient { region = "us-west-2" }.use { rdsClient ->
        val response = rdsClient.describeDbEngineVersions(versionsRequest)
        val dbEngines: List<DbEngineVersion>? = response.dbEngineVersions
        if (dbEngines != null) {
            for (dbEngine in dbEngines) {
                println("The engine version is ${dbEngine.engineVersion}")
                println("The engine description is
 ${dbEngine.dbEngineDescription}")
            }
        }
    }
}
```

```
// Modify the auto_increment_offset parameter.
suspend fun modifyDBParas(dbGroupName: String) {
    val parameter1 =
        Parameter {
            parameterName = "auto_increment_offset"
            applyMethod = ApplyMethod.Immediate
            parameterValue = "5"
        }

    val paraList: ArrayList<Parameter> = ArrayList()
    paraList.add(parameter1)
    val groupRequest =
        ModifyDbParameterGroupRequest {
            dbParameterGroupName = dbGroupName
            parameters = paraList
        }

    RdsClient { region = "us-west-2" }.use { rdsClient ->
        val response = rdsClient.modifyDbParameterGroup(groupRequest)
        println("The parameter group ${response.dbParameterGroupName} was
 successfully modified")
    }
}

// Retrieve parameters in the group.
suspend fun describeDbParameters(
    dbGroupName: String?,
    flag: Int,
) {
    val dbParameterGroupsRequest: DescribeDbParametersRequest
    dbParameterGroupsRequest =
        if (flag == 0) {
            DescribeDbParametersRequest {
                dbParameterGroupName = dbGroupName
            }
        } else {
            DescribeDbParametersRequest {
                dbParameterGroupName = dbGroupName
                source = "user"
            }
        }
    RdsClient { region = "us-west-2" }.use { rdsClient ->
        val response = rdsClient.describeDbParameters(dbParameterGroupsRequest)
        val dbParameters: List<Parameter>? = response.parameters
```

```
        var paraName: String
        if (dbParameters != null) {
            for (para in dbParameters) {
                // Only print out information about either auto_increment_offset
 or auto_increment_increment.
                paraName = para.parameterName.toString()
                if (paraName.compareTo("auto_increment_offset") == 0 ||
 paraName.compareTo("auto_increment_increment ") == 0) {
                    println("*** The parameter name is  $paraName")
                    System.out.println("*** The parameter value is
 ${para.parameterValue}")
                    System.out.println("*** The parameter data type is
 ${para.dataType}")
                    System.out.println("*** The parameter description is
 ${para.description}")
                    System.out.println("*** The parameter allowed values  is
 ${para.allowedValues}")
                }
            }
        }
    }
}

suspend fun describeDbParameterGroups(dbGroupName: String?) {
    val groupsRequest =
        DescribeDbParameterGroupsRequest {
            dbParameterGroupName = dbGroupName
            maxRecords = 20
        }
    RdsClient { region = "us-west-2" }.use { rdsClient ->
        val response = rdsClient.describeDbParameterGroups(groupsRequest)
        val groups = response.dbParameterGroups
        if (groups != null) {
            for (group in groups) {
                println("The group name is ${group.dbParameterGroupName}")
                println("The group description is ${group.description}")
            }
        }
    }
}

// Create a parameter group.
suspend fun createDBParameterGroup(
    dbGroupName: String?,
```

```
    dbParameterGroupFamilyVal: String?,
) {
    val groupRequest =
        CreateDbParameterGroupRequest {
            dbParameterGroupName = dbGroupName
            dbParameterGroupFamily = dbParameterGroupFamilyVal
            description = "Created by using the AWS SDK for Kotlin"
        }

    RdsClient { region = "us-west-2" }.use { rdsClient ->
        val response = rdsClient.createDbParameterGroup(groupRequest)
        println("The group name is
 ${response.dbParameterGroup?.dbParameterGroupName}")
    }
}

// Returns a list of the available DB engines.
suspend fun describeDBEngines() {
    val engineVersionsRequest =
        DescribeDbEngineVersionsRequest {
            defaultOnly = true
            engine = "mysql"
            maxRecords = 20
        }

    RdsClient { region = "us-west-2" }.use { rdsClient ->
        val response = rdsClient.describeDbEngineVersions(engineVersionsRequest)
        val engines: List<DbEngineVersion>? = response.dbEngineVersions

        // Get all DbEngineVersion objects.
        if (engines != null) {
            for (engineOb in engines) {
                println("The name of the DB parameter group family for the
 database engine is ${engineOb.dbParameterGroupFamily}.")
                println("The name of the database engine ${engineOb.engine}.")
                println("The version number of the database engine
 ${engineOb.engineVersion}")
            }
        }
    }
}

suspend fun getSecretValues(secretName: String?): String? {
    val valueRequest =
```

```
        GetSecretValueRequest {
            secretId = secretName
        }

    SecretsManagerClient { region = "us-west-2" }.use { secretsClient ->
        val valueResponse = secretsClient.getSecretValue(valueRequest)
        return valueResponse.secretString
    }
}
```

- 有关 API 详细信息，请参阅《Amazon SDK for Kotlin API 参考》中的以下主题。
  - CreateDBInstance
  - CreateDBParameterGroup
  - CreateDBSnapshot
  - DeleteDBInstance
  - DeleteDBParameterGroup
  - DescribeDBEngineVersions
  - DescribeDBInstances
  - DescribeDBParameterGroups
  - DescribeDBParameters
  - DescribeDBSnapshots
  - DescribeOrderableDBInstanceOptions
  - ModifyDBParameterGroup

Python

适用于 Python 的 SDK（Boto3）

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

在命令提示符中运行交互式场景。

```
class RdsInstanceScenario:
    """Runs a scenario that shows how to get started using Amazon RDS DB
 instances."""

    def __init__(self, instance_wrapper):
        """
        :param instance_wrapper: An object that wraps Amazon RDS DB instance
 actions.
        """
        self.instance_wrapper = instance_wrapper

    def create_parameter_group(self, parameter_group_name, db_engine):
        """
        Shows how to get available engine versions for a specified database
 engine and
        create a DB parameter group that is compatible with a selected engine
 family.

        :param parameter_group_name: The name given to the newly created
 parameter group.
        :param db_engine: The database engine to use as a basis.
        :return: The newly created parameter group.
        """
        print(
            f"Checking for an existing DB instance parameter group named
 {parameter_group_name}."
        )
        parameter_group = self.instance_wrapper.get_parameter_group(
            parameter_group_name
        )
        if parameter_group is None:
            print(f"Getting available database engine versions for {db_engine}.")
            engine_versions =
 self.instance_wrapper.get_engine_versions(db_engine)
            families = list({ver["DBParameterGroupFamily"] for ver in
 engine_versions})
            family_index = q.choose("Which family do you want to use? ",
 families)
            print(f"Creating a parameter group.")
            self.instance_wrapper.create_parameter_group(
                parameter_group_name, families[family_index], "Example parameter
 group."
            )
```

```
            parameter_group = self.instance_wrapper.get_parameter_group(
                parameter_group_name
            )
        print(f"Parameter group {parameter_group['DBParameterGroupName']}:")
        pp(parameter_group)
        print("-" * 88)
        return parameter_group

    def update_parameters(self, parameter_group_name):
        """
        Shows how to get the parameters contained in a custom parameter group and
        update some of the parameter values in the group.

        :param parameter_group_name: The name of the parameter group to query and
 modify.
        """
        print("Let's set some parameter values in your parameter group.")
        auto_inc_parameters = self.instance_wrapper.get_parameters(
            parameter_group_name, name_prefix="auto_increment"
        )
        update_params = []
        for auto_inc in auto_inc_parameters:
            if auto_inc["IsModifiable"] and auto_inc["DataType"] == "integer":
                print(f"The {auto_inc['ParameterName']} parameter is described
 as:")
                print(f"\t{auto_inc['Description']}")
                param_range = auto_inc["AllowedValues"].split("-")
                auto_inc["ParameterValue"] = str(
                    q.ask(
                        f"Enter a value between {param_range[0]} and
 {param_range[1]}: ",
                        q.is_int,
                        q.in_range(int(param_range[0]), int(param_range[1])),
                    )
                )
                update_params.append(auto_inc)
        self.instance_wrapper.update_parameters(parameter_group_name,
 update_params)
        print(
            "You can get a list of parameters you've set by specifying a source
 of 'user'."
        )
        user_parameters = self.instance_wrapper.get_parameters(
            parameter_group_name, source="user"
```

```
        )
        pp(user_parameters)
        print("-" * 88)

    def create_instance(self, instance_name, db_name, db_engine,
parameter_group):
        """
        Shows how to create a DB instance that contains a database of a specified
        type and is configured to use a custom DB parameter group.

        :param instance_name: The name given to the newly created DB instance.
        :param db_name: The name given to the created database.
        :param db_engine: The engine of the created database.
        :param parameter_group: The parameter group that is associated with the
DB instance.
        :return: The newly created DB instance.
        """
        print("Checking for an existing DB instance.")
        db_inst = self.instance_wrapper.get_db_instance(instance_name)
        if db_inst is None:
            print("Let's create a DB instance.")
            admin_username = q.ask(
                "Enter an administrator user name for the database: ",
q.non_empty
            )
            admin_password = q.ask(
                "Enter a password for the administrator (at least 8 characters):
",
                q.non_empty,
            )
            engine_versions = self.instance_wrapper.get_engine_versions(
                db_engine, parameter_group["DBParameterGroupFamily"]
            )
            engine_choices = [ver["EngineVersion"] for ver in engine_versions]
            print("The available engines for your parameter group are:")
            engine_index = q.choose("Which engine do you want to use? ",
engine_choices)
            engine_selection = engine_versions[engine_index]
            print(
                "The available micro DB instance classes for your database engine
are:"
            )
            inst_opts = self.instance_wrapper.get_orderable_instances(
                engine_selection["Engine"], engine_selection["EngineVersion"]
```

```
            )
            inst_choices = list(
                {
                    opt["DBInstanceClass"]
                    for opt in inst_opts
                    if "micro" in opt["DBInstanceClass"]
                }
            )
            inst_index = q.choose(
                "Which micro DB instance class do you want to use? ",
 inst_choices
            )
            group_name = parameter_group["DBParameterGroupName"]
            storage_type = "standard"
            allocated_storage = 5
            print(
                f"Creating a DB instance named {instance_name} and database
 {db_name}.\n"
                f"The DB instance is configured to use your custom parameter
 group {group_name},\n"
                f"selected engine {engine_selection['EngineVersion']},\n"
                f"selected DB instance class {inst_choices[inst_index]},"
                f"and {allocated_storage} GiB of {storage_type} storage.\n"
                f"This typically takes several minutes."
            )
            db_inst = self.instance_wrapper.create_db_instance(
                db_name,
                instance_name,
                group_name,
                engine_selection["Engine"],
                engine_selection["EngineVersion"],
                inst_choices[inst_index],
                storage_type,
                allocated_storage,
                admin_username,
                admin_password,
            )
            while db_inst.get("DBInstanceStatus") != "available":
                wait(10)
                db_inst = self.instance_wrapper.get_db_instance(instance_name)
        print("Instance data:")
        pp(db_inst)
        print("-" * 88)
        return db_inst
```

```
    @staticmethod
    def display_connection(db_inst):
        """
        Displays connection information about a DB instance and tips on how to
        connect to it.

        :param db_inst: The DB instance to display.
        """
        print(
            "You can now connect to your database using your favorite MySql
 client.\n"
            "One way to connect is by using the 'mysql' shell on an Amazon EC2
 instance\n"
            "that is running in the same VPC as your DB instance. Pass the
 endpoint,\n"
            "port, and administrator user name to 'mysql' and enter your password
\n"
            "when prompted:\n"
        )
        print(
            f"\n\tmysql -h {db_inst['Endpoint']['Address']} -P
 {db_inst['Endpoint']['Port']} "
            f"-u {db_inst['MasterUsername']} -p\n"
        )
        print(
            "For more information, see the User Guide for Amazon RDS:\n"
            "\thttps://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/
CHAP_GettingStarted.CreatingConnecting.MySQL.html#CHAP_GettingStarted.Connecting.MySQL"
        )
        print("-" * 88)

    def create_snapshot(self, instance_name):
        """
        Shows how to create a DB instance snapshot and wait until it's available.

        :param instance_name: The name of a DB instance to snapshot.
        """
        if q.ask(
            "Do you want to create a snapshot of your DB instance (y/n)? ",
 q.is_yesno
        ):
            snapshot_id = f"{instance_name}-{uuid.uuid4()}"
            print(
```

```
                f"Creating a snapshot named {snapshot_id}. This typically takes a 
few minutes."
            )
            snapshot = self.instance_wrapper.create_snapshot(snapshot_id, 
instance_name)
            while snapshot.get("Status") != "available":
                wait(10)
                snapshot = self.instance_wrapper.get_snapshot(snapshot_id)
            pp(snapshot)
            print("-" * 88)

    def cleanup(self, db_inst, parameter_group_name):
        """
        Shows how to clean up a DB instance and parameter group.
        Before the parameter group can be deleted, all associated DB instances 
must first
        be deleted.

        :param db_inst: The DB instance to delete.
        :param parameter_group_name: The DB parameter group to delete.
        """
        if q.ask(
            "\nDo you want to delete the DB instance and parameter group (y/n)? 
",
            q.is_yesno,
        ):
            print(f"Deleting DB instance {db_inst['DBInstanceIdentifier']}.")
 
self.instance_wrapper.delete_db_instance(db_inst["DBInstanceIdentifier"])
            print(
                "Waiting for the DB instance to delete. This typically takes 
several minutes."
            )
            while db_inst is not None:
                wait(10)
                db_inst = self.instance_wrapper.get_db_instance(
                    db_inst["DBInstanceIdentifier"]
                )
            print(f"Deleting parameter group {parameter_group_name}.")
            self.instance_wrapper.delete_parameter_group(parameter_group_name)

    def run_scenario(self, db_engine, parameter_group_name, instance_name, 
db_name):
```

```
        logging.basicConfig(level=logging.INFO, format="%(levelname)s:
 %(message)s")

        print("-" * 88)
        print(
            "Welcome to the Amazon Relational Database Service (Amazon RDS)\n"
            "get started with DB instances demo."
        )
        print("-" * 88)

        parameter_group = self.create_parameter_group(parameter_group_name,
 db_engine)
        self.update_parameters(parameter_group_name)
        db_inst = self.create_instance(
            instance_name, db_name, db_engine, parameter_group
        )
        self.display_connection(db_inst)
        self.create_snapshot(instance_name)
        self.cleanup(db_inst, parameter_group_name)

        print("\nThanks for watching!")
        print("-" * 88)


if __name__ == "__main__":
    try:
        scenario = RdsInstanceScenario(InstanceWrapper.from_client())
        scenario.run_scenario(
            "mysql",
            "doc-example-parameter-group",
            "doc-example-instance",
            "docexampledb",
        )
    except Exception:
        logging.exception("Something went wrong with the demo.")
```

定义场景调用以管理 Amazon RDS 操作的函数。

```
class InstanceWrapper:
    """Encapsulates Amazon RDS DB instance actions."""

    def __init__(self, rds_client):
```

```
        """
        :param rds_client: A Boto3 Amazon RDS client.
        """
        self.rds_client = rds_client

    @classmethod
    def from_client(cls):
        """
        Instantiates this class from a Boto3 client.
        """
        rds_client = boto3.client("rds")
        return cls(rds_client)


    def get_parameter_group(self, parameter_group_name):
        """
        Gets a DB parameter group.

        :param parameter_group_name: The name of the parameter group to retrieve.
        :return: The parameter group.
        """
        try:
            response = self.rds_client.describe_db_parameter_groups(
                DBParameterGroupName=parameter_group_name
            )
            parameter_group = response["DBParameterGroups"][0]
        except ClientError as err:
            if err.response["Error"]["Code"] == "DBParameterGroupNotFound":
                logger.info("Parameter group %s does not exist.",
 parameter_group_name)
            else:
                logger.error(
                    "Couldn't get parameter group %s. Here's why: %s: %s",
                    parameter_group_name,
                    err.response["Error"]["Code"],
                    err.response["Error"]["Message"],
                )
                raise
        else:
            return parameter_group


    def create_parameter_group(
        self, parameter_group_name, parameter_group_family, description
```

```
    ):
        """
        Creates a DB parameter group that is based on the specified parameter
group
        family.

        :param parameter_group_name: The name of the newly created parameter
group.
        :param parameter_group_family: The family that is used as the basis of
the new
                                        parameter group.
        :param description: A description given to the parameter group.
        :return: Data about the newly created parameter group.
        """
        try:
            response = self.rds_client.create_db_parameter_group(
                DBParameterGroupName=parameter_group_name,
                DBParameterGroupFamily=parameter_group_family,
                Description=description,
            )
        except ClientError as err:
            logger.error(
                "Couldn't create parameter group %s. Here's why: %s: %s",
                parameter_group_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response


    def delete_parameter_group(self, parameter_group_name):
        """
        Deletes a DB parameter group.

        :param parameter_group_name: The name of the parameter group to delete.
        :return: Data about the parameter group.
        """
        try:
            self.rds_client.delete_db_parameter_group(
                DBParameterGroupName=parameter_group_name
            )
        except ClientError as err:
```

```
            logger.error(
                "Couldn't delete parameter group %s. Here's why: %s: %s",
                parameter_group_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def get_parameters(self, parameter_group_name, name_prefix="", source=None):
        """
        Gets the parameters that are contained in a DB parameter group.

        :param parameter_group_name: The name of the parameter group to query.
        :param name_prefix: When specified, the retrieved list of parameters is
 filtered
                            to contain only parameters that start with this
 prefix.
        :param source: When specified, only parameters from this source are
 retrieved.
                       For example, a source of 'user' retrieves only parameters
 that
                       were set by a user.
        :return: The list of requested parameters.
        """
        try:
            kwargs = {"DBParameterGroupName": parameter_group_name}
            if source is not None:
                kwargs["Source"] = source
            parameters = []
            paginator = self.rds_client.get_paginator("describe_db_parameters")
            for page in paginator.paginate(**kwargs):
                parameters += [
                    p
                    for p in page["Parameters"]
                    if p["ParameterName"].startswith(name_prefix)
                ]
        except ClientError as err:
            logger.error(
                "Couldn't get parameters for %s. Here's why: %s: %s",
                parameter_group_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
```

```
            raise
        else:
            return parameters


    def update_parameters(self, parameter_group_name, update_parameters):
        """
        Updates parameters in a custom DB parameter group.

        :param parameter_group_name: The name of the parameter group to update.
        :param update_parameters: The parameters to update in the group.
        :return: Data about the modified parameter group.
        """
        try:
            response = self.rds_client.modify_db_parameter_group(
                DBParameterGroupName=parameter_group_name,
Parameters=update_parameters
            )
        except ClientError as err:
            logger.error(
                "Couldn't update parameters in %s. Here's why: %s: %s",
                parameter_group_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response


    def create_snapshot(self, snapshot_id, instance_id):
        """
        Creates a snapshot of a DB instance.

        :param snapshot_id: The ID to give the created snapshot.
        :param instance_id: The ID of the DB instance to snapshot.
        :return: Data about the newly created snapshot.
        """
        try:
            response = self.rds_client.create_db_snapshot(
                DBSnapshotIdentifier=snapshot_id,
DBInstanceIdentifier=instance_id
            )
            snapshot = response["DBSnapshot"]
```

```
        except ClientError as err:
            logger.error(
                "Couldn't create snapshot of %s. Here's why: %s: %s",
                instance_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return snapshot


    def get_snapshot(self, snapshot_id):
        """
        Gets a DB instance snapshot.

        :param snapshot_id: The ID of the snapshot to retrieve.
        :return: The retrieved snapshot.
        """
        try:
            response = self.rds_client.describe_db_snapshots(
                DBSnapshotIdentifier=snapshot_id
            )
            snapshot = response["DBSnapshots"][0]
        except ClientError as err:
            logger.error(
                "Couldn't get snapshot %s. Here's why: %s: %s",
                snapshot_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return snapshot


    def get_engine_versions(self, engine, parameter_group_family=None):
        """
        Gets database engine versions that are available for the specified engine
        and parameter group family.

        :param engine: The database engine to look up.
        :param parameter_group_family: When specified, restricts the returned
 list of
```

```
                                              engine versions to those that are
compatible with
                                              this parameter group family.
        :return: The list of database engine versions.
        """
        try:
            kwargs = {"Engine": engine}
            if parameter_group_family is not None:
                kwargs["DBParameterGroupFamily"] = parameter_group_family
            response = self.rds_client.describe_db_engine_versions(**kwargs)
            versions = response["DBEngineVersions"]
        except ClientError as err:
            logger.error(
                "Couldn't get engine versions for %s. Here's why: %s: %s",
                engine,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return versions


    def get_orderable_instances(self, db_engine, db_engine_version):
        """
        Gets DB instance options that can be used to create DB instances that are
        compatible with a set of specifications.

        :param db_engine: The database engine that must be supported by the DB
instance.
        :param db_engine_version: The engine version that must be supported by
the DB instance.
        :return: The list of DB instance options that can be used to create a
compatible DB instance.
        """
        try:
            inst_opts = []
            paginator = self.rds_client.get_paginator(
                "describe_orderable_db_instance_options"
            )
            for page in paginator.paginate(
                Engine=db_engine, EngineVersion=db_engine_version
            ):
                inst_opts += page["OrderableDBInstanceOptions"]
```

```
        except ClientError as err:
            logger.error(
                "Couldn't get orderable DB instances. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return inst_opts


    def get_db_instance(self, instance_id):
        """
        Gets data about a DB instance.

        :param instance_id: The ID of the DB instance to retrieve.
        :return: The retrieved DB instance.
        """
        try:
            response = self.rds_client.describe_db_instances(
                DBInstanceIdentifier=instance_id
            )
            db_inst = response["DBInstances"][0]
        except ClientError as err:
            if err.response["Error"]["Code"] == "DBInstanceNotFound":
                logger.info("Instance %s does not exist.", instance_id)
            else:
                logger.error(
                    "Couldn't get DB instance %s. Here's why: %s: %s",
                    instance_id,
                    err.response["Error"]["Code"],
                    err.response["Error"]["Message"],
                )
                raise
        else:
            return db_inst


    def create_db_instance(
        self,
        db_name,
        instance_id,
        parameter_group_name,
        db_engine,
```

```
        db_engine_version,
        instance_class,
        storage_type,
        allocated_storage,
        admin_name,
        admin_password,
    ):
        """
        Creates a DB instance.

        :param db_name: The name of the database that is created in the DB
 instance.
        :param instance_id: The ID to give the newly created DB instance.
        :param parameter_group_name: A parameter group to associate with the DB
 instance.
        :param db_engine: The database engine of a database to create in the DB
 instance.
        :param db_engine_version: The engine version for the created database.
        :param instance_class: The DB instance class for the newly created DB
 instance.
        :param storage_type: The storage type of the DB instance.
        :param allocated_storage: The amount of storage allocated on the DB
 instance, in GiBs.
        :param admin_name: The name of the admin user for the created database.
        :param admin_password: The admin password for the created database.
        :return: Data about the newly created DB instance.
        """
        try:
            response = self.rds_client.create_db_instance(
                DBName=db_name,
                DBInstanceIdentifier=instance_id,
                DBParameterGroupName=parameter_group_name,
                Engine=db_engine,
                EngineVersion=db_engine_version,
                DBInstanceClass=instance_class,
                StorageType=storage_type,
                AllocatedStorage=allocated_storage,
                MasterUsername=admin_name,
                MasterUserPassword=admin_password,
            )
            db_inst = response["DBInstance"]
        except ClientError as err:
            logger.error(
                "Couldn't create DB instance %s. Here's why: %s: %s",
```

```
                instance_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return db_inst


    def delete_db_instance(self, instance_id):
        """
        Deletes a DB instance.

        :param instance_id: The ID of the DB instance to delete.
        :return: Data about the deleted DB instance.
        """
        try:
            response = self.rds_client.delete_db_instance(
                DBInstanceIdentifier=instance_id,
                SkipFinalSnapshot=True,
                DeleteAutomatedBackups=True,
            )
            db_inst = response["DBInstance"]
        except ClientError as err:
            logger.error(
                "Couldn't delete DB instance %s. Here's why: %s: %s",
                instance_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return db_inst
```

- 有关 API 详细信息，请参阅《Amazon SDK for Python (Boto3) API Reference》中的以下主题。
  - CreateDBInstance
  - CreateDBParameterGroup

- CreateDBSnapshot
- DeleteDBInstance
- DeleteDBParameterGroup
- DescribeDBEngineVersions
- DescribeDBInstances
- DescribeDBParameterGroups
- DescribeDBParameters
- DescribeDBSnapshots
- DescribeOrderableDBInstanceOptions
- ModifyDBParameterGroup

有关 Amazon SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 Amazon SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

适用于使用 Amazon SDK 的 Amazon RDS 的操作

以下代码示例演示了如何使用 Amazon SDK 来执行各个 Amazon RDS 操作。每个示例都包含一个指向 GitHub 的链接，您可以在其中找到有关设置和运行代码的说明。

这些代码节选调用了 Amazon RDS API，是必须在上下文中运行的较大型程序的代码节选。您可以在适用于使用 Amazon SDK 的 Amazon RDS 的场景 中结合上下文查看操作。

以下示例仅包括最常用的操作。有关完整列表，请参阅 Amazon Relational Database Service API Reference。

示例

- 将 CreateDBInstance 与 Amazon SDK 或 CLI 配合使用
- 将 CreateDBParameterGroup 与 Amazon SDK 或 CLI 配合使用
- 将 CreateDBSnapshot 与 Amazon SDK 或 CLI 配合使用
- 将 DeleteDBInstance 与 Amazon SDK 或 CLI 配合使用
- 将 DeleteDBParameterGroup 与 Amazon SDK 或 CLI 配合使用
- 将 DescribeAccountAttributes 与 Amazon SDK 或 CLI 配合使用
- 将 DescribeDBEngineVersions 与 Amazon SDK 或 CLI 配合使用
- 将 DescribeDBInstances 与 Amazon SDK 或 CLI 配合使用

- 将 DescribeDBParameterGroups 与 Amazon SDK 或 CLI 配合使用
- 将 DescribeDBParameters 与 Amazon SDK 或 CLI 配合使用
- 将 DescribeDBSnapshots 与 Amazon SDK 或 CLI 配合使用
- 将 DescribeOrderableDBInstanceOptions 与 Amazon SDK 或 CLI 配合使用
- 将 GenerateRDSAuthToken 和 Amazon SDK 搭配使用
- 将 ModifyDBInstance 与 Amazon SDK 或 CLI 配合使用
- 将 ModifyDBParameterGroup 与 Amazon SDK 或 CLI 配合使用
- 将 RebootDBInstance 与 Amazon SDK 或 CLI 配合使用

将 **CreateDBInstance** 与 Amazon SDK 或 CLI 配合使用

以下代码示例演示如何使用 `CreateDBInstance`。

操作示例是大型程序的代码摘录，必须在上下文中运行。在以下代码示例中，您可以查看此操作的上下文：

- 了解基础知识

.NET

适用于 .NET 的 Amazon SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
    /// <summary>
    /// Create an RDS DB instance with a particular set of properties. Use the
 action DescribeDBInstancesAsync
    /// to determine when the DB instance is ready to use.
    /// </summary>
    /// <param name="dbName">Name for the DB instance.</param>
    /// <param name="dbInstanceIdentifier">DB instance identifier.</param>
```

```
    /// <param name="parameterGroupName">DB parameter group to associate with the
 instance.</param>
    /// <param name="dbEngine">The engine for the DB instance.</param>
    /// <param name="dbEngineVersion">Version for the DB instance.</param>
    /// <param name="instanceClass">Class for the DB instance.</param>
    /// <param name="allocatedStorage">The amount of storage in gibibytes (GiB)
 to allocate to the DB instance.</param>
    /// <param name="adminName">Admin user name.</param>
    /// <param name="adminPassword">Admin user password.</param>
    /// <returns>DB instance object.</returns>
    public async Task<DBInstance> CreateDBInstance(string dbName, string
 dbInstanceIdentifier,
        string parameterGroupName, string dbEngine, string dbEngineVersion,
        string instanceClass, int allocatedStorage, string adminName, string
 adminPassword)
    {
        var response = await _amazonRDS.CreateDBInstanceAsync(
            new CreateDBInstanceRequest()
            {
                DBName = dbName,
                DBInstanceIdentifier = dbInstanceIdentifier,
                DBParameterGroupName = parameterGroupName,
                Engine = dbEngine,
                EngineVersion = dbEngineVersion,
                DBInstanceClass = instanceClass,
                AllocatedStorage = allocatedStorage,
                MasterUsername = adminName,
                MasterUserPassword = adminPassword
            });

        return response.DBInstance;
    }
```

- 有关 API 详细信息，请参阅《适用于 .NET 的 Amazon SDK API 参考》中的 CreateDBInstance。

C++

SDK for C++

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::RDS::RDSClient client(clientConfig);

        Aws::RDS::Model::CreateDBInstanceRequest request;
        request.SetDBName(DB_NAME);
        request.SetDBInstanceIdentifier(DB_INSTANCE_IDENTIFIER);
        request.SetDBParameterGroupName(PARAMETER_GROUP_NAME);
        request.SetEngine(engineVersion.GetEngine());
        request.SetEngineVersion(engineVersion.GetEngineVersion());
        request.SetDBInstanceClass(dbInstanceClass);
        request.SetStorageType(DB_STORAGE_TYPE);
        request.SetAllocatedStorage(DB_ALLOCATED_STORAGE);
        request.SetMasterUsername(administratorName);
        request.SetMasterUserPassword(administratorPassword);

        Aws::RDS::Model::CreateDBInstanceOutcome outcome =
                client.CreateDBInstance(request);

        if (outcome.IsSuccess()) {
            std::cout << "The DB instance creation has started."
                      << std::endl;
        }
        else {
            std::cerr << "Error with RDS::CreateDBInstance. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            cleanUpResources(PARAMETER_GROUP_NAME, "", client);
            return false;
        }
```

- 有关 API 详细信息，请参阅《适用于 C++ 的 Amazon SDK API 参考》中的 CreateDBInstance。

CLI

Amazon CLI

创建数据库实例

以下 `create-db-instance` 示例使用所需的选项启动新的数据库实例。

```
aws rds create-db-instance \
    --db-instance-identifier test-mysql-instance \
    --db-instance-class db.t3.micro \
    --engine mysql \
    --master-username admin \
    --master-user-password secret99 \
    --allocated-storage 20
```

输出：

```
{
    "DBInstance": {
        "DBInstanceIdentifier": "test-mysql-instance",
        "DBInstanceClass": "db.t3.micro",
        "Engine": "mysql",
        "DBInstanceStatus": "creating",
        "MasterUsername": "admin",
        "AllocatedStorage": 20,
        "PreferredBackupWindow": "12:55-13:25",
        "BackupRetentionPeriod": 1,
        "DBSecurityGroups": [],
        "VpcSecurityGroups": [
            {
                "VpcSecurityGroupId": "sg-12345abc",
                "Status": "active"
            }
        ],
        "DBParameterGroups": [
            {
```

```
                "DBParameterGroupName": "default.mysql5.7",
                "ParameterApplyStatus": "in-sync"
            }
        ],
        "DBSubnetGroup": {
            "DBSubnetGroupName": "default",
            "DBSubnetGroupDescription": "default",
            "VpcId": "vpc-2ff2ff2f",
            "SubnetGroupStatus": "Complete",
            "Subnets": [
                {
                    "SubnetIdentifier": "subnet-########",
                    "SubnetAvailabilityZone": {
                        "Name": "us-west-2c"
                    },
                    "SubnetStatus": "Active"
                },
                {
                    "SubnetIdentifier": "subnet-########",
                    "SubnetAvailabilityZone": {
                        "Name": "us-west-2d"
                    },
                    "SubnetStatus": "Active"
                },
                {
                    "SubnetIdentifier": "subnet-########",
                    "SubnetAvailabilityZone": {
                        "Name": "us-west-2a"
                    },
                    "SubnetStatus": "Active"
                },
                {
                    "SubnetIdentifier": "subnet-########",
                    "SubnetAvailabilityZone": {
                        "Name": "us-west-2b"
                    },
                    "SubnetStatus": "Active"
                }
            ]
        },
        "PreferredMaintenanceWindow": "sun:08:07-sun:08:37",
        "PendingModifiedValues": {
            "MasterUserPassword": "****"
        },
```

```
        "MultiAZ": false,
        "EngineVersion": "5.7.22",
        "AutoMinorVersionUpgrade": true,
        "ReadReplicaDBInstanceIdentifiers": [],
        "LicenseModel": "general-public-license",
        "OptionGroupMemberships": [
            {
                "OptionGroupName": "default:mysql-5-7",
                "Status": "in-sync"
            }
        ],
        "PubliclyAccessible": true,
        "StorageType": "gp2",
        "DbInstancePort": 0,
        "StorageEncrypted": false,
        "DbiResourceId": "db-5555EXAMPLE44444444EXAMPLE",
        "CACertificateIdentifier": "rds-ca-2019",
        "DomainMemberships": [],
        "CopyTagsToSnapshot": false,
        "MonitoringInterval": 0,
        "DBInstanceArn": "arn:aws:rds:us-west-2:123456789012:db:test-mysql-
instance",
        "IAMDatabaseAuthenticationEnabled": false,
        "PerformanceInsightsEnabled": false,
        "DeletionProtection": false,
        "AssociatedRoles": []
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的创建 Amazon RDS 数据库实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDBInstance。

Go

适用于 Go V2 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import (
 "context"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/rds"
 "github.com/aws/aws-sdk-go-v2/service/rds/types"
)

type DbInstances struct {
 RdsClient *rds.Client
}



// CreateInstance creates a DB instance.
func (instances *DbInstances) CreateInstance(ctx context.Context, instanceName
 string, dbName string,
 dbEngine string, dbEngineVersion string, parameterGroupName string,
 dbInstanceClass string,
 storageType string, allocatedStorage int32, adminName string, adminPassword
 string) (
 *types.DBInstance, error) {
 output, err := instances.RdsClient.CreateDBInstance(ctx,
 &rds.CreateDBInstanceInput{
  DBInstanceIdentifier: aws.String(instanceName),
  DBName:               aws.String(dbName),
  DBParameterGroupName: aws.String(parameterGroupName),
  Engine:               aws.String(dbEngine),
  EngineVersion:        aws.String(dbEngineVersion),
  DBInstanceClass:      aws.String(dbInstanceClass),
  StorageType:          aws.String(storageType),
  AllocatedStorage:     aws.Int32(allocatedStorage),
  MasterUsername:       aws.String(adminName),
  MasterUserPassword:   aws.String(adminPassword),
 })
 if err != nil {
  log.Printf("Couldn't create instance %v: %v\n", instanceName, err)
  return nil, err
 } else {
  return output.DBInstance, nil
```

```
  }
}
```

- 有关 API 详细信息，请参阅《适用于 Go 的 Amazon SDK API 参考》中的 CreateDBInstance。

Java

适用于 Java 的 SDK 2.x

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import com.google.gson.Gson;
import
 software.amazon.awssdk.auth.credentials.EnvironmentVariableCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.rds.RdsClient;
import software.amazon.awssdk.services.rds.model.DescribeDbInstancesRequest;
import software.amazon.awssdk.services.rds.model.CreateDbInstanceRequest;
import software.amazon.awssdk.services.rds.model.CreateDbInstanceResponse;
import software.amazon.awssdk.services.rds.model.RdsException;
import software.amazon.awssdk.services.rds.model.DescribeDbInstancesResponse;
import software.amazon.awssdk.services.rds.model.DBInstance;
import software.amazon.awssdk.services.secretsmanager.SecretsManagerClient;
import
 software.amazon.awssdk.services.secretsmanager.model.GetSecretValueRequest;
import
 software.amazon.awssdk.services.secretsmanager.model.GetSecretValueResponse;

import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
```

```
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 *
 * This example requires an AWS Secrets Manager secret that contains the
 * database credentials. If you do not create a
 * secret, this example will not work. For more details, see:
 *
 * https://docs.aws.amazon.com/secretsmanager/latest/userguide/integrating_how-
services-use-secrets_RS.html
 *
 *
 */

public class CreateDBInstance {
    public static long sleepTime = 20;

    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <dbInstanceIdentifier> <dbName> <secretName>

                Where:
                    dbInstanceIdentifier - The database instance identifier.\s
                    dbName - The database name.\s
                    secretName - The name of the AWS Secrets Manager secret that
 contains the database credentials."
                """;

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String dbInstanceIdentifier = args[0];
        String dbName = args[1];
        String secretName = args[2];
        Gson gson = new Gson();
        User user = gson.fromJson(String.valueOf(getSecretValues(secretName)),
 User.class);
        Region region = Region.US_WEST_2;
        RdsClient rdsClient = RdsClient.builder()
```

```
                .region(region)
                .build();

        createDatabaseInstance(rdsClient, dbInstanceIdentifier, dbName,
user.getUsername(), user.getPassword());
        waitForInstanceReady(rdsClient, dbInstanceIdentifier);
        rdsClient.close();
    }

    private static SecretsManagerClient getSecretClient() {
        Region region = Region.US_WEST_2;
        return SecretsManagerClient.builder()
                .region(region)

.credentialsProvider(EnvironmentVariableCredentialsProvider.create())
                .build();
    }

    private static String getSecretValues(String secretName) {
        SecretsManagerClient secretClient = getSecretClient();
        GetSecretValueRequest valueRequest = GetSecretValueRequest.builder()
                .secretId(secretName)
                .build();

        GetSecretValueResponse valueResponse =
secretClient.getSecretValue(valueRequest);
        return valueResponse.secretString();
    }

    public static void createDatabaseInstance(RdsClient rdsClient,
            String dbInstanceIdentifier,
            String dbName,
            String userName,
            String userPassword) {

        try {
            CreateDbInstanceRequest instanceRequest =
CreateDbInstanceRequest.builder()
                .dbInstanceIdentifier(dbInstanceIdentifier)
                .allocatedStorage(100)
                .dbName(dbName)
                .engine("mysql")
                .dbInstanceClass("db.t3.medium") // Updated to a supported class
```

```
                .engineVersion("8.0.32")         // Updated to a supported
version
                .storageType("gp2")             // Changed to General Purpose SSD
(gp2)
                .masterUsername(userName)
                .masterUserPassword(userPassword)
                .build();

            CreateDbInstanceResponse response =
rdsClient.createDBInstance(instanceRequest);
            System.out.print("The status is " +
response.dbInstance().dbInstanceStatus());

        } catch (RdsException e) {
            System.out.println(e.getLocalizedMessage());
            System.exit(1);
        }
    }

    // Waits until the database instance is available.
    public static void waitForInstanceReady(RdsClient rdsClient, String
dbInstanceIdentifier) {
        boolean instanceReady = false;
        String instanceReadyStr;
        System.out.println("Waiting for instance to become available.");
        try {
            DescribeDbInstancesRequest instanceRequest =
DescribeDbInstancesRequest.builder()
                    .dbInstanceIdentifier(dbInstanceIdentifier)
                    .build();

            // Loop until the cluster is ready.
            while (!instanceReady) {
                DescribeDbInstancesResponse response =
rdsClient.describeDBInstances(instanceRequest);
                List<DBInstance> instanceList = response.dbInstances();
                for (DBInstance instance : instanceList) {
                    instanceReadyStr = instance.dbInstanceStatus();
                    if (instanceReadyStr.contains("available"))
                        instanceReady = true;
                    else {
                        System.out.print(".");
                        Thread.sleep(sleepTime * 1000);
                    }
```

```
                }
            }
            System.out.println("Database instance is available!");

        } catch (RdsException | InterruptedException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
```

- 有关 API 详细信息，请参阅《Amazon SDK for Java 2.x API 参考》中的 CreateDBInstance。

Kotlin

适用于 Kotlin 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
suspend fun createDatabaseInstance(
    dbInstanceIdentifierVal: String?,
    dbNamedbVal: String?,
    masterUsernameVal: String?,
    masterUserPasswordVal: String?,
) {
    val instanceRequest =
        CreateDbInstanceRequest {
            dbInstanceIdentifier = dbInstanceIdentifierVal
            allocatedStorage = 100
            dbName = dbNamedbVal
            engine = "mysql"
            dbInstanceClass = "db.t3.micro" // Use a supported instance class
            engineVersion = "8.0.39" // Use a supported engine version
            storageType = "gp2"
            masterUsername = masterUsernameVal
            masterUserPassword = masterUserPasswordVal
```

```
        }

    RdsClient { region = "us-west-2" }.use { rdsClient ->
        val response = rdsClient.createDbInstance(instanceRequest)
        print("The status is ${response.dbInstance?.dbInstanceStatus}")
    }
}

// Waits until the database instance is available.
suspend fun waitForInstanceReady(dbInstanceIdentifierVal: String?) {
    val sleepTime: Long = 20
    var instanceReady = false
    var instanceReadyStr: String
    println("Waiting for instance to become available.")

    val instanceRequest =
        DescribeDbInstancesRequest {
            dbInstanceIdentifier = dbInstanceIdentifierVal
        }

    RdsClient { region = "us-west-2" }.use { rdsClient ->
        while (!instanceReady) {
            val response = rdsClient.describeDbInstances(instanceRequest)
            val instanceList = response.dbInstances
            if (instanceList != null) {
                for (instance in instanceList) {
                    instanceReadyStr = instance.dbInstanceStatus.toString()
                    if (instanceReadyStr.contains("available")) {
                        instanceReady = true
                    } else {
                        println("...$instanceReadyStr")
                        delay(sleepTime * 1000)
                    }
                }
            }
        }
        println("Database instance is available!")
    }
}
```

- 有关 API 详细信息，请参阅《Amazon SDK for Kotlin API 参考》中的 CreateDBInstance。

PHP

适用于 PHP 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
require __DIR__ . '/vendor/autoload.php';

use Aws\Exception\AwsException;



$rdsClient = new Aws\Rds\RdsClient([
    'region' => 'us-east-2'
]);

$dbIdentifier = '<<{{db-identifier}}>>';
$dbClass = 'db.t2.micro';
$storage = 5;
$engine = 'MySQL';
$username = 'MyUser';
$password = 'MyPassword';

try {
    $result = $rdsClient->createDBInstance([
        'DBInstanceIdentifier' => $dbIdentifier,
        'DBInstanceClass' => $dbClass,
        'AllocatedStorage' => $storage,
        'Engine' => $engine,
        'MasterUsername' => $username,
        'MasterUserPassword' => $password,
    ]);
    var_dump($result);
} catch (AwsException $e) {
    echo $e->getMessage();
    echo "\n";
}
```

- 有关 API 详细信息，请参阅《适用于 PHP 的 Amazon SDK API 参考》中的 CreateDBInstance。

Python

适用于 Python 的 SDK（Boto3）

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
class InstanceWrapper:
    """Encapsulates Amazon RDS DB instance actions."""

    def __init__(self, rds_client):
        """
        :param rds_client: A Boto3 Amazon RDS client.
        """
        self.rds_client = rds_client

    @classmethod
    def from_client(cls):
        """
        Instantiates this class from a Boto3 client.
        """
        rds_client = boto3.client("rds")
        return cls(rds_client)


    def create_db_instance(
        self,
        db_name,
        instance_id,
        parameter_group_name,
        db_engine,
        db_engine_version,
```

```
        instance_class,
        storage_type,
        allocated_storage,
        admin_name,
        admin_password,
    ):
        """
        Creates a DB instance.

        :param db_name: The name of the database that is created in the DB
 instance.
        :param instance_id: The ID to give the newly created DB instance.
        :param parameter_group_name: A parameter group to associate with the DB
 instance.
        :param db_engine: The database engine of a database to create in the DB
 instance.
        :param db_engine_version: The engine version for the created database.
        :param instance_class: The DB instance class for the newly created DB
 instance.
        :param storage_type: The storage type of the DB instance.
        :param allocated_storage: The amount of storage allocated on the DB
 instance, in GiBs.
        :param admin_name: The name of the admin user for the created database.
        :param admin_password: The admin password for the created database.
        :return: Data about the newly created DB instance.
        """
        try:
            response = self.rds_client.create_db_instance(
                DBName=db_name,
                DBInstanceIdentifier=instance_id,
                DBParameterGroupName=parameter_group_name,
                Engine=db_engine,
                EngineVersion=db_engine_version,
                DBInstanceClass=instance_class,
                StorageType=storage_type,
                AllocatedStorage=allocated_storage,
                MasterUsername=admin_name,
                MasterUserPassword=admin_password,
            )
            db_inst = response["DBInstance"]
        except ClientError as err:
            logger.error(
                "Couldn't create DB instance %s. Here's why: %s: %s",
                instance_id,
```

```
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return db_inst
```

- 有关 API 详细信息，请参阅《适用于 Python 的 Amazon SDK（Boto3）API 参考》中的 CreateDBInstance。

有关 Amazon SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 Amazon SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **CreateDBParameterGroup** 与 Amazon SDK 或 CLI 配合使用

以下代码示例演示如何使用 `CreateDBParameterGroup`。

操作示例是大型程序的代码摘录，必须在上下文中运行。在以下代码示例中，您可以查看此操作的上下文：

- 了解基础知识

.NET

适用于 .NET 的 Amazon SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
    /// <summary>
    /// Create a new DB parameter group. Use the action
DescribeDBParameterGroupsAsync
    /// to determine when the DB parameter group is ready to use.
    /// </summary>
```

```
    /// <param name="name">Name of the DB parameter group.</param>
    /// <param name="family">Family of the DB parameter group.</param>
    /// <param name="description">Description of the DB parameter group.</param>
    /// <returns>The new DB parameter group.</returns>
    public async Task<DBParameterGroup> CreateDBParameterGroup(
        string name, string family, string description)
    {
        var response = await _amazonRDS.CreateDBParameterGroupAsync(
            new CreateDBParameterGroupRequest()
            {
                DBParameterGroupName = name,
                DBParameterGroupFamily = family,
                Description = description
            });
        return response.DBParameterGroup;
    }
```

- 有关 API 详细信息，请参阅《适用于 .NET 的 Amazon SDK API 参考》中的 CreateDBParameterGroup。

C++

SDK for C++

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::RDS::RDSClient client(clientConfig);

        Aws::RDS::Model::CreateDBParameterGroupRequest request;
        request.SetDBParameterGroupName(PARAMETER_GROUP_NAME);
        request.SetDBParameterGroupFamily(dbParameterGroupFamily);
```

```
        request.SetDescription("Example parameter group.");

        Aws::RDS::Model::CreateDBParameterGroupOutcome outcome =
                client.CreateDBParameterGroup(request);

        if (outcome.IsSuccess()) {
            std::cout << "The DB parameter group was successfully created."
                      << std::endl;
        }
        else {
            std::cerr << "Error with RDS::CreateDBParameterGroup. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            return false;
        }
```

- 有关 API 详细信息，请参阅《适用于 C++ 的 Amazon SDK API 参考》中的 CreateDBParameterGroup。

CLI

Amazon CLI

创建数据库参数组

以下 `create-db-parameter-group` 示例创建一个数据库参数组。

```
aws rds create-db-parameter-group \
    --db-parameter-group-name mydbparametergroup \
    --db-parameter-group-family MySQL5.6 \
    --description "My new parameter group"
```

输出：

```
{
    "DBParameterGroup": {
        "DBParameterGroupName": "mydbparametergroup",
        "DBParameterGroupFamily": "mysql5.6",
        "Description": "My new parameter group",
        "DBParameterGroupArn": "arn:aws:rds:us-
east-1:123456789012:pg:mydbparametergroup"
```

```
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的创建数据库参数组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDBParameterGroup。

Go

适用于 Go V2 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import (
 "context"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/rds"
 "github.com/aws/aws-sdk-go-v2/service/rds/types"
)

type DbInstances struct {
 RdsClient *rds.Client
}



// CreateParameterGroup creates a DB parameter group that is based on the
 specified
// parameter group family.
func (instances *DbInstances) CreateParameterGroup(
 ctx context.Context, parameterGroupName string, parameterGroupFamily string,
 description string) (
 *types.DBParameterGroup, error) {
```

```
	output, err := instances.RdsClient.CreateDBParameterGroup(ctx,
	 &rds.CreateDBParameterGroupInput{
	  DBParameterGroupName:   aws.String(parameterGroupName),
	  DBParameterGroupFamily: aws.String(parameterGroupFamily),
	  Description:            aws.String(description),
	 })
	if err != nil {
	 log.Printf("Couldn't create parameter group %v: %v\n", parameterGroupName, err)
	 return nil, err
	} else {
	 return output.DBParameterGroup, err
	}
}
```

- 有关 API 详细信息，请参阅《适用于 Go 的 Amazon SDK API 参考》中的 CreateDBParameterGroup。

Java

适用于 Java 的 SDK 2.x

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
    public static void createDBParameterGroup(RdsClient rdsClient, String
 dbGroupName, String dbParameterGroupFamily) {
        try {
            CreateDbParameterGroupRequest groupRequest =
 CreateDbParameterGroupRequest.builder()
                    .dbParameterGroupName(dbGroupName)
                    .dbParameterGroupFamily(dbParameterGroupFamily)
                    .description("Created by using the AWS SDK for Java")
                    .build();

            CreateDbParameterGroupResponse response =
 rdsClient.createDBParameterGroup(groupRequest);
```

```
            System.out.println("The group name is " +
 response.dbParameterGroup().dbParameterGroupName());

        } catch (RdsException e) {
            System.out.println(e.getLocalizedMessage());
            System.exit(1);
        }
    }
```

- 有关 API 详细信息，请参阅《Amazon SDK for Java 2.x API 参考》中的 CreateDBParameterGroup。

Python

适用于 Python 的 SDK（Boto3）

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
class InstanceWrapper:
    """Encapsulates Amazon RDS DB instance actions."""

    def __init__(self, rds_client):
        """
        :param rds_client: A Boto3 Amazon RDS client.
        """
        self.rds_client = rds_client

    @classmethod
    def from_client(cls):
        """
        Instantiates this class from a Boto3 client.
        """
        rds_client = boto3.client("rds")
        return cls(rds_client)
```

```
    def create_parameter_group(
        self, parameter_group_name, parameter_group_family, description
    ):
        """
        Creates a DB parameter group that is based on the specified parameter
 group
        family.

        :param parameter_group_name: The name of the newly created parameter
 group.
        :param parameter_group_family: The family that is used as the basis of
 the new
                                       parameter group.
        :param description: A description given to the parameter group.
        :return: Data about the newly created parameter group.
        """
        try:
            response = self.rds_client.create_db_parameter_group(
                DBParameterGroupName=parameter_group_name,
                DBParameterGroupFamily=parameter_group_family,
                Description=description,
            )
        except ClientError as err:
            logger.error(
                "Couldn't create parameter group %s. Here's why: %s: %s",
                parameter_group_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response
```

- 有关 API 详细信息，请参阅《适用于 Python 的 Amazon SDK（Boto3）API 参考》中的 CreateDBParameterGroup。

有关 Amazon SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 Amazon SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 CreateDBSnapshot 与 Amazon SDK 或 CLI 配合使用

以下代码示例演示如何使用 CreateDBSnapshot。

操作示例是大型程序的代码摘录，必须在上下文中运行。在以下代码示例中，您可以查看此操作的上下文：

- 了解基础知识

.NET

适用于 .NET 的 Amazon SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
    /// <summary>
    /// Create a snapshot of a DB instance.
    /// </summary>
    /// <param name="dbInstanceIdentifier">DB instance identifier.</param>
    /// <param name="snapshotIdentifier">Identifier for the snapshot.</param>
    /// <returns>DB snapshot object.</returns>
    public async Task<DBSnapshot> CreateDBSnapshot(string dbInstanceIdentifier,
 string snapshotIdentifier)
    {
        var response = await _amazonRDS.CreateDBSnapshotAsync(
            new CreateDBSnapshotRequest()
            {
                DBSnapshotIdentifier = snapshotIdentifier,
                DBInstanceIdentifier = dbInstanceIdentifier
            });

        return response.DBSnapshot;
    }
```

- 有关 API 详细信息，请参阅《适用于 .NET 的 Amazon SDK API 参考》中的 CreateDBSnapshot。

C++

SDK for C++

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::RDS::RDSClient client(clientConfig);

            Aws::RDS::Model::CreateDBSnapshotRequest request;
            request.SetDBInstanceIdentifier(DB_INSTANCE_IDENTIFIER);
            request.SetDBSnapshotIdentifier(snapshotID);

            Aws::RDS::Model::CreateDBSnapshotOutcome outcome =
                    client.CreateDBSnapshot(request);

            if (outcome.IsSuccess()) {
                std::cout << "Snapshot creation has started."
                          << std::endl;
            }
            else {
                std::cerr << "Error with RDS::CreateDBSnapshot. "
                          << outcome.GetError().GetMessage()
                          << std::endl;
                cleanUpResources(PARAMETER_GROUP_NAME, DB_INSTANCE_IDENTIFIER,
 client);
                return false;
            }
```

- 有关 API 详细信息，请参阅《适用于 C++ 的 Amazon SDK API 参考》中的 CreateDBSnapshot。

CLI

Amazon CLI

创建数据库快照

以下 `create-db-snapshot` 示例创建数据库快照。

```
aws rds create-db-snapshot \
    --db-instance-identifier database-mysql \
    --db-snapshot-identifier mydbsnapshot
```

输出：

```
{
    "DBSnapshot": {
        "DBSnapshotIdentifier": "mydbsnapshot",
        "DBInstanceIdentifier": "database-mysql",
        "Engine": "mysql",
        "AllocatedStorage": 100,
        "Status": "creating",
        "Port": 3306,
        "AvailabilityZone": "us-east-1b",
        "VpcId": "vpc-6594f31c",
        "InstanceCreateTime": "2019-04-30T15:45:53.663Z",
        "MasterUsername": "admin",
        "EngineVersion": "5.6.40",
        "LicenseModel": "general-public-license",
        "SnapshotType": "manual",
        "Iops": 1000,
        "OptionGroupName": "default:mysql-5-6",
        "PercentProgress": 0,
        "StorageType": "io1",
        "Encrypted": true,
        "KmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/
AKIAIOSFODNN7EXAMPLE",
        "DBSnapshotArn": "arn:aws:rds:us-
east-1:123456789012:snapshot:mydbsnapshot",
        "IAMDatabaseAuthenticationEnabled": false,
```

```
        "ProcessorFeatures": [],
        "DbiResourceId": "db-AKIAIOSFODNN7EXAMPLE"
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的创建数据库快照。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 CreateDBSnapshot。

Go

适用于 Go V2 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import (
 "context"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/rds"
 "github.com/aws/aws-sdk-go-v2/service/rds/types"
)

type DbInstances struct {
 RdsClient *rds.Client
}



// CreateSnapshot creates a snapshot of a DB instance.
func (instances *DbInstances) CreateSnapshot(ctx context.Context, instanceName
 string, snapshotName string) (
 *types.DBSnapshot, error) {
 output, err := instances.RdsClient.CreateDBSnapshot(ctx,
 &rds.CreateDBSnapshotInput{
```

```
        DBInstanceIdentifier: aws.String(instanceName),
        DBSnapshotIdentifier: aws.String(snapshotName),
    })
    if err != nil {
        log.Printf("Couldn't create snapshot %v: %v\n", snapshotName, err)
        return nil, err
    } else {
        return output.DBSnapshot, nil
    }
}
```

- 有关 API 详细信息，请参阅《适用于 Go 的 Amazon SDK API 参考》中的 CreateDBSnapshot。

Java

适用于 Java 的 SDK 2.x

> **Note**
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
    // Create an Amazon RDS snapshot.
    public static void createSnapshot(RdsClient rdsClient, String
dbInstanceIdentifier, String dbSnapshotIdentifier) {
        try {
            CreateDbSnapshotRequest snapshotRequest =
CreateDbSnapshotRequest.builder()
                    .dbInstanceIdentifier(dbInstanceIdentifier)
                    .dbSnapshotIdentifier(dbSnapshotIdentifier)
                    .build();

            CreateDbSnapshotResponse response =
rdsClient.createDBSnapshot(snapshotRequest);
            System.out.println("The Snapshot id is " +
response.dbSnapshot().dbiResourceId());
```

```
        } catch (RdsException e) {
            System.out.println(e.getLocalizedMessage());
            System.exit(1);
        }
    }
```

- 有关 API 详细信息，请参阅《Amazon SDK for Java 2.x API 参考》中的 CreateDBSnapshot。

PHP

适用于 PHP 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
require __DIR__ . '/vendor/autoload.php';

use Aws\Exception\AwsException;



$rdsClient = new Aws\Rds\RdsClient([
    'region' => 'us-east-2'
]);

$dbIdentifier = '<<{{db-identifier}}>>';
$snapshotName = '<<{{backup_2018_12_25}}>>';

try {
    $result = $rdsClient->createDBSnapshot([
        'DBInstanceIdentifier' => $dbIdentifier,
        'DBSnapshotIdentifier' => $snapshotName,
    ]);
    var_dump($result);
} catch (AwsException $e) {
```

```
    echo $e->getMessage();
    echo "\n";
}
```

- 有关 API 详细信息，请参阅《适用于 PHP 的 Amazon SDK API 参考》中的 CreateDBSnapshot。

Python

适用于 Python 的 SDK（Boto3）

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
class InstanceWrapper:
    """Encapsulates Amazon RDS DB instance actions."""

    def __init__(self, rds_client):
        """
        :param rds_client: A Boto3 Amazon RDS client.
        """
        self.rds_client = rds_client

    @classmethod
    def from_client(cls):
        """
        Instantiates this class from a Boto3 client.
        """
        rds_client = boto3.client("rds")
        return cls(rds_client)


    def create_snapshot(self, snapshot_id, instance_id):
        """
        Creates a snapshot of a DB instance.
```

```
        :param snapshot_id: The ID to give the created snapshot.
        :param instance_id: The ID of the DB instance to snapshot.
        :return: Data about the newly created snapshot.
        """
        try:
            response = self.rds_client.create_db_snapshot(
                DBSnapshotIdentifier=snapshot_id,
 DBInstanceIdentifier=instance_id
            )
            snapshot = response["DBSnapshot"]
        except ClientError as err:
            logger.error(
                "Couldn't create snapshot of %s. Here's why: %s: %s",
                instance_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return snapshot
```

- 有关 API 详细信息，请参阅《适用于 Python 的 Amazon SDK（Boto3）API 参考》中的 CreateDBSnapshot。

Ruby

适用于 Ruby 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
require 'aws-sdk-rds' # v2: require 'aws-sdk'

# Create a snapshot for an Amazon Relational Database Service (Amazon RDS)
# DB instance.
#
```

```
# @param rds_resource [Aws::RDS::Resource] The resource containing SDK logic.
# @param db_instance_name [String] The name of the Amazon RDS DB instance.
# @return [Aws::RDS::DBSnapshot, nil] The snapshot created, or nil if error.
def create_snapshot(rds_resource, db_instance_name)
  id = "snapshot-#{rand(10**6)}"
  db_instance = rds_resource.db_instance(db_instance_name)
  db_instance.create_snapshot({
                                db_snapshot_identifier: id
                              })
rescue Aws::Errors::ServiceError => e
  puts "Couldn't create DB instance snapshot #{id}:\n #{e.message}"
end
```

- 有关 API 详细信息，请参阅《适用于 Ruby 的 Amazon SDK API 参考》中的 CreateDBSnapshot。

有关 Amazon SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 Amazon SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 `DeleteDBInstance` 与 Amazon SDK 或 CLI 配合使用

以下代码示例演示如何使用 `DeleteDBInstance`。

操作示例是大型程序的代码摘录，必须在上下文中运行。在以下代码示例中，您可以查看此操作的上下文：

- 了解基础知识

.NET

适用于 .NET 的 Amazon SDK

> Note
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
    /// <summary>
    /// Delete a particular DB instance.
    /// </summary>
    /// <param name="dbInstanceIdentifier">DB instance identifier.</param>
    /// <returns>DB instance object.</returns>
    public async Task<DBInstance> DeleteDBInstance(string dbInstanceIdentifier)
    {
        var response = await _amazonRDS.DeleteDBInstanceAsync(
            new DeleteDBInstanceRequest()
            {
                DBInstanceIdentifier = dbInstanceIdentifier,
                SkipFinalSnapshot = true,
                DeleteAutomatedBackups = true
            });

        return response.DBInstance;
    }
```

- 有关 API 详细信息，请参阅《适用于 .NET 的 Amazon SDK API 参考》中的 DeleteDBInstance。

C++

SDK for C++

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::RDS::RDSClient client(clientConfig);

            Aws::RDS::Model::DeleteDBInstanceRequest request;
            request.SetDBInstanceIdentifier(dbInstanceIdentifier);
```

```
            request.SetSkipFinalSnapshot(true);
            request.SetDeleteAutomatedBackups(true);

            Aws::RDS::Model::DeleteDBInstanceOutcome outcome =
                    client.DeleteDBInstance(request);

            if (outcome.IsSuccess()) {
                std::cout << "DB instance deletion has started."
                          << std::endl;
            }
            else {
                std::cerr << "Error with RDS::DeleteDBInstance. "
                          << outcome.GetError().GetMessage()
                          << std::endl;
                result = false;
            }
```

- 有关 API 详细信息，请参阅《适用于 C++ 的 Amazon SDK API 参考》中的 DeleteDBInstance。

CLI

Amazon CLI

删除数据库实例

以下 `delete-db-instance` 示例在创建名为 `test-instance-final-snap` 的最终数据库快照后删除指定的数据库实例。

```
aws rds delete-db-instance \
    --db-instance-identifier test-instance \
    --final-db-snapshot-identifier test-instance-final-snap
```

输出：

```
{
    "DBInstance": {
        "DBInstanceIdentifier": "test-instance",
        "DBInstanceStatus": "deleting",
        ...some output truncated...
    }
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDBInstance。

Go

适用于 Go V2 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import (
 "context"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/rds"
 "github.com/aws/aws-sdk-go-v2/service/rds/types"
)

type DbInstances struct {
 RdsClient *rds.Client
}



// DeleteInstance deletes a DB instance.
func (instances *DbInstances) DeleteInstance(ctx context.Context, instanceName
 string) error {
 _, err := instances.RdsClient.DeleteDBInstance(ctx, &rds.DeleteDBInstanceInput{
  DBInstanceIdentifier:   aws.String(instanceName),
  SkipFinalSnapshot:      aws.Bool(true),
  DeleteAutomatedBackups: aws.Bool(true),
 })
 if err != nil {
  log.Printf("Couldn't delete instance %v: %v\n", instanceName, err)
  return err
```

```
 } else {
  return nil
 }
}
```

- 有关 API 详细信息，请参阅《适用于 Go 的 Amazon SDK API 参考》中的 DeleteDBInstance。

Java

适用于 Java 的 SDK 2.x

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.rds.RdsClient;
import software.amazon.awssdk.services.rds.model.DeleteDbInstanceRequest;
import software.amazon.awssdk.services.rds.model.DeleteDbInstanceResponse;
import software.amazon.awssdk.services.rds.model.RdsException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeleteDBInstance {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <dbInstanceIdentifier>\s
```

```
                Where:
                    dbInstanceIdentifier - The database instance identifier\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String dbInstanceIdentifier = args[0];
        Region region = Region.US_WEST_2;
        RdsClient rdsClient = RdsClient.builder()
                .region(region)
                .build();

        deleteDatabaseInstance(rdsClient, dbInstanceIdentifier);
        rdsClient.close();
    }

    public static void deleteDatabaseInstance(RdsClient rdsClient, String
 dbInstanceIdentifier) {
        try {
            DeleteDbInstanceRequest deleteDbInstanceRequest =
 DeleteDbInstanceRequest.builder()
                    .dbInstanceIdentifier(dbInstanceIdentifier)
                    .deleteAutomatedBackups(true)
                    .skipFinalSnapshot(true)
                    .build();

            DeleteDbInstanceResponse response =
 rdsClient.deleteDBInstance(deleteDbInstanceRequest);
            System.out.print("The status of the database is " +
 response.dbInstance().dbInstanceStatus());

        } catch (RdsException e) {
            System.out.println(e.getLocalizedMessage());
            System.exit(1);
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon SDK for Java 2.x API 参考》中的 DeleteDBInstance。

Kotlin

适用于 Kotlin 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
suspend fun deleteDatabaseInstance(dbInstanceIdentifierVal: String?) {
    val deleteDbInstanceRequest =
        DeleteDbInstanceRequest {
            dbInstanceIdentifier = dbInstanceIdentifierVal
            deleteAutomatedBackups = true
            skipFinalSnapshot = true
        }

    RdsClient { region = "us-west-2" }.use { rdsClient ->
        val response = rdsClient.deleteDbInstance(deleteDbInstanceRequest)
        print("The status of the database is
 ${response.dbInstance?.dbInstanceStatus}")
    }
}
```

- 有关 API 详细信息，请参阅《Amazon SDK for Kotlin API 参考》中的 DeleteDBInstance。

PHP

适用于 PHP 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
require __DIR__ . '/vendor/autoload.php';

use Aws\Exception\AwsException;


//Create an RDSClient
$rdsClient = new Aws\Rds\RdsClient([
    'region' => 'us-east-1'
]);

$dbIdentifier = '<<{{db-identifier}}>>';

try {
    $result = $rdsClient->deleteDBInstance([
        'DBInstanceIdentifier' => $dbIdentifier,
    ]);
    var_dump($result);
} catch (AwsException $e) {
    echo $e->getMessage();
    echo "\n";
}
```

- 有关 API 详细信息，请参阅《适用于 PHP 的 Amazon SDK API 参考》中的 DeleteDBInstance。

Python

适用于 Python 的 SDK（Boto3）

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
class InstanceWrapper:
    """Encapsulates Amazon RDS DB instance actions."""
```

```
    def __init__(self, rds_client):
        """
        :param rds_client: A Boto3 Amazon RDS client.
        """
        self.rds_client = rds_client

    @classmethod
    def from_client(cls):
        """
        Instantiates this class from a Boto3 client.
        """
        rds_client = boto3.client("rds")
        return cls(rds_client)


    def delete_db_instance(self, instance_id):
        """
        Deletes a DB instance.

        :param instance_id: The ID of the DB instance to delete.
        :return: Data about the deleted DB instance.
        """
        try:
            response = self.rds_client.delete_db_instance(
                DBInstanceIdentifier=instance_id,
                SkipFinalSnapshot=True,
                DeleteAutomatedBackups=True,
            )
            db_inst = response["DBInstance"]
        except ClientError as err:
            logger.error(
                "Couldn't delete DB instance %s. Here's why: %s: %s",
                instance_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return db_inst
```

- 有关 API 详细信息，请参阅《适用于 Python 的 Amazon SDK（Boto3）API 参考》中的 DeleteDBInstance。

有关 Amazon SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 Amazon SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DeleteDBParameterGroup** 与 Amazon SDK 或 CLI 配合使用

以下代码示例演示如何使用 DeleteDBParameterGroup。

操作示例是大型程序的代码摘录，必须在上下文中运行。在以下代码示例中，您可以查看此操作的上下文：

- 了解基础知识

.NET

适用于 .NET 的 Amazon SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
    /// <summary>
    /// Delete a DB parameter group. The group cannot be a default DB parameter
 group
    /// or be associated with any DB instances.
    /// </summary>
    /// <param name="name">Name of the DB parameter group.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> DeleteDBParameterGroup(string name)
    {
        var response = await _amazonRDS.DeleteDBParameterGroupAsync(
            new DeleteDBParameterGroupRequest()
            {
                DBParameterGroupName = name,
            });
```

```
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- 有关 API 详细信息，请参阅《适用于 .NET 的 Amazon SDK API 参考》中的 DeleteDBParameterGroup。

C++

SDK for C++

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::RDS::RDSClient client(clientConfig);

        Aws::RDS::Model::DeleteDBParameterGroupRequest request;
        request.SetDBParameterGroupName(parameterGroupName);

        Aws::RDS::Model::DeleteDBParameterGroupOutcome outcome =
                client.DeleteDBParameterGroup(request);

        if (outcome.IsSuccess()) {
            std::cout << "The DB parameter group was successfully deleted."
                      << std::endl;
        }
        else {
            std::cerr << "Error with RDS::DeleteDBParameterGroup. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            result = false;
        }
```

- 有关 API 详细信息，请参阅《适用于 C++ 的 Amazon SDK API 参考》中的 DeleteDBParameterGroup。

CLI

Amazon CLI

删除数据库参数组

以下 command 示例删除一个数据库参数组。

```
aws rds delete-db-parameter-group \
    --db-parameter-group-name mydbparametergroup
```

此命令不生成任何输出。

有关更多信息，请参阅《Amazon RDS 用户指南》中的使用数据库参数组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DeleteDBParameterGroup。

Go

适用于 Go V2 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import (
 "context"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/rds"
 "github.com/aws/aws-sdk-go-v2/service/rds/types"
)
```

```
type DbInstances struct {
 RdsClient *rds.Client
}



// DeleteParameterGroup deletes the named DB parameter group.
func (instances *DbInstances) DeleteParameterGroup(ctx context.Context,
 parameterGroupName string) error {
 _, err := instances.RdsClient.DeleteDBParameterGroup(ctx,
  &rds.DeleteDBParameterGroupInput{
   DBParameterGroupName: aws.String(parameterGroupName),
  })
 if err != nil {
  log.Printf("Couldn't delete parameter group %v: %v\n", parameterGroupName, err)
  return err
 } else {
  return nil
 }
}
```

- 有关 API 详细信息，请参阅《适用于 Go 的 Amazon SDK API 参考》中的 [DeleteDBParameterGroup](#)。

Java

适用于 Java 的 SDK 2.x

> **Note**
>
> 查看 GitHub，了解更多信息。在 [Amazon 代码示例存储库](#)中查找完整示例，了解如何进行设置和运行。

```
    // Delete the parameter group after database has been deleted.
    // An exception is thrown if you attempt to delete the para group while
 database
    // exists.
```

```
    public static void deleteParaGroup(RdsClient rdsClient, String dbGroupName,
 String dbARN)
            throws InterruptedException {
        try {
            boolean isDataDel = false;
            boolean didFind;
            String instanceARN;

            // Make sure that the database has been deleted.
            while (!isDataDel) {
                DescribeDbInstancesResponse response =
 rdsClient.describeDBInstances();
                List<DBInstance> instanceList = response.dbInstances();
                int listSize = instanceList.size();
                didFind = false;
                int index = 1;
                for (DBInstance instance : instanceList) {
                    instanceARN = instance.dbInstanceArn();
                    if (instanceARN.compareTo(dbARN) == 0) {
                        System.out.println(dbARN + " still exists");
                        didFind = true;
                    }
                    if ((index == listSize) && (!didFind)) {
                        // Went through the entire list and did not find the
 database ARN.
                        isDataDel = true;
                    }
                    Thread.sleep(sleepTime * 1000);
                    index++;
                }
            }

            // Delete the para group.
            DeleteDbParameterGroupRequest parameterGroupRequest =
 DeleteDbParameterGroupRequest.builder()
                    .dbParameterGroupName(dbGroupName)
                    .build();

            rdsClient.deleteDBParameterGroup(parameterGroupRequest);
            System.out.println(dbGroupName + " was deleted.");

        } catch (RdsException e) {
            System.out.println(e.getLocalizedMessage());
            System.exit(1);
```

```
        }
    }
```

- 有关 API 详细信息，请参阅《Amazon SDK for Java 2.x API 参考》中的 DeleteDBParameterGroup。

Python

适用于 Python 的 SDK（Boto3）

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
class InstanceWrapper:
    """Encapsulates Amazon RDS DB instance actions."""

    def __init__(self, rds_client):
        """
        :param rds_client: A Boto3 Amazon RDS client.
        """
        self.rds_client = rds_client

    @classmethod
    def from_client(cls):
        """
        Instantiates this class from a Boto3 client.
        """
        rds_client = boto3.client("rds")
        return cls(rds_client)


    def delete_parameter_group(self, parameter_group_name):
        """
        Deletes a DB parameter group.

        :param parameter_group_name: The name of the parameter group to delete.
        :return: Data about the parameter group.
```

```
        """
        try:
            self.rds_client.delete_db_parameter_group(
                DBParameterGroupName=parameter_group_name
            )
        except ClientError as err:
            logger.error(
                "Couldn't delete parameter group %s. Here's why: %s: %s",
                parameter_group_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- 有关 API 详细信息，请参阅《适用于 Python 的 Amazon SDK（Boto3）API 参考》中的 DeleteDBParameterGroup。

有关 Amazon SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 Amazon SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 `DescribeAccountAttributes` 与 Amazon SDK 或 CLI 配合使用

以下代码示例演示如何使用 `DescribeAccountAttributes`。

CLI

Amazon CLI

描述账户属性

以下 `describe-account-attributes` 示例将检索当前 Amazon 账户的属性。

```
aws rds describe-account-attributes
```

输出：

```
{
    "AccountQuotas": [
        {
            "Max": 40,
```

```
            "Used": 4,
            "AccountQuotaName": "DBInstances"
        },
        {
            "Max": 40,
            "Used": 0,
            "AccountQuotaName": "ReservedDBInstances"
        },
        {
            "Max": 100000,
            "Used": 40,
            "AccountQuotaName": "AllocatedStorage"
        },
        {
            "Max": 25,
            "Used": 0,
            "AccountQuotaName": "DBSecurityGroups"
        },
        {
            "Max": 20,
            "Used": 0,
            "AccountQuotaName": "AuthorizationsPerDBSecurityGroup"
        },
        {
            "Max": 50,
            "Used": 1,
            "AccountQuotaName": "DBParameterGroups"
        },
        {
            "Max": 100,
            "Used": 3,
            "AccountQuotaName": "ManualSnapshots"
        },
        {
            "Max": 20,
            "Used": 0,
            "AccountQuotaName": "EventSubscriptions"
        },
        {
            "Max": 50,
            "Used": 1,
            "AccountQuotaName": "DBSubnetGroups"
        },
        {
```

```
                "Max": 20,
                "Used": 1,
                "AccountQuotaName": "OptionGroups"
            },
            {
                "Max": 20,
                "Used": 6,
                "AccountQuotaName": "SubnetsPerDBSubnetGroup"
            },
            {
                "Max": 5,
                "Used": 0,
                "AccountQuotaName": "ReadReplicasPerMaster"
            },
            {
                "Max": 40,
                "Used": 1,
                "AccountQuotaName": "DBClusters"
            },
            {
                "Max": 50,
                "Used": 0,
                "AccountQuotaName": "DBClusterParameterGroups"
            },
            {
                "Max": 5,
                "Used": 0,
                "AccountQuotaName": "DBClusterRoles"
            }
        ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeAccountAttributes。

Java

适用于 Java 的 SDK 2.x

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.rds.RdsClient;
import software.amazon.awssdk.services.rds.model.AccountQuota;
import software.amazon.awssdk.services.rds.model.RdsException;
import
 software.amazon.awssdk.services.rds.model.DescribeAccountAttributesResponse;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DescribeAccountAttributes {
    public static void main(String[] args) {
        Region region = Region.US_WEST_2;
        RdsClient rdsClient = RdsClient.builder()
                .region(region)
                .build();

        getAccountAttributes(rdsClient);
        rdsClient.close();
    }

    public static void getAccountAttributes(RdsClient rdsClient) {
        try {
            DescribeAccountAttributesResponse response =
 rdsClient.describeAccountAttributes();
            List<AccountQuota> quotasList = response.accountQuotas();
            for (AccountQuota quotas : quotasList) {
                System.out.println("Name is: " + quotas.accountQuotaName());
                System.out.println("Max value is " + quotas.max());
            }

        } catch (RdsException e) {
            System.out.println(e.getLocalizedMessage());
            System.exit(1);
        }
    }
```

```
}
```

- 有关 API 详细信息，请参阅《Amazon SDK for Java 2.x API 参考》中的 DescribeAccountAttributes。

Kotlin

适用于 Kotlin 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
suspend fun getAccountAttributes() {
    RdsClient { region = "us-west-2" }.use { rdsClient ->
        val response =
 rdsClient.describeAccountAttributes(DescribeAccountAttributesRequest {})
        response.accountQuotas?.forEach { quotas ->
            val response = response.accountQuotas
            println("Name is: ${quotas.accountQuotaName}")
            println("Max value is ${quotas.max}")
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon SDK for Kotlin API 参考》中的 DescribeAccountAttributes。

有关 Amazon SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 Amazon SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 `DescribeDBEngineVersions` 与 Amazon SDK 或 CLI 配合使用

以下代码示例演示如何使用 `DescribeDBEngineVersions`。

操作示例是大型程序的代码摘录，必须在上下文中运行。在以下代码示例中，您可以查看此操作的上下文：

- 了解基础知识

.NET

适用于 .NET 的 Amazon SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
    /// <summary>
    /// Get a list of DB engine versions for a particular DB engine.
    /// </summary>
    /// <param name="engine">Name of the engine.</param>
    /// <param name="dbParameterGroupFamily">Optional parameter group family
name.</param>
    /// <returns>List of DBEngineVersions.</returns>
    public async Task<List<DBEngineVersion>> DescribeDBEngineVersions(string
engine,
        string dbParameterGroupFamily = null)
    {
        var response = await _amazonRDS.DescribeDBEngineVersionsAsync(
            new DescribeDBEngineVersionsRequest()
            {
                Engine = engine,
                DBParameterGroupFamily = dbParameterGroupFamily
            });
        return response.DBEngineVersions;
    }
```

- 有关 API 详细信息，请参阅《适用于 .NET 的 Amazon SDK API 参考》中的 DescribeDBEngineVersions。

C++

SDK for C++

> **Note**
>
> 查看 GitHub，了解更多信息。在 [Amazon 代码示例存储库](#)中查找完整示例，了解如何进行设置和运行。

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::RDS::RDSClient client(clientConfig);


//! Routine which gets available DB engine versions for an engine name and
//! an optional parameter group family.
/*!
 \sa getDBEngineVersions()
 \param engineName: A DB engine name.
 \param parameterGroupFamily: A parameter group family name, ignored if empty.
 \param engineVersionsResult: Vector of 'DBEngineVersion' objects returned by the
 routine.
 \param client: 'RDSClient' instance.
 \return bool: Successful completion.
 */
bool AwsDoc::RDS::getDBEngineVersions(const Aws::String &engineName,
                                      const Aws::String &parameterGroupFamily,

 Aws::Vector<Aws::RDS::Model::DBEngineVersion> &engineVersionsResult,
                                      const Aws::RDS::RDSClient &client) {
    Aws::RDS::Model::DescribeDBEngineVersionsRequest request;
    request.SetEngine(engineName);
    if (!parameterGroupFamily.empty()) {
        request.SetDBParameterGroupFamily(parameterGroupFamily);
    }

    engineVersionsResult.clear();
    Aws::String marker; // Used for pagination.
```

```
    do {
        if (!marker.empty()) {
            request.SetMarker(marker);
        }


        Aws::RDS::Model::DescribeDBEngineVersionsOutcome outcome =
                client.DescribeDBEngineVersions(request);

        if (outcome.IsSuccess()) {
            auto &engineVersions = outcome.GetResult().GetDBEngineVersions();
            engineVersionsResult.insert(engineVersionsResult.end(),
 engineVersions.begin(),
                                        engineVersions.end());
            marker = outcome.GetResult().GetMarker();
        }
        else {
            std::cerr << "Error with RDS::DescribeDBEngineVersionsRequest. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            return false;
        }

    } while (!marker.empty());


    return true;
}
```

- 有关 API 详细信息，请参阅《适用于 C++ 的 Amazon SDK API 参考》中的 DescribeDBEngineVersions。

CLI

Amazon CLI

描述 MySQL 数据库引擎的数据库引擎版本

以下 `describe-db-engine-versions` 示例显示了有关指定数据库引擎的每个数据库引擎版本的详细信息。

```
aws rds describe-db-engine-versions \
```

```
    --engine mysql
```

输出：


```
{
    "DBEngineVersions": [
        {
            "Engine": "mysql",
            "EngineVersion": "5.5.46",
            "DBParameterGroupFamily": "mysql5.5",
            "DBEngineDescription": "MySQL Community Edition",
            "DBEngineVersionDescription": "MySQL 5.5.46",
            "ValidUpgradeTarget": [
                {
                    "Engine": "mysql",
                    "EngineVersion": "5.5.53",
                    "Description": "MySQL 5.5.53",
                    "AutoUpgrade": false,
                    "IsMajorVersionUpgrade": false
                },
                {
                    "Engine": "mysql",
                    "EngineVersion": "5.5.54",
                    "Description": "MySQL 5.5.54",
                    "AutoUpgrade": false,
                    "IsMajorVersionUpgrade": false
                },
                {
                    "Engine": "mysql",
                    "EngineVersion": "5.5.57",
                    "Description": "MySQL 5.5.57",
                    "AutoUpgrade": false,
                    "IsMajorVersionUpgrade": false
                },
                ...some output truncated...
            ]
        }
```


有关更多信息，请参阅《Amazon RDS 用户指南》中的[什么是 Amazon Relational Database Service（Amazon RDS）？](#)。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 [DescribeDBEngineVersions](#)。

Go

适用于 Go V2 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import (
 "context"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/rds"
 "github.com/aws/aws-sdk-go-v2/service/rds/types"
)

type DbInstances struct {
 RdsClient *rds.Client
}



// GetEngineVersions gets database engine versions that are available for the
 specified engine
// and parameter group family.
func (instances *DbInstances) GetEngineVersions(ctx context.Context, engine
 string, parameterGroupFamily string) (
 []types.DBEngineVersion, error) {
 output, err := instances.RdsClient.DescribeDBEngineVersions(ctx,
  &rds.DescribeDBEngineVersionsInput{
   Engine:                 aws.String(engine),
   DBParameterGroupFamily: aws.String(parameterGroupFamily),
  })
 if err != nil {
  log.Printf("Couldn't get engine versions for %v: %v\n", engine, err)
  return nil, err
 } else {
```

```
        return output.DBEngineVersions, nil
    }
}
```

- 有关 API 详细信息，请参阅《适用于 Go 的 Amazon SDK API 参考》中的 DescribeDBEngineVersions。

Java

适用于 Java 的 SDK 2.x

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
    public static void describeDBEngines(RdsClient rdsClient) {
        try {
            DescribeDbEngineVersionsRequest engineVersionsRequest =
DescribeDbEngineVersionsRequest.builder()
                    .defaultOnly(true)
                    .engine("mysql")
                    .maxRecords(20)
                    .build();

            DescribeDbEngineVersionsResponse response =
rdsClient.describeDBEngineVersions(engineVersionsRequest);
            List<DBEngineVersion> engines = response.dbEngineVersions();

            // Get all DBEngineVersion objects.
            for (DBEngineVersion engineOb : engines) {
                System.out.println("The name of the DB parameter group family for
the database engine is "
                        + engineOb.dbParameterGroupFamily());
                System.out.println("The name of the database engine " +
engineOb.engine());
                System.out.println("The version number of the database engine " +
engineOb.engineVersion());
```

```
            }

        } catch (RdsException e) {
            System.out.println(e.getLocalizedMessage());
            System.exit(1);
        }
    }
```

- 有关 API 详细信息，请参阅《Amazon SDK for Java 2.x API 参考》中的 DescribeDBEngineVersions。

Python

适用于 Python 的 SDK（Boto3）

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
class InstanceWrapper:
    """Encapsulates Amazon RDS DB instance actions."""

    def __init__(self, rds_client):
        """
        :param rds_client: A Boto3 Amazon RDS client.
        """
        self.rds_client = rds_client

    @classmethod
    def from_client(cls):
        """
        Instantiates this class from a Boto3 client.
        """
        rds_client = boto3.client("rds")
        return cls(rds_client)


    def get_engine_versions(self, engine, parameter_group_family=None):
```

```
        """
        Gets database engine versions that are available for the specified engine
        and parameter group family.

        :param engine: The database engine to look up.
        :param parameter_group_family: When specified, restricts the returned
 list of
                                       engine versions to those that are
 compatible with
                                       this parameter group family.
        :return: The list of database engine versions.
        """
        try:
            kwargs = {"Engine": engine}
            if parameter_group_family is not None:
                kwargs["DBParameterGroupFamily"] = parameter_group_family
            response = self.rds_client.describe_db_engine_versions(**kwargs)
            versions = response["DBEngineVersions"]
        except ClientError as err:
            logger.error(
                "Couldn't get engine versions for %s. Here's why: %s: %s",
                engine,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return versions
```

- 有关 API 详细信息，请参阅《适用于 Python 的 Amazon SDK（Boto3）API 参考》中的 DescribeDBEngineVersions。

有关 Amazon SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 Amazon SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 `DescribeDBInstances` 与 Amazon SDK 或 CLI 配合使用

以下代码示例演示如何使用 `DescribeDBInstances`。

操作示例是大型程序的代码摘录，必须在上下文中运行。在以下代码示例中，您可以查看此操作的上下文：

- 了解基础知识

.NET

适用于 .NET 的 Amazon SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
    /// <summary>
    /// Returns a list of DB instances.
    /// </summary>
    /// <param name="dbInstanceIdentifier">Optional name of a specific DB
 instance.</param>
    /// <returns>List of DB instances.</returns>
    public async Task<List<DBInstance>> DescribeDBInstances(string
 dbInstanceIdentifier = null)
    {
        var results = new List<DBInstance>();
        var instancesPaginator = _amazonRDS.Paginators.DescribeDBInstances(
            new DescribeDBInstancesRequest
            {
                DBInstanceIdentifier = dbInstanceIdentifier
            });
        // Get the entire list using the paginator.
        await foreach (var instances in instancesPaginator.DBInstances)
        {
            results.Add(instances);
        }
        return results;
    }
```

- 有关 API 详细信息，请参阅《适用于 .NET 的 Amazon SDK API 参考》中的 DescribeDBInstances。

C++

SDK for C++

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::RDS::RDSClient client(clientConfig);


//! Routine which gets a DB instance description.
/*!
 \sa describeDBInstance()
 \param dbInstanceIdentifier: A DB instance identifier.
 \param instanceResult: The 'DBInstance' object containing the description.
 \param client: 'RDSClient' instance.
 \return bool: Successful completion.
 */
bool AwsDoc::RDS::describeDBInstance(const Aws::String &dbInstanceIdentifier,
                                     Aws::RDS::Model::DBInstance &instanceResult,
                                     const Aws::RDS::RDSClient &client) {
    Aws::RDS::Model::DescribeDBInstancesRequest request;
    request.SetDBInstanceIdentifier(dbInstanceIdentifier);

    Aws::RDS::Model::DescribeDBInstancesOutcome outcome =
            client.DescribeDBInstances(request);

    bool result = true;
    if (outcome.IsSuccess()) {
        instanceResult = outcome.GetResult().GetDBInstances()[0];
    }
```

```
    else if (outcome.GetError().GetErrorType() !=
             Aws::RDS::RDSErrors::D_B_INSTANCE_NOT_FOUND_FAULT) {
        result = false;
        std::cerr << "Error with RDS::DescribeDBInstances. "
                  << outcome.GetError().GetMessage()
                  << std::endl;
    }
        // This example does not log an error if the DB instance does not exist.
        // Instead, instanceResult is set to empty.
    else {
        instanceResult = Aws::RDS::Model::DBInstance();
    }

    return result;
}
```

- 有关 API 详细信息，请参阅《适用于 C++ 的 Amazon SDK API 参考》中的 DescribeDBInstances。

CLI

Amazon CLI

描述数据库实例

以下 `describe-db-instances` 示例将检索有关指定数据库实例的详细信息。

```
aws rds describe-db-instances \
    --db-instance-identifier mydbinstancecf
```

输出：

```
{
    "DBInstances": [
        {
            "DBInstanceIdentifier": "mydbinstancecf",
            "DBInstanceClass": "db.t3.small",
            "Engine": "mysql",
            "DBInstanceStatus": "available",
            "MasterUsername": "masterawsuser",
            "Endpoint": {
```

```
                "Address": "mydbinstancecf.abcexample.us-
east-1.rds.amazonaws.com",
                "Port": 3306,
                "HostedZoneId": "Z2R2ITUGPM61AM"
            },
            ...some output truncated...
        }
    ]
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDBInstances。

Go

适用于 Go V2 的 SDK

> **Note**
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import (
 "context"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/rds"
 "github.com/aws/aws-sdk-go-v2/service/rds/types"
)

type DbInstances struct {
 RdsClient *rds.Client
}



// GetInstance gets data about a DB instance.
func (instances *DbInstances) GetInstance(ctx context.Context, instanceName
 string) (
```

```
 *types.DBInstance, error) {
 output, err := instances.RdsClient.DescribeDBInstances(ctx,
  &rds.DescribeDBInstancesInput{
   DBInstanceIdentifier: aws.String(instanceName),
  })
 if err != nil {
  var notFoundError *types.DBInstanceNotFoundFault
  if errors.As(err, &notFoundError) {
   log.Printf("DB instance %v does not exist.\n", instanceName)
   err = nil
  } else {
   log.Printf("Couldn't get instance %v: %v\n", instanceName, err)
  }
  return nil, err
 } else {
  return &output.DBInstances[0], nil
 }
}
```

- 有关 API 详细信息，请参阅《适用于 Go 的 Amazon SDK API 参考》中的 DescribeDBInstances。

Java

适用于 Java 的 SDK 2.x

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.rds.RdsClient;
import software.amazon.awssdk.services.rds.model.DescribeDbInstancesResponse;
import software.amazon.awssdk.services.rds.model.DBInstance;
import software.amazon.awssdk.services.rds.model.RdsException;
import java.util.List;
```

```
/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DescribeDBInstances {

    public static void main(String[] args) {
        Region region = Region.US_EAST_1;
        RdsClient rdsClient = RdsClient.builder()
                .region(region)
                .build();

        describeInstances(rdsClient);
        rdsClient.close();
    }

    public static void describeInstances(RdsClient rdsClient) {
        try {
            DescribeDbInstancesResponse response =
 rdsClient.describeDBInstances();
            List<DBInstance> instanceList = response.dbInstances();
            for (DBInstance instance : instanceList) {
                System.out.println("Instance ARN is: " +
 instance.dbInstanceArn());
                System.out.println("The Engine is " + instance.engine());
                System.out.println("Connection endpoint is" +
 instance.endpoint().address());
            }

        } catch (RdsException e) {
            System.out.println(e.getLocalizedMessage());
            System.exit(1);
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon SDK for Java 2.x API 参考》中的 DescribeDBInstances。

Kotlin

适用于 Kotlin 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
suspend fun describeInstances() {
    RdsClient { region = "us-west-2" }.use { rdsClient ->
        val response = rdsClient.describeDbInstances(DescribeDbInstancesRequest
 {})
        response.dbInstances?.forEach { instance ->
            println("Instance Identifier is ${instance.dbInstanceIdentifier}")
            println("The Engine is ${instance.engine}")
            println("Connection endpoint is ${instance.endpoint?.address}")
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon SDK for Kotlin API 参考》中的 DescribeDBInstances。

PHP

适用于 PHP 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
require __DIR__ . '/vendor/autoload.php';

use Aws\Exception\AwsException;


//Create an RDSClient
$rdsClient = new Aws\Rds\RdsClient([
    'region' => 'us-east-2'
]);

try {
    $result = $rdsClient->describeDBInstances();
    foreach ($result['DBInstances'] as $instance) {
        print('<p>DB Identifier: ' . $instance['DBInstanceIdentifier']);
        print('<br />Endpoint: ' . $instance['Endpoint']["Address"]
            . ':' . $instance['Endpoint']["Port"]);
        print('<br />Current Status: ' . $instance["DBInstanceStatus"]);
        print('</p>');
    }
    print(" Raw Result ");
    var_dump($result);
} catch (AwsException $e) {
    echo $e->getMessage();
    echo "\n";
}
```

- 有关 API 详细信息，请参阅《适用于 PHP 的 Amazon SDK API 参考》中的 DescribeDBInstances。

Python

适用于 Python 的 SDK（Boto3）

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
class InstanceWrapper:
    """Encapsulates Amazon RDS DB instance actions."""

    def __init__(self, rds_client):
        """
        :param rds_client: A Boto3 Amazon RDS client.
        """
        self.rds_client = rds_client

    @classmethod
    def from_client(cls):
        """
        Instantiates this class from a Boto3 client.
        """
        rds_client = boto3.client("rds")
        return cls(rds_client)


    def get_db_instance(self, instance_id):
        """
        Gets data about a DB instance.

        :param instance_id: The ID of the DB instance to retrieve.
        :return: The retrieved DB instance.
        """
        try:
            response = self.rds_client.describe_db_instances(
                DBInstanceIdentifier=instance_id
            )
            db_inst = response["DBInstances"][0]
        except ClientError as err:
            if err.response["Error"]["Code"] == "DBInstanceNotFound":
                logger.info("Instance %s does not exist.", instance_id)
            else:
                logger.error(
                    "Couldn't get DB instance %s. Here's why: %s: %s",
                    instance_id,
                    err.response["Error"]["Code"],
                    err.response["Error"]["Message"],
                )
                raise
        else:
            return db_inst
```

- 有关 API 详细信息，请参阅《适用于 Python 的 Amazon SDK（Boto3）API 参考》中的 DescribeDBInstances。

Ruby

适用于 Ruby 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
require 'aws-sdk-rds' # v2: require 'aws-sdk'

# List all Amazon Relational Database Service (Amazon RDS) DB instances.
#
# @param rds_resource [Aws::RDS::Resource] An SDK for Ruby Amazon RDS resource.
# @return [Array, nil] List of all DB instances, or nil if error.
def list_instances(rds_resource)
  db_instances = []
  rds_resource.db_instances.each do |i|
    db_instances.append({
                          "name": i.id,
                          "status": i.db_instance_status
                        })
  end
  db_instances
rescue Aws::Errors::ServiceError => e
  puts "Couldn't list instances:\n#{e.message}"
end
```

- 有关 API 详细信息，请参阅《适用于 Ruby 的 Amazon SDK API 参考》中的 DescribeDBInstances。

有关 Amazon SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 Amazon SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 DescribeDBParameterGroups 与 Amazon SDK 或 CLI 配合使用

以下代码示例演示如何使用 DescribeDBParameterGroups。

操作示例是大型程序的代码摘录，必须在上下文中运行。在以下代码示例中，您可以查看此操作的上下文：

- 了解基础知识

.NET

适用于 .NET 的 Amazon SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
    /// <summary>
    /// Get descriptions of DB parameter groups.
    /// </summary>
    /// <param name="name">Optional name of the DB parameter group to describe.</
param>
    /// <returns>The list of DB parameter group descriptions.</returns>
    public async Task<List<DBParameterGroup>> DescribeDBParameterGroups(string
 name = null)
    {
        var response = await _amazonRDS.DescribeDBParameterGroupsAsync(
            new DescribeDBParameterGroupsRequest()
            {
                DBParameterGroupName = name
            });
        return response.DBParameterGroups;
    }
```

- 有关 API 详细信息，请参阅《适用于 .NET 的 Amazon SDK API 参考》中的 DescribeDBParameterGroups。

C++

SDK for C++

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::RDS::RDSClient client(clientConfig);

        Aws::RDS::Model::DescribeDBParameterGroupsRequest request;
        request.SetDBParameterGroupName(PARAMETER_GROUP_NAME);

        Aws::RDS::Model::DescribeDBParameterGroupsOutcome outcome =
                client.DescribeDBParameterGroups(request);

        if (outcome.IsSuccess()) {
            std::cout << "DB parameter group named '" <<
                      PARAMETER_GROUP_NAME << "' already exists." << std::endl;
            dbParameterGroupFamily = outcome.GetResult().GetDBParameterGroups()
[0].GetDBParameterGroupFamily();
        }

        else {
            std::cerr << "Error with RDS::DescribeDBParameterGroups. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            return false;
        }
```

- 有关 API 详细信息，请参阅《适用于 C++ 的 Amazon SDK API 参考》中的 DescribeDBParameterGroups。

CLI

Amazon CLI

描述数据库参数组

以下 describe-db-parameter-groups 示例将检索有关数据库参数组的详细信息。

```
aws rds describe-db-parameter-groups
```

输出：

```
{
    "DBParameterGroups": [
        {
            "DBParameterGroupName": "default.aurora-mysql5.7",
            "DBParameterGroupFamily": "aurora-mysql5.7",
            "Description": "Default parameter group for aurora-mysql5.7",
            "DBParameterGroupArn": "arn:aws:rds:us-
east-1:123456789012:pg:default.aurora-mysql5.7"
        },
        {
            "DBParameterGroupName": "default.aurora-postgresql9.6",
            "DBParameterGroupFamily": "aurora-postgresql9.6",
            "Description": "Default parameter group for aurora-postgresql9.6",
            "DBParameterGroupArn": "arn:aws:rds:us-
east-1:123456789012:pg:default.aurora-postgresql9.6"
        },
        {
            "DBParameterGroupName": "default.aurora5.6",
            "DBParameterGroupFamily": "aurora5.6",
            "Description": "Default parameter group for aurora5.6",
            "DBParameterGroupArn": "arn:aws:rds:us-
east-1:123456789012:pg:default.aurora5.6"
        },
        {
            "DBParameterGroupName": "default.mariadb10.1",
            "DBParameterGroupFamily": "mariadb10.1",
            "Description": "Default parameter group for mariadb10.1",
```

```
                "DBParameterGroupArn": "arn:aws:rds:us-
east-1:123456789012:pg:default.mariadb10.1"
        },
        ...some output truncated...
    ]
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的使用数据库参数组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDBParameterGroups。

Go

适用于 Go V2 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import (
 "context"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/rds"
 "github.com/aws/aws-sdk-go-v2/service/rds/types"
)

type DbInstances struct {
 RdsClient *rds.Client
}



// GetParameterGroup gets a DB parameter group by name.
func (instances *DbInstances) GetParameterGroup(ctx context.Context,
 parameterGroupName string) (
 *types.DBParameterGroup, error) {
```

```
 output, err := instances.RdsClient.DescribeDBParameterGroups(
  ctx, &rds.DescribeDBParameterGroupsInput{
   DBParameterGroupName: aws.String(parameterGroupName),
  })
 if err != nil {
  var notFoundError *types.DBParameterGroupNotFoundFault
  if errors.As(err, &notFoundError) {
   log.Printf("Parameter group %v does not exist.\n", parameterGroupName)
   err = nil
  } else {
   log.Printf("Error getting parameter group %v: %v\n", parameterGroupName, err)
  }
  return nil, err
 } else {
  return &output.DBParameterGroups[0], err
 }
}
```

- 有关 API 详细信息，请参阅《适用于 Go 的 Amazon SDK API 参考》中的 DescribeDBParameterGroups。

Java

适用于 Java 的 SDK 2.x

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
   public static void describeDbParameterGroups(RdsClient rdsClient, String
dbGroupName) {
        try {
            DescribeDbParameterGroupsRequest groupsRequest =
DescribeDbParameterGroupsRequest.builder()
                    .dbParameterGroupName(dbGroupName)
                    .maxRecords(20)
                    .build();
```

```
            DescribeDbParameterGroupsResponse response =
rdsClient.describeDBParameterGroups(groupsRequest);
            List<DBParameterGroup> groups = response.dbParameterGroups();
            for (DBParameterGroup group : groups) {
                System.out.println("The group name is " +
group.dbParameterGroupName());
                System.out.println("The group description is " +
group.description());
            }

        } catch (RdsException e) {
            System.out.println(e.getLocalizedMessage());
            System.exit(1);
        }
    }
```

- 有关 API 详细信息，请参阅《Amazon SDK for Java 2.x API 参考》中的 [DescribeDBParameterGroups](#)。

Python

适用于 Python 的 SDK（Boto3）

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
class InstanceWrapper:
    """Encapsulates Amazon RDS DB instance actions."""

    def __init__(self, rds_client):
        """
        :param rds_client: A Boto3 Amazon RDS client.
        """
        self.rds_client = rds_client

    @classmethod
```

```
    def from_client(cls):
        """
        Instantiates this class from a Boto3 client.
        """
        rds_client = boto3.client("rds")
        return cls(rds_client)


    def get_parameter_group(self, parameter_group_name):
        """
        Gets a DB parameter group.

        :param parameter_group_name: The name of the parameter group to retrieve.
        :return: The parameter group.
        """
        try:
            response = self.rds_client.describe_db_parameter_groups(
                DBParameterGroupName=parameter_group_name
            )
            parameter_group = response["DBParameterGroups"][0]
        except ClientError as err:
            if err.response["Error"]["Code"] == "DBParameterGroupNotFound":
                logger.info("Parameter group %s does not exist.",
 parameter_group_name)
            else:
                logger.error(
                    "Couldn't get parameter group %s. Here's why: %s: %s",
                    parameter_group_name,
                    err.response["Error"]["Code"],
                    err.response["Error"]["Message"],
                )
                raise
        else:
            return parameter_group
```

- 有关 API 详细信息，请参阅《适用于 Python 的 Amazon SDK（Boto3）API 参考》中的 DescribeDBParameterGroups。

Ruby

适用于 Ruby 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
require 'aws-sdk-rds' # v2: require 'aws-sdk'

# List all Amazon Relational Database Service (Amazon RDS) parameter groups.
#
# @param rds_resource [Aws::RDS::Resource] An SDK for Ruby Amazon RDS resource.
# @return [Array, nil] List of all parameter groups, or nil if error.
def list_parameter_groups(rds_resource)
  parameter_groups = []
  rds_resource.db_parameter_groups.each do |p|
    parameter_groups.append({
                              "name": p.db_parameter_group_name,
                              "description": p.description
                            })
  end
  parameter_groups
rescue Aws::Errors::ServiceError => e
  puts "Couldn't list parameter groups:\n #{e.message}"
end
```

- 有关 API 详细信息，请参阅《适用于 Ruby 的 Amazon SDK API 参考》中的 DescribeDBParameterGroups。

有关 Amazon SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 Amazon SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 `DescribeDBParameters` 与 Amazon SDK 或 CLI 配合使用

以下代码示例演示如何使用 `DescribeDBParameters`。

操作示例是大型程序的代码摘录，必须在上下文中运行。在以下代码示例中，您可以查看此操作的上下文：

- 了解基础知识

.NET

适用于 .NET 的 Amazon SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
    /// <summary>
    /// Get a list of DB parameters from a specific parameter group.
    /// </summary>
    /// <param name="dbParameterGroupName">Name of a specific DB parameter
 group.</param>
    /// <param name="source">Optional source for selecting parameters.</param>
    /// <returns>List of parameter values.</returns>
    public async Task<List<Parameter>> DescribeDBParameters(string
 dbParameterGroupName, string source = null)
    {
        var results = new List<Parameter>();
        var paginateParameters = _amazonRDS.Paginators.DescribeDBParameters(
            new DescribeDBParametersRequest()
            {
                DBParameterGroupName = dbParameterGroupName,
                Source = source
            });
        // Get the entire list using the paginator.
        await foreach (var parameters in paginateParameters.Parameters)
        {
            results.Add(parameters);
        }
        return results;
    }
```

- 有关 API 详细信息，请参阅《适用于 .NET 的 Amazon SDK API 参考》中的 DescribeDBParameters。

C++

SDK for C++

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::RDS::RDSClient client(clientConfig);


//! Routine which gets DB parameters using the 'DescribeDBParameters' api.
/*!
 \sa getDBParameters()
 \param parameterGroupName: The name of the parameter group.
 \param namePrefix: Prefix string to filter results by parameter name.
 \param source: A source such as 'user', ignored if empty.
 \param parametersResult: Vector of 'Parameter' objects returned by the routine.
 \param client: 'RDSClient' instance.
 \return bool: Successful completion.
 */
bool AwsDoc::RDS::getDBParameters(const Aws::String &parameterGroupName,
                                  const Aws::String &namePrefix,
                                  const Aws::String &source,
                                  Aws::Vector<Aws::RDS::Model::Parameter>
 &parametersResult,
                                  const Aws::RDS::RDSClient &client) {
    Aws::String marker;
    do {
        Aws::RDS::Model::DescribeDBParametersRequest request;
```

```
        request.SetDBParameterGroupName(PARAMETER_GROUP_NAME);
        if (!marker.empty()) {
            request.SetMarker(marker);
        }
        if (!source.empty()) {
            request.SetSource(source);
        }

        Aws::RDS::Model::DescribeDBParametersOutcome outcome =
                client.DescribeDBParameters(request);

        if (outcome.IsSuccess()) {
            const Aws::Vector<Aws::RDS::Model::Parameter> &parameters =
                    outcome.GetResult().GetParameters();
            for (const Aws::RDS::Model::Parameter &parameter: parameters) {
                if (!namePrefix.empty()) {
                    if (parameter.GetParameterName().find(namePrefix) == 0) {
                        parametersResult.push_back(parameter);
                    }
                }
                else {
                    parametersResult.push_back(parameter);
                }
            }

            marker = outcome.GetResult().GetMarker();
        }
        else {
            std::cerr << "Error with RDS::DescribeDBParameters. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            return false;
        }
    } while (!marker.empty());

    return true;
}
```

- 有关 API 详细信息，请参阅《适用于 C++ 的 Amazon SDK API 参考》中的 DescribeDBParameters。

CLI

Amazon CLI

描述数据库参数组中的参数

以下 `describe-db-parameters` 示例将检索有关指定数据库参数组的详细信息。

```
aws rds describe-db-parameters \
    --db-parameter-group-name mydbpg
```

输出：

```
{
    "Parameters": [
        {
            "ParameterName": "allow-suspicious-udfs",
            "Description": "Controls whether user-defined functions that have
 only an xxx symbol for the main function can be loaded",
            "Source": "engine-default",
            "ApplyType": "static",
            "DataType": "boolean",
            "AllowedValues": "0,1",
            "IsModifiable": false,
            "ApplyMethod": "pending-reboot"
        },
        {
            "ParameterName": "auto_generate_certs",
            "Description": "Controls whether the server autogenerates SSL key and
 certificate files in the data directory, if they do not already exist.",
            "Source": "engine-default",
            "ApplyType": "static",
            "DataType": "boolean",
            "AllowedValues": "0,1",
            "IsModifiable": false,
            "ApplyMethod": "pending-reboot"
        },
        ...some output truncated...
    ]
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的使用数据库参数组。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDBParameters。

Go

适用于 Go V2 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import (
 "context"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/rds"
 "github.com/aws/aws-sdk-go-v2/service/rds/types"
)

type DbInstances struct {
 RdsClient *rds.Client
}



// GetParameters gets the parameters that are contained in a DB parameter group.
func (instances *DbInstances) GetParameters(ctx context.Context,
 parameterGroupName string, source string) (
 []types.Parameter, error) {

 var output *rds.DescribeDBParametersOutput
 var params []types.Parameter
 var err error
 parameterPaginator := rds.NewDescribeDBParametersPaginator(instances.RdsClient,
  &rds.DescribeDBParametersInput{
   DBParameterGroupName: aws.String(parameterGroupName),
   Source:               aws.String(source),
  })
```

```
 for parameterPaginator.HasMorePages() {
  output, err = parameterPaginator.NextPage(ctx)
  if err != nil {
   log.Printf("Couldn't get parameters for %v: %v\n", parameterGroupName, err)
   break
  } else {
   params = append(params, output.Parameters...)
  }
 }
 return params, err
}
```

- 有关 API 详细信息，请参阅《适用于 Go 的 Amazon SDK API 参考》中的 DescribeDBParameters。

Java

适用于 Java 的 SDK 2.x

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
    // Retrieve parameters in the group.
    public static void describeDbParameters(RdsClient rdsClient, String
 dbGroupName, int flag) {
        try {
            DescribeDbParametersRequest dbParameterGroupsRequest;
            if (flag == 0) {
                dbParameterGroupsRequest = DescribeDbParametersRequest.builder()
                        .dbParameterGroupName(dbGroupName)
                        .build();
            } else {
                dbParameterGroupsRequest = DescribeDbParametersRequest.builder()
                        .dbParameterGroupName(dbGroupName)
                        .source("user")
                        .build();
```

```
            }

            DescribeDbParametersResponse response =
rdsClient.describeDBParameters(dbParameterGroupsRequest);
            List<Parameter> dbParameters = response.parameters();
            String paraName;
            for (Parameter para : dbParameters) {
                // Only print out information about either auto_increment_offset
or
                // auto_increment_increment.
                paraName = para.parameterName();
                if ((paraName.compareTo("auto_increment_offset") == 0)
                        || (paraName.compareTo("auto_increment_increment ") ==
0)) {
                    System.out.println("*** The parameter name is  " + paraName);
                    System.out.println("*** The parameter value is  " +
para.parameterValue());
                    System.out.println("*** The parameter data type is " +
para.dataType());
                    System.out.println("*** The parameter description is " +
para.description());
                    System.out.println("*** The parameter allowed values  is " +
para.allowedValues());
                }
            }

        } catch (RdsException e) {
            System.out.println(e.getLocalizedMessage());
            System.exit(1);
        }
    }
```

- 有关 API 详细信息，请参阅《Amazon SDK for Java 2.x API 参考》中的 DescribeDBParameters。

Python

适用于 Python 的 SDK（Boto3）

> **Note**
>
> 查看 GitHub，了解更多信息。在 [Amazon 代码示例存储库](#)中查找完整示例，了解如何进行设置和运行。

```
class InstanceWrapper:
    """Encapsulates Amazon RDS DB instance actions."""

    def __init__(self, rds_client):
        """
        :param rds_client: A Boto3 Amazon RDS client.
        """
        self.rds_client = rds_client

    @classmethod
    def from_client(cls):
        """
        Instantiates this class from a Boto3 client.
        """
        rds_client = boto3.client("rds")
        return cls(rds_client)


    def get_parameters(self, parameter_group_name, name_prefix="", source=None):
        """
        Gets the parameters that are contained in a DB parameter group.

        :param parameter_group_name: The name of the parameter group to query.
        :param name_prefix: When specified, the retrieved list of parameters is
 filtered
                            to contain only parameters that start with this
 prefix.
        :param source: When specified, only parameters from this source are
 retrieved.
                       For example, a source of 'user' retrieves only parameters
 that
                       were set by a user.
```

```
        :return: The list of requested parameters.
        """
        try:
            kwargs = {"DBParameterGroupName": parameter_group_name}
            if source is not None:
                kwargs["Source"] = source
            parameters = []
            paginator = self.rds_client.get_paginator("describe_db_parameters")
            for page in paginator.paginate(**kwargs):
                parameters += [
                    p
                    for p in page["Parameters"]
                    if p["ParameterName"].startswith(name_prefix)
                ]
        except ClientError as err:
            logger.error(
                "Couldn't get parameters for %s. Here's why: %s: %s",
                parameter_group_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return parameters
```

- 有关 API 详细信息，请参阅《适用于 Python 的 Amazon SDK（Boto3）API 参考》中的 DescribeDBParameters。

Ruby

适用于 Ruby 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
require 'aws-sdk-rds' # v2: require 'aws-sdk'
```

```
# List all Amazon Relational Database Service (Amazon RDS) parameter groups.
#
# @param rds_resource [Aws::RDS::Resource] An SDK for Ruby Amazon RDS resource.
# @return [Array, nil] List of all parameter groups, or nil if error.
def list_parameter_groups(rds_resource)
  parameter_groups = []
  rds_resource.db_parameter_groups.each do |p|
    parameter_groups.append({
                              "name": p.db_parameter_group_name,
                              "description": p.description
                            })
  end
  parameter_groups
rescue Aws::Errors::ServiceError => e
  puts "Couldn't list parameter groups:\n #{e.message}"
end
```

- 有关 API 详细信息，请参阅《适用于 Ruby 的 Amazon SDK API 参考》中的 DescribeDBParameters。

有关 Amazon SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 Amazon SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DescribeDBSnapshots** 与 Amazon SDK 或 CLI 配合使用

以下代码示例演示如何使用 `DescribeDBSnapshots`。

操作示例是大型程序的代码摘录，必须在上下文中运行。在以下代码示例中，您可以查看此操作的上下文：

- 了解基础知识

.NET

适用于 .NET 的 Amazon SDK

Note

查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
    /// <summary>
    /// Return a list of DB snapshots for a particular DB instance.
    /// </summary>
    /// <param name="dbInstanceIdentifier">DB instance identifier.</param>
    /// <returns>List of DB snapshots.</returns>
    public async Task<List<DBSnapshot>> DescribeDBSnapshots(string
 dbInstanceIdentifier)
    {
        var results = new List<DBSnapshot>();
        var snapshotsPaginator = _amazonRDS.Paginators.DescribeDBSnapshots(
            new DescribeDBSnapshotsRequest()
            {
                DBInstanceIdentifier = dbInstanceIdentifier
            });

        // Get the entire list using the paginator.
        await foreach (var snapshots in snapshotsPaginator.DBSnapshots)
        {
            results.Add(snapshots);
        }
        return results;
    }
```

- 有关 API 详细信息，请参阅《适用于 .NET 的 Amazon SDK API 参考》中的 DescribeDBSnapshots。

C++

SDK for C++

> **Note**
>
> 查看 GitHub，了解更多信息。在 [Amazon 代码示例存储库](#)中查找完整示例，了解如何进行设置和运行。

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::RDS::RDSClient client(clientConfig);

            Aws::RDS::Model::DescribeDBSnapshotsRequest request;
            request.SetDBSnapshotIdentifier(snapshotID);

            Aws::RDS::Model::DescribeDBSnapshotsOutcome outcome =
                    client.DescribeDBSnapshots(request);

            if (outcome.IsSuccess()) {
                snapshot = outcome.GetResult().GetDBSnapshots()[0];
            }
            else {
                std::cerr << "Error with RDS::DescribeDBSnapshots. "
                          << outcome.GetError().GetMessage()
                          << std::endl;
                cleanUpResources(PARAMETER_GROUP_NAME, DB_INSTANCE_IDENTIFIER,
 client);
                return false;
            }
```

- 有关 API 详细信息，请参阅《适用于 C++ 的 Amazon SDK API 参考》中的 DescribeDBSnapshots。

CLI

Amazon CLI

示例 1：描述数据库实例的数据库快照

以下 `describe-db-snapshots` 示例将检索有关数据库实例的数据库快照的详细信息。

```
aws rds describe-db-snapshots \
    --db-snapshot-identifier mydbsnapshot
```

输出：

```
{
    "DBSnapshots": [
        {
            "DBSnapshotIdentifier": "mydbsnapshot",
            "DBInstanceIdentifier": "mysqldb",
            "SnapshotCreateTime": "2018-02-08T22:28:08.598Z",
            "Engine": "mysql",
            "AllocatedStorage": 20,
            "Status": "available",
            "Port": 3306,
            "AvailabilityZone": "us-east-1f",
            "VpcId": "vpc-6594f31c",
            "InstanceCreateTime": "2018-02-08T22:24:55.973Z",
            "MasterUsername": "mysqladmin",
            "EngineVersion": "5.6.37",
            "LicenseModel": "general-public-license",
            "SnapshotType": "manual",
            "OptionGroupName": "default:mysql-5-6",
            "PercentProgress": 100,
            "StorageType": "gp2",
            "Encrypted": false,
            "DBSnapshotArn": "arn:aws:rds:us-
east-1:123456789012:snapshot:mydbsnapshot",
            "IAMDatabaseAuthenticationEnabled": false,
            "ProcessorFeatures": [],
            "DbiResourceId": "db-AKIAIOSFODNN7EXAMPLE"
        }
    ]
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的创建数据库快照。

示例 2：查找手动拍摄的快照数量

以下 describe-db-snapshots 示例使用 --query 选项中的 length 运算符来返回在特定 Amazon 区域拍摄的手动快照的数量。

```
aws rds describe-db-snapshots \
    --snapshot-type manual \
    --query "length(*[].{DBSnapshots:SnapshotType})" \
    --region eu-central-1
```

输出：

```
35
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的创建数据库快照。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 DescribeDBSnapshots。

Go

适用于 Go V2 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import (
 "context"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/rds"
 "github.com/aws/aws-sdk-go-v2/service/rds/types"
)
```

```
type DbInstances struct {
 RdsClient *rds.Client
}



// GetSnapshot gets a DB instance snapshot.
func (instances *DbInstances) GetSnapshot(ctx context.Context, snapshotName
 string) (*types.DBSnapshot, error) {
 output, err := instances.RdsClient.DescribeDBSnapshots(ctx,
  &rds.DescribeDBSnapshotsInput{
   DBSnapshotIdentifier: aws.String(snapshotName),
  })
 if err != nil {
  log.Printf("Couldn't get snapshot %v: %v\n", snapshotName, err)
  return nil, err
 } else {
  return &output.DBSnapshots[0], nil
 }
}
```

- 有关 API 详细信息，请参阅《适用于 Go 的 Amazon SDK API 参考》中的 DescribeDBSnapshots。

Python

适用于 Python 的 SDK（Boto3）

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
class InstanceWrapper:
    """Encapsulates Amazon RDS DB instance actions."""

    def __init__(self, rds_client):
        """
```

```
        :param rds_client: A Boto3 Amazon RDS client.
        """
        self.rds_client = rds_client

    @classmethod
    def from_client(cls):
        """
        Instantiates this class from a Boto3 client.
        """
        rds_client = boto3.client("rds")
        return cls(rds_client)


    def get_snapshot(self, snapshot_id):
        """
        Gets a DB instance snapshot.

        :param snapshot_id: The ID of the snapshot to retrieve.
        :return: The retrieved snapshot.
        """
        try:
            response = self.rds_client.describe_db_snapshots(
                DBSnapshotIdentifier=snapshot_id
            )
            snapshot = response["DBSnapshots"][0]
        except ClientError as err:
            logger.error(
                "Couldn't get snapshot %s. Here's why: %s: %s",
                snapshot_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return snapshot
```

- 有关 API 详细信息，请参阅《适用于 Python 的 Amazon SDK（Boto3）API 参考》中的 [DescribeDBSnapshots](#)。

Ruby

适用于 Ruby 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
require 'aws-sdk-rds' # v2: require 'aws-sdk'

# List all Amazon Relational Database Service (Amazon RDS) DB instance
# snapshots.
#
# @param rds_resource [Aws::RDS::Resource] An SDK for Ruby Amazon RDS resource.
# @return instance_snapshots [Array, nil] All instance snapshots, or nil if
 error.
def list_instance_snapshots(rds_resource)
  instance_snapshots = []
  rds_resource.db_snapshots.each do |s|
    instance_snapshots.append({
                                "id": s.snapshot_id,
                                "status": s.status
                              })
  end
  instance_snapshots
rescue Aws::Errors::ServiceError => e
  puts "Couldn't list instance snapshots:\n #{e.message}"
end
```

- 有关 API 详细信息，请参阅《适用于 Ruby 的 Amazon SDK API 参考》中的 DescribeDBSnapshots。

有关 Amazon SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 Amazon SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DescribeOrderableDBInstanceOptions** 与 Amazon SDK 或 CLI 配合使用

以下代码示例演示如何使用 `DescribeOrderableDBInstanceOptions`。

操作示例是大型程序的代码摘录，必须在上下文中运行。在以下代码示例中，您可以查看此操作的上下文：

- 了解基础知识

.NET

适用于 .NET 的 Amazon SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
    /// <summary>
    /// Get a list of orderable DB instance options for a specific
    /// engine and engine version.
    /// </summary>
    /// <param name="engine">Name of the engine.</param>
    /// <param name="engineVersion">Version of the engine.</param>
    /// <returns>List of OrderableDBInstanceOptions.</returns>
    public async Task<List<OrderableDBInstanceOption>>
 DescribeOrderableDBInstanceOptions(string engine, string engineVersion)
    {
        // Use a paginator to get a list of DB instance options.
        var results = new List<OrderableDBInstanceOption>();
        var paginateInstanceOptions =
 _amazonRDS.Paginators.DescribeOrderableDBInstanceOptions(
            new DescribeOrderableDBInstanceOptionsRequest()
            {
                Engine = engine,
                EngineVersion = engineVersion,
            });
        // Get the entire list using the paginator.
        await foreach (var instanceOptions in
 paginateInstanceOptions.OrderableDBInstanceOptions)
```

```
        {
            results.Add(instanceOptions);
        }
        return results;
    }
```

- 有关 API 详细信息，请参阅《适用于 .NET 的 Amazon SDK API 参考》中的 DescribeOrderableDBInstanceOptions。

C++

SDK for C++

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::RDS::RDSClient client(clientConfig);


//! Routine which gets available 'micro' DB instance classes, displays the list
//! to the user, and returns the user selection.
/*!
 \sa chooseMicroDBInstanceClass()
 \param engineName: The DB engine name.
 \param engineVersion: The DB engine version.
 \param dbInstanceClass: String for DB instance class chosen by the user.
 \param client: 'RDSClient' instance.
 \return bool: Successful completion.
 */
bool AwsDoc::RDS::chooseMicroDBInstanceClass(const Aws::String &engine,
                                             const Aws::String &engineVersion,
                                             Aws::String &dbInstanceClass,
```

```
                                                const Aws::RDS::RDSClient &client) {
    std::vector<Aws::String> instanceClasses;
    Aws::String marker;
    do {
        Aws::RDS::Model::DescribeOrderableDBInstanceOptionsRequest request;
        request.SetEngine(engine);
        request.SetEngineVersion(engineVersion);
        if (!marker.empty()) {
            request.SetMarker(marker);
        }

        Aws::RDS::Model::DescribeOrderableDBInstanceOptionsOutcome outcome =
                client.DescribeOrderableDBInstanceOptions(request);

        if (outcome.IsSuccess()) {
            const Aws::Vector<Aws::RDS::Model::OrderableDBInstanceOption>
 &options =
                    outcome.GetResult().GetOrderableDBInstanceOptions();
            for (const Aws::RDS::Model::OrderableDBInstanceOption &option:
 options) {
                const Aws::String &instanceClass = option.GetDBInstanceClass();
                if (instanceClass.find("micro") != std::string::npos) {
                    if (std::find(instanceClasses.begin(), instanceClasses.end(),
                                  instanceClass) ==
                        instanceClasses.end()) {
                        instanceClasses.push_back(instanceClass);
                    }
                }
            }
            marker = outcome.GetResult().GetMarker();
        }
        else {
            std::cerr << "Error with RDS::DescribeOrderableDBInstanceOptions. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            return false;
        }
    } while (!marker.empty());

    std::cout << "The available micro DB instance classes for your database
 engine are:"
              << std::endl;
    for (int i = 0; i < instanceClasses.size(); ++i) {
        std::cout << "   " << i + 1 << ": " << instanceClasses[i] << std::endl;
```

```
    }

    int choice = askQuestionForIntRange(
            "Which micro DB instance class do you want to use? ",
            1, static_cast<int>(instanceClasses.size()));
    dbInstanceClass = instanceClasses[choice - 1];
    return true;
}
```

- 有关 API 详细信息，请参阅《适用于 C++ 的 Amazon SDK API 参考》中的 DescribeOrderableDBInstanceOptions。

CLI

Amazon CLI

描述可订购的数据库实例选项

以下 `describe-orderable-db-instance-options` 示例将检索有关运行 MySQL 数据库引擎的数据库实例的可订购选项的详细信息。

```
aws rds describe-orderable-db-instance-options \
    --engine mysql
```

输出：

```
{
    "OrderableDBInstanceOptions": [
        {
            "MinStorageSize": 5,
            "ReadReplicaCapable": true,
            "MaxStorageSize": 6144,
            "AvailabilityZones": [
                {
                    "Name": "us-east-1a"
                },
                {
                    "Name": "us-east-1b"
                },
                {
                    "Name": "us-east-1c"
```

```
                },
                {
                    "Name": "us-east-1d"
                }
            ],
            "SupportsIops": false,
            "AvailableProcessorFeatures": [],
            "MultiAZCapable": true,
            "DBInstanceClass": "db.m1.large",
            "Vpc": true,
            "StorageType": "gp2",
            "LicenseModel": "general-public-license",
            "EngineVersion": "5.5.46",
            "SupportsStorageEncryption": false,
            "SupportsEnhancedMonitoring": true,
            "Engine": "mysql",
            "SupportsIAMDatabaseAuthentication": false,
            "SupportsPerformanceInsights": false
        }
    ]
    ...some output truncated...
}
```

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 [DescribeOrderableDBInstanceOptions](https://docs.aws.amazon.com/goto/aws-cli/rds-2014-10-31/DescribeOrderableDBInstanceOptions)。

Go

适用于 Go V2 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import (
 "context"
 "errors"
 "log"
```

```
 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/rds"
 "github.com/aws/aws-sdk-go-v2/service/rds/types"
)

type DbInstances struct {
 RdsClient *rds.Client
}



// GetOrderableInstances uses a paginator to get DB instance options that can be
 used to create DB instances that are
// compatible with a set of specifications.
func (instances *DbInstances) GetOrderableInstances(ctx context.Context, engine
 string, engineVersion string) (
 []types.OrderableDBInstanceOption, error) {

 var output *rds.DescribeOrderableDBInstanceOptionsOutput
 var instanceOptions []types.OrderableDBInstanceOption
 var err error
 orderablePaginator :=
 rds.NewDescribeOrderableDBInstanceOptionsPaginator(instances.RdsClient,
  &rds.DescribeOrderableDBInstanceOptionsInput{
   Engine:        aws.String(engine),
   EngineVersion: aws.String(engineVersion),
  })
 for orderablePaginator.HasMorePages() {
  output, err = orderablePaginator.NextPage(ctx)
  if err != nil {
   log.Printf("Couldn't get orderable DB instance options: %v\n", err)
   break
  } else {
   instanceOptions = append(instanceOptions,
 output.OrderableDBInstanceOptions...)
  }
 }
 return instanceOptions, err
}
```

- 有关 API 详细信息，请参阅《适用于 Go 的 Amazon SDK API 参考》中的 DescribeOrderableDBInstanceOptions。

Python

适用于 Python 的 SDK（Boto3）

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
class InstanceWrapper:
    """Encapsulates Amazon RDS DB instance actions."""

    def __init__(self, rds_client):
        """
        :param rds_client: A Boto3 Amazon RDS client.
        """
        self.rds_client = rds_client

    @classmethod
    def from_client(cls):
        """
        Instantiates this class from a Boto3 client.
        """
        rds_client = boto3.client("rds")
        return cls(rds_client)


    def get_orderable_instances(self, db_engine, db_engine_version):
        """
        Gets DB instance options that can be used to create DB instances that are
        compatible with a set of specifications.

        :param db_engine: The database engine that must be supported by the DB
 instance.
        :param db_engine_version: The engine version that must be supported by
 the DB instance.
```

```
        :return: The list of DB instance options that can be used to create a
compatible DB instance.
        """
        try:
            inst_opts = []
            paginator = self.rds_client.get_paginator(
                "describe_orderable_db_instance_options"
            )
            for page in paginator.paginate(
                Engine=db_engine, EngineVersion=db_engine_version
            ):
                inst_opts += page["OrderableDBInstanceOptions"]
        except ClientError as err:
            logger.error(
                "Couldn't get orderable DB instances. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return inst_opts
```

- 有关 API 详细信息，请参阅《适用于 Python 的 Amazon SDK（Boto3）API 参考》中的 DescribeOrderableDBInstanceOptions。

有关 Amazon SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 Amazon SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **GenerateRDSAuthToken** 和 Amazon SDK 搭配使用

以下代码示例演示了如何使用 GenerateRDSAuthToken。

Java

适用于 Java 的 SDK 2.x

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

使用 RdsUtilities 类生成身份验证令牌。

```
public class GenerateRDSAuthToken {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <dbInstanceIdentifier> <masterUsername>

                Where:
                    dbInstanceIdentifier - The database instance identifier.\s
                    masterUsername - The master user name.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String dbInstanceIdentifier = args[0];
        String masterUsername = args[1];
        Region region = Region.US_WEST_2;
        RdsClient rdsClient = RdsClient.builder()
                .region(region)
                .build();

        String token = getAuthToken(rdsClient, dbInstanceIdentifier,
 masterUsername);
        System.out.println("The token response is " + token);
    }

    public static String getAuthToken(RdsClient rdsClient, String
 dbInstanceIdentifier, String masterUsername) {
```

```
        RdsUtilities utilities = rdsClient.utilities();
        try {
            GenerateAuthenticationTokenRequest tokenRequest =
 GenerateAuthenticationTokenRequest.builder()
                    .credentialsProvider(ProfileCredentialsProvider.create())
                    .username(masterUsername)
                    .port(3306)
                    .hostname(dbInstanceIdentifier)
                    .build();

            return utilities.generateAuthenticationToken(tokenRequest);

        } catch (RdsException e) {
            System.out.println(e.getLocalizedMessage());
            System.exit(1);
        }
        return "";
    }
}
```

- 有关 API 详细信息，请参阅《Amazon SDK for Java 2.x API 参考》中的 GenerateRDSAuthToken。

有关 Amazon SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 Amazon SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **ModifyDBInstance** 与 Amazon SDK 或 CLI 配合使用

以下代码示例演示如何使用 `ModifyDBInstance`。

CLI

Amazon CLI

示例 1：修改数据库实例

以下 `modify-db-instance` 示例将选项组和参数组与兼容的 Microsoft SQL Server 数据库实例相关联。`--apply-immediately` 参数使选项和参数组立即关联，而不是等到下一个维护时段。

```
aws rds modify-db-instance \
    --db-instance-identifier database-2 \
    --option-group-name test-se-2017 \
    --db-parameter-group-name test-sqlserver-se-2017 \
    --apply-immediately
```

输出：


```
{
    "DBInstance": {
        "DBInstanceIdentifier": "database-2",
        "DBInstanceClass": "db.r4.large",
        "Engine": "sqlserver-se",
        "DBInstanceStatus": "available",

        ...output omitted...

        "DBParameterGroups": [
            {
                "DBParameterGroupName": "test-sqlserver-se-2017",
                "ParameterApplyStatus": "applying"
            }
        ],
        "AvailabilityZone": "us-west-2d",

        ...output omitted...

        "MultiAZ": true,
        "EngineVersion": "14.00.3281.6.v1",
        "AutoMinorVersionUpgrade": false,
        "ReadReplicaDBInstanceIdentifiers": [],
        "LicenseModel": "license-included",
        "OptionGroupMemberships": [
            {
                "OptionGroupName": "test-se-2017",
                "Status": "pending-apply"
            }
        ],
        "CharacterSetName": "SQL_Latin1_General_CP1_CI_AS",
        "SecondaryAvailabilityZone": "us-west-2c",
        "PubliclyAccessible": true,
        "StorageType": "gp2",
```

```
        ...output omitted...

        "DeletionProtection": false,
        "AssociatedRoles": [],
        "MaxAllocatedStorage": 1000
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的修改 Amazon RDS 数据库实例。

示例 2：将 VPC 安全组与数据库实例关联

以下 `modify-db-instance` 示例关联特定 VPC 安全组并从数据库实例中移除数据库安全组：

```
aws rds modify-db-instance \
    --db-instance-identifier dbName \
    --vpc-security-group-ids sg-ID
```

输出：

```
{
"DBInstance": {
    "DBInstanceIdentifier": "dbName",
    "DBInstanceClass": "db.t3.micro",
    "Engine": "mysql",
    "DBInstanceStatus": "available",
    "MasterUsername": "admin",
    "Endpoint": {
        "Address": "dbName.abcdefghijk.us-west-2.rds.amazonaws.com",
        "Port": 3306,
        "HostedZoneId": "ABCDEFGHIJK1234"
    },
    "AllocatedStorage": 20,
    "InstanceCreateTime": "2024-02-15T00:37:58.793000+00:00",
    "PreferredBackupWindow": "11:57-12:27",
    "BackupRetentionPeriod": 7,
    "DBSecurityGroups": [],
    "VpcSecurityGroups": [
        {
            "VpcSecurityGroupId": "sg-ID",
            "Status": "active"
        }
    ],
```

```
    ... output omitted ...
    "MultiAZ": false,
    "EngineVersion": "8.0.35",
    "AutoMinorVersionUpgrade": true,
    "ReadReplicaDBInstanceIdentifiers": [],
    "LicenseModel": "general-public-license",

    ... output ommited ...
    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的使用安全组控制访问权限。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyDBInstance。

Java

适用于 Java 的 SDK 2.x

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.rds.RdsClient;
import software.amazon.awssdk.services.rds.model.ModifyDbInstanceRequest;
import software.amazon.awssdk.services.rds.model.ModifyDbInstanceResponse;
import software.amazon.awssdk.services.rds.model.RdsException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ModifyDBInstance {
    public static void main(String[] args) {
```

```
        final String usage = """

                Usage:
                    <dbInstanceIdentifier> <dbSnapshotIdentifier>\s
                Where:
                    dbInstanceIdentifier - The database instance identifier.\s
                    masterUserPassword - The updated password that corresponds to
 the master user name.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String dbInstanceIdentifier = args[0];
        String masterUserPassword = args[1];
        Region region = Region.US_WEST_2;
        RdsClient rdsClient = RdsClient.builder()
                .region(region)
                .build();

        updateIntance(rdsClient, dbInstanceIdentifier, masterUserPassword);
        rdsClient.close();
    }

    public static void updateIntance(RdsClient rdsClient, String
 dbInstanceIdentifier, String masterUserPassword) {
        try {
            // For a demo - modify the DB instance by modifying the master
 password.
            ModifyDbInstanceRequest modifyDbInstanceRequest =
 ModifyDbInstanceRequest.builder()
                    .dbInstanceIdentifier(dbInstanceIdentifier)
                    .publiclyAccessible(true)
                    .masterUserPassword(masterUserPassword)
                    .build();

            ModifyDbInstanceResponse instanceResponse =
 rdsClient.modifyDBInstance(modifyDbInstanceRequest);
            System.out.print("The ARN of the modified database is: " +
 instanceResponse.dbInstance().dbInstanceArn());

        } catch (RdsException e) {
```

```
                System.out.println(e.getLocalizedMessage());
                System.exit(1);
            }
        }
    }
```

- 有关 API 详细信息，请参阅《Amazon SDK for Java 2.x API 参考》中的 ModifyDBInstance。

Kotlin

适用于 Kotlin 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
suspend fun updateIntance(
    dbInstanceIdentifierVal: String?,
    masterUserPasswordVal: String?,
) {
    val request =
        ModifyDbInstanceRequest {
            dbInstanceIdentifier = dbInstanceIdentifierVal
            publiclyAccessible = true
            masterUserPassword = masterUserPasswordVal
        }

    RdsClient { region = "us-west-2" }.use { rdsClient ->
        val instanceResponse = rdsClient.modifyDbInstance(request)
        println("The ARN of the modified database is
 ${instanceResponse.dbInstance?.dbInstanceArn}")
    }
}
```

- 有关 API 详细信息，请参阅《Amazon SDK for Kotlin API 参考》中的 ModifyDBInstance。

有关 Amazon SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 Amazon SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 ModifyDBParameterGroup 与 Amazon SDK 或 CLI 配合使用

以下代码示例演示如何使用 ModifyDBParameterGroup。

操作示例是大型程序的代码摘录，必须在上下文中运行。在以下代码示例中，您可以查看此操作的上下文：

- 了解基础知识

.NET

适用于 .NET 的 Amazon SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
    /// <summary>
    /// Update a DB parameter group. Use the action
 DescribeDBParameterGroupsAsync
    /// to determine when the DB parameter group is ready to use.
    /// </summary>
    /// <param name="name">Name of the DB parameter group.</param>
    /// <param name="parameters">List of parameters. Maximum of 20 per request.</
param>
    /// <returns>The updated DB parameter group name.</returns>
    public async Task<string> ModifyDBParameterGroup(
        string name, List<Parameter> parameters)
    {
        var response = await _amazonRDS.ModifyDBParameterGroupAsync(
            new ModifyDBParameterGroupRequest()
            {
                DBParameterGroupName = name,
                Parameters = parameters,
            });
```

```
        return response.DBParameterGroupName;
    }
```

- 有关 API 详细信息，请参阅《适用于 .NET 的 Amazon SDK API 参考》中的 ModifyDBParameterGroup。

C++

SDK for C++

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::RDS::RDSClient client(clientConfig);

        Aws::RDS::Model::ModifyDBParameterGroupRequest request;
        request.SetDBParameterGroupName(PARAMETER_GROUP_NAME);
        request.SetParameters(updateParameters);

        Aws::RDS::Model::ModifyDBParameterGroupOutcome outcome =
                client.ModifyDBParameterGroup(request);

        if (outcome.IsSuccess()) {
            std::cout << "The DB parameter group was successfully modified."
                      << std::endl;
        }
        else {
            std::cerr << "Error with RDS::ModifyDBParameterGroup. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
        }
```

- 有关 API 详细信息，请参阅《适用于 C++ 的 Amazon SDK API 参考》中的 ModifyDBParameterGroup。

CLI

Amazon CLI

修改数据库参数组

以下 modify-db-parameter-group 示例将更改数据库参数组中 clr enabled 参数的值。--apply-immediately 参数使数据库参数组得到立即修改，而不是等到下一个维护时段。

```
aws rds modify-db-parameter-group \
    --db-parameter-group-name test-sqlserver-se-2017 \
    --parameters "ParameterName='clr
 enabled',ParameterValue=1,ApplyMethod=immediate"
```

输出：

```
{
    "DBParameterGroupName": "test-sqlserver-se-2017"
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的修改数据库参数组中的参数。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 ModifyDBParameterGroup。

Go

适用于 Go V2 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import (
 "context"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/rds"
 "github.com/aws/aws-sdk-go-v2/service/rds/types"
)

type DbInstances struct {
 RdsClient *rds.Client
}



// UpdateParameters updates parameters in a named DB parameter group.
func (instances *DbInstances) UpdateParameters(ctx context.Context,
 parameterGroupName string, params []types.Parameter) error {
 _, err := instances.RdsClient.ModifyDBParameterGroup(ctx,
  &rds.ModifyDBParameterGroupInput{
   DBParameterGroupName: aws.String(parameterGroupName),
   Parameters:           params,
  })
 if err != nil {
  log.Printf("Couldn't update parameters in %v: %v\n", parameterGroupName, err)
  return err
 } else {
  return nil
 }
}
```

- 有关 API 详细信息，请参阅《适用于 Go 的 Amazon SDK API 参考》中的 ModifyDBParameterGroup。

Java

适用于 Java 的 SDK 2.x

> **Note**
>
> 查看 GitHub，了解更多信息。在 [Amazon 代码示例存储库](https://github.com/awsdocs/aws-doc-sdk-examples)中查找完整示例，了解如何进行设置和运行。

```
    // Modify auto_increment_offset and auto_increment_increment parameters.
    public static void modifyDBParas(RdsClient rdsClient, String dbGroupName) {
        try {
            Parameter parameter1 = Parameter.builder()
                    .parameterName("auto_increment_offset")
                    .applyMethod("immediate")
                    .parameterValue("5")
                    .build();

            List<Parameter> paraList = new ArrayList<>();
            paraList.add(parameter1);
            ModifyDbParameterGroupRequest groupRequest =
 ModifyDbParameterGroupRequest.builder()
                    .dbParameterGroupName(dbGroupName)
                    .parameters(paraList)
                    .build();

            ModifyDbParameterGroupResponse response =
 rdsClient.modifyDBParameterGroup(groupRequest);
            System.out.println("The parameter group " +
 response.dbParameterGroupName() + " was successfully modified");

        } catch (RdsException e) {
            System.out.println(e.getLocalizedMessage());
            System.exit(1);
        }
    }
```

- 有关 API 详细信息，请参阅《Amazon SDK for Java 2.x API 参考》中的 ModifyDBParameterGroup。

Python

适用于 Python 的 SDK（Boto3）

> **Note**
>
> 查看 GitHub，了解更多信息。在 [Amazon 代码示例存储库]中查找完整示例，了解如何进行设置和运行。

```
class InstanceWrapper:
    """Encapsulates Amazon RDS DB instance actions."""

    def __init__(self, rds_client):
        """
        :param rds_client: A Boto3 Amazon RDS client.
        """
        self.rds_client = rds_client

    @classmethod
    def from_client(cls):
        """
        Instantiates this class from a Boto3 client.
        """
        rds_client = boto3.client("rds")
        return cls(rds_client)


    def update_parameters(self, parameter_group_name, update_parameters):
        """
        Updates parameters in a custom DB parameter group.

        :param parameter_group_name: The name of the parameter group to update.
        :param update_parameters: The parameters to update in the group.
        :return: Data about the modified parameter group.
        """
        try:
            response = self.rds_client.modify_db_parameter_group(
                DBParameterGroupName=parameter_group_name,
 Parameters=update_parameters
            )
        except ClientError as err:
```

```
            logger.error(
                "Couldn't update parameters in %s. Here's why: %s: %s",
                parameter_group_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response
```

- 有关 API 详细信息，请参阅《适用于 Python 的 Amazon SDK（Boto3）API 参考》中的 ModifyDBParameterGroup。

有关 Amazon SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 Amazon SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **RebootDBInstance** 与 Amazon SDK 或 CLI 配合使用

以下代码示例演示如何使用 `RebootDBInstance`。

CLI

Amazon CLI

重启数据库实例

以下 `reboot-db-instance` 示例开始重启指定的数据库实例。

```
aws rds reboot-db-instance \
    --db-instance-identifier test-mysql-instance
```

输出：

```
{
    "DBInstance": {
        "DBInstanceIdentifier": "test-mysql-instance",
        "DBInstanceClass": "db.t3.micro",
        "Engine": "mysql",
        "DBInstanceStatus": "rebooting",
        "MasterUsername": "admin",
```

```
        "Endpoint": {
            "Address": "test-mysql-instance.############.us-
west-2.rds.amazonaws.com",
            "Port": 3306,
            "HostedZoneId": "Z1PVIF0EXAMPLE"
        },

    ... output omitted...

    }
}
```

有关更多信息，请参阅《Amazon RDS 用户指南》中的重启数据库实例。

- 有关 API 详细信息，请参阅《Amazon CLI 命令参考》中的 RebootDBInstance。

Java

适用于 Java 的 SDK 2.x

> **Note**
>
> 查看 GitHub，了解更多信息。在 Amazon 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.rds.RdsClient;
import software.amazon.awssdk.services.rds.model.RebootDbInstanceRequest;
import software.amazon.awssdk.services.rds.model.RebootDbInstanceResponse;
import software.amazon.awssdk.services.rds.model.RdsException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class RebootDBInstance {
```

```
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <dbInstanceIdentifier>\s

                Where:
                    dbInstanceIdentifier - The database instance identifier\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String dbInstanceIdentifier = args[0];
        Region region = Region.US_WEST_2;
        RdsClient rdsClient = RdsClient.builder()
                .region(region)
                .build();

        rebootInstance(rdsClient, dbInstanceIdentifier);
        rdsClient.close();
    }

    public static void rebootInstance(RdsClient rdsClient, String
 dbInstanceIdentifier) {
        try {
            RebootDbInstanceRequest rebootDbInstanceRequest =
 RebootDbInstanceRequest.builder()
                    .dbInstanceIdentifier(dbInstanceIdentifier)
                    .build();

            RebootDbInstanceResponse instanceResponse =
 rdsClient.rebootDBInstance(rebootDbInstanceRequest);
            System.out.print("The database " +
 instanceResponse.dbInstance().dbInstanceArn() + " was rebooted");

        } catch (RdsException e) {
            System.out.println(e.getLocalizedMessage());
            System.exit(1);
        }
    }
}
```

- 有关 API 详细信息，请参阅《Amazon SDK for Java 2.x API 参考》中的 RebootDBInstance。

有关 Amazon SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 Amazon SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

适用于使用 Amazon SDK 的 Amazon RDS 的场景

以下代码示例向您展示如何通过 Amazon SDK 实施 Amazon RDS 中的常见场景。这些场景向您展示了如何通过调用 Amazon RDS 中的多个函数或与其他 Amazon Web Services 服务结合来完成特定任务。每个场景都包含完整源代码的链接，您可以在其中找到有关如何设置和运行代码的说明。

场景以中等水平的经验为目标，可帮助您结合具体环境了解服务操作。

示例

- 创建 Aurora Serverless 工作项跟踪器

创建 Aurora Serverless 工作项跟踪器

以下代码示例演示了如何创建用于跟踪 Amazon Aurora Serverless 数据库中的工作项，以及使用 Amazon Simple Email Service (Amazon SES) 发送报告的 Web 应用程序。

.NET

适用于 .NET 的 Amazon SDK

展示如何使用 适用于 .NET 的 Amazon SDK 通过 Amazon Simple Email Service (Amazon SES) 创建 Web 应用程序来跟踪 Amazon Aurora 数据库中的工作项和电子邮件报告。此示例使用由 React.js 构建的前端与 RESTful .NET 后端进行交互。

- 将 React Web 应用程序与 Amazon 服务集成。
- 列出、添加、更新和删除 Aurora 表中的项目。
- 使用 Amazon SES 以电子邮件形式发送已筛选工作项的报告。
- 使用随附的 Amazon CloudFormation 脚本部署与管理示例资源。

有关完整的源代码以及如何设置和运行的说明，请参阅 GitHub 上的完整示例。

本示例中使用的服务

- Aurora
- Amazon RDS
- Amazon RDS 数据服务
- Amazon SES

C++

SDK for C++

演示了如何创建用于跟踪和报告存储在 Amazon Aurora Serverless 数据库中的工作项的 Web 应用程序。

有关完整源代码以及如何设置可查询 Amazon Aurora Serverless 数据以及供 React 应用程序使用的 C++ REST API 的说明，请参阅 GitHub 上的完整示例。

本示例中使用的服务

- Aurora
- Amazon RDS
- Amazon RDS 数据服务
- Amazon SES

Java

适用于 Java 的 SDK 2.x

展示如何创建 Web 应用程序来跟踪与报告存储与 Amazon RDS 数据库的工作项。

有关完整源代码以及如何设置查询 Amazon Aurora 无服务器数据以及供 React 应用程序使用的 Spring REST API 的说明，请参阅 GitHub 上的完整示例。

有关完整的源代码以及如何设置和运行使用 JDBC API 的示例的说明，请参阅 GitHub 上的完整示例。

本示例中使用的服务

- Aurora
- Amazon RDS

- Amazon RDS 数据服务
- Amazon SES

JavaScript

SDK for JavaScript (v3)

展示了如何使用 适用于 JavaScript 的 Amazon SDK (v3) 创建可跟踪 Amazon Aurora 数据库中的工作项，以及使用 Amazon Simple Email Service (Amazon SES) 通过电子邮件发送报告的 Web 应用程序。此示例使用由 React.js 构建的前端与 Express Node.js 后端进行交互。

- 将 React.js Web 应用程序与 Amazon Web Services 服务 集成。
- 列出、添加以及更新 Aurora 表中的项目。
- 使用 Amazon SES 以电子邮件形式发送已筛选工作项的报告。
- 使用随附的 Amazon CloudFormation 脚本部署与管理示例资源。

有关完整的源代码以及如何设置和运行的说明，请参阅 GitHub 上的完整示例。

本示例中使用的服务

- Aurora
- Amazon RDS
- Amazon RDS 数据服务
- Amazon SES

Kotlin

适用于 Kotlin 的 SDK

展示如何创建 Web 应用程序来跟踪与报告存储与 Amazon RDS 数据库的工作项。

有关完整源代码以及如何设置查询 Amazon Aurora 无服务器数据以及供 React 应用程序使用的 Spring REST API 的说明，请参阅 GitHub 上的完整示例。

本示例中使用的服务

- Aurora
- Amazon RDS
- Amazon RDS 数据服务
- Amazon SES

PHP

适用于 PHP 的 SDK

展示了如何使用 适用于 PHP 的 Amazon SDK 来创建用于跟踪 Amazon RDS 数据库中的工作项，并通过 Amazon Simple Email Service (Amazon SES) 以电子邮件发送报告的 Web 应用程序。此示例使用由 React.js 构建的前端与 RESTful PHP 后端进行交互。

- 将 React.js Web 应用程序与 Amazon 服务集成。
- 列出、添加、更新和删除 Amazon RDS 表中的项目。
- 使用 Amazon SES 以电子邮件发送已筛选工作项的报告。
- 使用随附的 Amazon CloudFormation 脚本部署与管理示例资源。

有关完整的源代码以及如何设置和运行的说明，请参阅 GitHub 上的完整示例。

本示例中使用的服务

- Aurora
- Amazon RDS
- Amazon RDS 数据服务
- Amazon SES

Python

适用于 Python 的 SDK（Boto3）

演示了如何使用 适用于 Python (Boto3) 的 Amazon SDK 创建可跟踪 Amazon Aurora Serverless 数据库中的工作项，以及使用 Amazon Simple Email Service (Amazon SES) 通过电子邮件发送报告的 REST 服务。此示例使用 Flask Web 框架处理 HTTP 路由，并与 React 网页集成以呈现功能齐全的 Web 应用程序。

- 构建与 Amazon Web Services 服务 集成的 Flask REST 服务。
- 读取、写入和更新存储在 Aurora Serverless 数据库中的工作项。
- 创建包含数据库凭证的 Amazon Secrets Manager 密钥，并使用它来验证对数据库的调用。
- 使用 Amazon SES 发送工作项的电子邮件报告。

有关完整的源代码以及如何设置和运行的说明，请参阅 GitHub 上的完整示例。

本示例中使用的服务

- Aurora

- Amazon RDS
- Amazon RDS 数据服务
- Amazon SES

有关 Amazon SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 Amazon SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

Amazon RDS 的无服务器示例

以下代码示例显示了如何将 Amazon RDS 与 Amazon SDK 一起使用。

示例

- 使用 Lambda 函数连接到 Amazon RDS 数据库

使用 Lambda 函数连接到 Amazon RDS 数据库

以下代码示例显示如何实现连接到 RDS 数据库的 Lambda 函数。该函数发出一个简单的数据库请求并返回结果。

.NET

适用于 .NET 的 Amazon SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在无服务器示例存储库中查找完整示例，并了解如何进行设置和运行。

在 Lambda 函数中使用 .NET 连接到 Amazon RDS 数据库。

```
using System.Data;
using System.Text.Json;
using Amazon.Lambda.APIGatewayEvents;
using Amazon.Lambda.Core;
using MySql.Data.MySqlClient;

// Assembly attribute to enable the Lambda function's JSON input to be converted
 into a .NET class.
```

```
[assembly:
 LambdaSerializer(typeof(Amazon.Lambda.Serialization.SystemTextJson.DefaultLambdaJsonSeri

namespace aws_rds;

public class InputModel
{
    public string key1 { get; set; }
    public string key2 { get; set; }
}

public class Function
{
    /// <summary>
    // Handles the Lambda function execution for connecting to RDS using IAM
 authentication.
    /// </summary>
    /// <param name="input">The input event data passed to the Lambda function</
param>
    /// <param name="context">The Lambda execution context that provides runtime
 information</param>
    /// <returns>A response object containing the execution result</returns>

    public async Task<APIGatewayProxyResponse>
 FunctionHandler(APIGatewayProxyRequest request, ILambdaContext context)
    {
        // Sample Input: {"body": "{\"key1\":\"20\", \"key2\":\"25\"}"}
        var input = JsonSerializer.Deserialize<InputModel>(request.Body);

        /// Obtain authentication token
        var authToken = RDSAuthTokenGenerator.GenerateAuthToken(
            Environment.GetEnvironmentVariable("RDS_ENDPOINT"),
            Convert.ToInt32(Environment.GetEnvironmentVariable("RDS_PORT")),
            Environment.GetEnvironmentVariable("RDS_USERNAME")
        );

        /// Build the Connection String with the Token
        string connectionString =
 $"Server={Environment.GetEnvironmentVariable("RDS_ENDPOINT")};" +

 $"Port={Environment.GetEnvironmentVariable("RDS_PORT")};" +

 $"Uid={Environment.GetEnvironmentVariable("RDS_USERNAME")};" +
                                  $"Pwd={authToken};";
```

```
        try
        {
            await using var connection = new MySqlConnection(connectionString);
            await connection.OpenAsync();

            const string sql = "SELECT @param1 + @param2 AS Sum";

            await using var command = new MySqlCommand(sql, connection);
            command.Parameters.AddWithValue("@param1", int.Parse(input.key1 ??
 "0"));
            command.Parameters.AddWithValue("@param2", int.Parse(input.key2 ??
 "0"));

            await using var reader = await command.ExecuteReaderAsync();
            if (await reader.ReadAsync())
            {
                int result = reader.GetInt32("Sum");

                //Sample Response: {"statusCode":200,"body":"{\"message\":\"The
 sum is: 45\"}","isBase64Encoded":false}
                return new APIGatewayProxyResponse
                {
                    StatusCode = 200,
                    Body = JsonSerializer.Serialize(new { message = $"The sum is:
 {result}" })
                };
            }

        }
        catch (Exception ex)
        {
            Console.WriteLine($"Error: {ex.Message}");
        }

        return new APIGatewayProxyResponse
        {
            StatusCode = 500,
            Body = JsonSerializer.Serialize(new { error = "Internal server
 error" })
        };
    }
}
```

Go

适用于 Go V2 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在无服务器示例存储库中查找完整示例，并了解如何进行设置和运行。

在 Lambda 函数中使用 Go 连接到 Amazon RDS 数据库。

```
/*
Golang v2 code here.
*/

package main

import (
 "context"
 "database/sql"
 "encoding/json"
 "fmt"
 "os"

 "github.com/aws/aws-lambda-go/lambda"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/feature/rds/auth"
 _ "github.com/go-sql-driver/mysql"
)

type MyEvent struct {
 Name string `json:"name"`
}

func HandleRequest(event *MyEvent) (map[string]interface{}, error) {

 var dbName string = os.Getenv("DatabaseName")
 var dbUser string = os.Getenv("DatabaseUser")
 var dbHost string = os.Getenv("DBHost") // Add hostname without https
```

```
var dbPort int = os.Getenv("Port")      // Add port number
var dbEndpoint string = fmt.Sprintf("%s:%d", dbHost, dbPort)
var region string = os.Getenv("AWS_REGION")

cfg, err := config.LoadDefaultConfig(context.TODO())
if err != nil {
 panic("configuration error: " + err.Error())
}

authenticationToken, err := auth.BuildAuthToken(
 context.TODO(), dbEndpoint, region, dbUser, cfg.Credentials)
if err != nil {
 panic("failed to create authentication token: " + err.Error())
}

dsn := fmt.Sprintf("%s:%s@tcp(%s)/%s?tls=true&allowCleartextPasswords=true",
 dbUser, authenticationToken, dbEndpoint, dbName,
)

db, err := sql.Open("mysql", dsn)
if err != nil {
 panic(err)
}

defer db.Close()

var sum int
err = db.QueryRow("SELECT ?+? AS sum", 3, 2).Scan(&sum)
if err != nil {
 panic(err)
}
s := fmt.Sprint(sum)
message := fmt.Sprintf("The selected sum is: %s", s)

messageBytes, err := json.Marshal(message)
if err != nil {
 return nil, err
}

messageString := string(messageBytes)
return map[string]interface{}{
 "statusCode": 200,
 "headers":    map[string]string{"Content-Type": "application/json"},
 "body":       messageString,
```

```
    }, nil
}

func main() {
    lambda.Start(HandleRequest)
}
```

Java

适用于 Java 的 SDK 2.x

> **Note**
>
> 查看 GitHub，了解更多信息。在无服务器示例存储库中查找完整示例，并了解如何进行设置和运行。

在 Lambda 函数中使用 Java 连接到 Amazon RDS 数据库。

```
import com.amazonaws.services.lambda.runtime.Context;
import com.amazonaws.services.lambda.runtime.RequestHandler;
import com.amazonaws.services.lambda.runtime.events.APIGatewayProxyRequestEvent;
import com.amazonaws.services.lambda.runtime.events.APIGatewayProxyResponseEvent;
import software.amazon.awssdk.auth.credentials.DefaultCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.rdsdata.RdsDataClient;
import software.amazon.awssdk.services.rdsdata.model.ExecuteStatementRequest;
import software.amazon.awssdk.services.rdsdata.model.ExecuteStatementResponse;
import software.amazon.awssdk.services.rdsdata.model.Field;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;

public class RdsLambdaHandler implements
 RequestHandler<APIGatewayProxyRequestEvent, APIGatewayProxyResponseEvent> {

    @Override
    public APIGatewayProxyResponseEvent handleRequest(APIGatewayProxyRequestEvent
 event, Context context) {
```

```
        APIGatewayProxyResponseEvent response = new
 APIGatewayProxyResponseEvent();

        try {
            // Obtain auth token
            String token = createAuthToken();

            // Define connection configuration
            String connectionString = String.format("jdbc:mysql://%s:%s/%s?
useSSL=true&requireSSL=true",
                    System.getenv("ProxyHostName"),
                    System.getenv("Port"),
                    System.getenv("DBName"));

            // Establish a connection to the database
            try (Connection connection =
 DriverManager.getConnection(connectionString, System.getenv("DBUserName"),
 token);
                 PreparedStatement statement =
 connection.prepareStatement("SELECT ? + ? AS sum")) {

                statement.setInt(1, 3);
                statement.setInt(2, 2);

                try (ResultSet resultSet = statement.executeQuery()) {
                    if (resultSet.next()) {
                        int sum = resultSet.getInt("sum");
                        response.setStatusCode(200);
                        response.setBody("The selected sum is: " + sum);
                    }
                }
            }

        } catch (Exception e) {
            response.setStatusCode(500);
            response.setBody("Error: " + e.getMessage());
        }

        return response;
    }

    private String createAuthToken() {
        // Create RDS Data Service client
        RdsDataClient rdsDataClient = RdsDataClient.builder()
```

```
                .region(Region.of(System.getenv("AWS_REGION")))
                .credentialsProvider(DefaultCredentialsProvider.create())
                .build();

        // Define authentication request
        ExecuteStatementRequest request = ExecuteStatementRequest.builder()
                .resourceArn(System.getenv("ProxyHostName"))
                .secretArn(System.getenv("DBUserName"))
                .database(System.getenv("DBName"))
                .sql("SELECT 'RDS IAM Authentication'")
                .build();

        // Execute request and obtain authentication token
        ExecuteStatementResponse response =
 rdsDataClient.executeStatement(request);
        Field tokenField = response.records().get(0).get(0);

        return tokenField.stringValue();
    }
}
```

JavaScript

适用于 JavaScript 的 SDK（v3）

> **Note**
>
> 查看 GitHub，了解更多信息。在无服务器示例存储库中查找完整示例，并了解如何进行设置和运行。

在 Lambda 函数中使用 JavaScript 连接到 Amazon RDS 数据库。

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
/*
Node.js code here.
*/
// ES6+ example
import { Signer } from "@aws-sdk/rds-signer";
import mysql from 'mysql2/promise';
```

```
async function createAuthToken() {
  // Define connection authentication parameters
  const dbinfo = {

    hostname: process.env.ProxyHostName,
    port: process.env.Port,
    username: process.env.DBUserName,
    region: process.env.AWS_REGION,

  }

  // Create RDS Signer object
  const signer = new Signer(dbinfo);

  // Request authorization token from RDS, specifying the username
  const token = await signer.getAuthToken();
  return token;
}

async function dbOps() {

  // Obtain auth token
  const token = await createAuthToken();
  // Define connection configuration
  let connectionConfig = {
    host: process.env.ProxyHostName,
    user: process.env.DBUserName,
    password: token,
    database: process.env.DBName,
    ssl: 'Amazon RDS'
  }
  // Create the connection to the DB
  const conn = await mysql.createConnection(connectionConfig);
  // Obtain the result of the query
  const [res,] = await conn.execute('select ?+? as sum', [3, 2]);
  return res;

}

export const handler = async (event) => {
  // Execute database flow
  const result = await dbOps();
  // Return result
```

```
  return {
    statusCode: 200,
    body: JSON.stringify("The selected sum is: " + result[0].sum)
  }
};
```

在 Lambda 函数中使用 TypeScript 连接到 Amazon RDS 数据库。

```
import { Signer } from "@aws-sdk/rds-signer";
import mysql from 'mysql2/promise';

// RDS settings
// Using '!' (non-null assertion operator) to tell the TypeScript compiler that
 the DB settings are not null or undefined,
const proxy_host_name = process.env.PROXY_HOST_NAME!
const port = parseInt(process.env.PORT!)
const db_name = process.env.DB_NAME!
const db_user_name = process.env.DB_USER_NAME!
const aws_region = process.env.AWS_REGION!


async function createAuthToken(): Promise<string> {

    // Create RDS Signer object
    const signer = new Signer({
        hostname: proxy_host_name,
        port: port,
        region: aws_region,
        username: db_user_name
    });

    // Request authorization token from RDS, specifying the username
    const token = await signer.getAuthToken();
    return token;
}

async function dbOps(): Promise<mysql.QueryResult | undefined> {
    try {
        // Obtain auth token
        const token = await createAuthToken();
        const conn = await mysql.createConnection({
```

```
            host: proxy_host_name,
            user: db_user_name,
            password: token,
            database: db_name,
            ssl: 'Amazon RDS' // Ensure you have the CA bundle for SSL connection
        });
        const [rows, fields] = await conn.execute('SELECT ? + ? AS sum', [3, 2]);
        console.log('result:', rows);
        return rows;
    }
    catch (err) {
        console.log(err);
    }
}

export const lambdaHandler = async (event: any): Promise<{ statusCode: number;
 body: string }> => {
    // Execute database flow
    const result = await dbOps();

    // Return error is result is undefined
    if (result == undefined)
        return {
            statusCode: 500,
            body: JSON.stringify(`Error with connection to DB host`)
        }

    // Return result
    return {
        statusCode: 200,
        body: JSON.stringify(`The selected sum is: ${result[0].sum}`)
    };
};
```

PHP

适用于 PHP 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在无服务器示例存储库中查找完整示例，并了解如何进行设置和运行。

在 Lambda 函数中使用 PHP 连接到 Amazon RDS 数据库。

```
<?php
# Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
# SPDX-License-Identifier: Apache-2.0

# using bref/bref and bref/logger for simplicity

use Bref\Context\Context;
use Bref\Event\Handler as StdHandler;
use Bref\Logger\StderrLogger;
use Aws\Rds\AuthTokenGenerator;
use Aws\Credentials\CredentialProvider;

require __DIR__ . '/vendor/autoload.php';

class Handler implements StdHandler
{
    private StderrLogger $logger;
    public function __construct(StderrLogger $logger)
    {
        $this->logger = $logger;
    }


    private function getAuthToken(): string {
        // Define connection authentication parameters
        $dbConnection = [
            'hostname' => getenv('DB_HOSTNAME'),
            'port' => getenv('DB_PORT'),
            'username' => getenv('DB_USERNAME'),
            'region' => getenv('AWS_REGION'),
        ];
```

```
        // Create RDS AuthTokenGenerator object
        $generator = new
 AuthTokenGenerator(CredentialProvider::defaultProvider());

        // Request authorization token from RDS, specifying the username
        return $generator->createToken(
            $dbConnection['hostname'] . ':' . $dbConnection['port'],
            $dbConnection['region'],
            $dbConnection['username']
        );
    }

    private function getQueryResults() {
        // Obtain auth token
        $token = $this->getAuthToken();

        // Define connection configuration
        $connectionConfig = [
            'host' => getenv('DB_HOSTNAME'),
            'user' => getenv('DB_USERNAME'),
            'password' => $token,
            'database' => getenv('DB_NAME'),
        ];

        // Create the connection to the DB
        $conn = new PDO(

 "mysql:host={$connectionConfig['host']};dbname={$connectionConfig['database']}",
            $connectionConfig['user'],
            $connectionConfig['password'],
            [
                PDO::MYSQL_ATTR_SSL_CA => '/path/to/rds-ca-2019-root.pem',
                PDO::MYSQL_ATTR_SSL_VERIFY_SERVER_CERT => true,
            ]
        );

        // Obtain the result of the query
        $stmt = $conn->prepare('SELECT ?+? AS sum');
        $stmt->execute([3, 2]);

        return $stmt->fetch(PDO::FETCH_ASSOC);
    }
```

```
    /**
     * @param mixed $event
     * @param Context $context
     * @return array
     */
    public function handle(mixed $event, Context $context): array
    {
        $this->logger->info("Processing query");

        // Execute database flow
        $result = $this->getQueryResults();

        return [
            'sum' => $result['sum']
        ];
    }
}

$logger = new StderrLogger();
return new Handler($logger);
```

Python

适用于 Python 的 SDK（Boto3）

> **Note**
>
> 查看 GitHub，了解更多信息。在无服务器示例存储库中查找完整示例，并了解如何进行设置和运行。

在 Lambda 函数中使用 Python 连接到 Amazon RDS 数据库。

```
import json
import os
import boto3
import pymysql

# RDS settings
proxy_host_name = os.environ['PROXY_HOST_NAME']
port = int(os.environ['PORT'])
db_name = os.environ['DB_NAME']
```

```
db_user_name = os.environ['DB_USER_NAME']
aws_region = os.environ['AWS_REGION']


# Fetch RDS Auth Token
def get_auth_token():
    client = boto3.client('rds')
    token = client.generate_db_auth_token(
        DBHostname=proxy_host_name,
        Port=port
        DBUsername=db_user_name
        Region=aws_region
    )
    return token

def lambda_handler(event, context):
    token = get_auth_token()
    try:
        connection = pymysql.connect(
            host=proxy_host_name,
            user=db_user_name,
            password=token,
            db=db_name,
            port=port,
            ssl={'ca': 'Amazon RDS'}  # Ensure you have the CA bundle for SSL
 connection
        )

        with connection.cursor() as cursor:
            cursor.execute('SELECT %s + %s AS sum', (3, 2))
            result = cursor.fetchone()

        return result

    except Exception as e:
        return (f"Error: {str(e)}")  # Return an error message if an exception
 occurs
```

Ruby

适用于 Ruby 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在无服务器示例存储库中查找完整示例，并了解如何进行设置和运行。

在 Lambda 函数中使用 Ruby 连接到 Amazon RDS 数据库。

```
# Ruby code here.

require 'aws-sdk-rds'
require 'json'
require 'mysql2'

def lambda_handler(event:, context:)
  endpoint = ENV['DBEndpoint'] # Add the endpoint without https"
  port = ENV['Port']           # 3306
  user = ENV['DBUser']
  region = ENV['DBRegion']     # 'us-east-1'
  db_name = ENV['DBName']

  credentials = Aws::Credentials.new(
    ENV['AWS_ACCESS_KEY_ID'],
    ENV['AWS_SECRET_ACCESS_KEY'],
    ENV['AWS_SESSION_TOKEN']
  )
  rds_client = Aws::RDS::AuthTokenGenerator.new(
    region: region,
    credentials: credentials
  )

  token = rds_client.auth_token(
    endpoint: endpoint+ ':' + port,
    user_name: user,
    region: region
  )

  begin
    conn = Mysql2::Client.new(
```

```
        host: endpoint,
        username: user,
        password: token,
        port: port,
        database: db_name,
        sslca: '/var/task/global-bundle.pem',
        sslverify: true,
        enable_cleartext_plugin: true
      )
      a = 3
      b = 2
      result = conn.query("SELECT #{a} + #{b} AS sum").first['sum']
      puts result
      conn.close
      {
        statusCode: 200,
        body: result.to_json
      }
    rescue => e
      puts "Database connection failed due to #{e}"
    end
end
```

Rust

适用于 Rust 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在无服务器示例存储库中查找完整示例，并了解如何进行设置和运行。

在 Lambda 函数中使用 Rust 连接到 Amazon RDS 数据库。

```
use aws_config::BehaviorVersion;
use aws_credential_types::provider::ProvideCredentials;
use aws_sigv4::{
    http_request::{sign, SignableBody, SignableRequest, SigningSettings},
    sign::v4,
};
use lambda_runtime::{run, service_fn, Error, LambdaEvent};
```

```
use serde_json::{json, Value};
use sqlx::postgres::PgConnectOptions;
use std::env;
use std::time::{Duration, SystemTime};

const RDS_CERTS: &[u8] = include_bytes!("global-bundle.pem");

async fn generate_rds_iam_token(
    db_hostname: &str,
    port: u16,
    db_username: &str,
) -> Result<String, Error> {
    let config = aws_config::load_defaults(BehaviorVersion::v2024_03_28()).await;

    let credentials = config
        .credentials_provider()
        .expect("no credentials provider found")
        .provide_credentials()
        .await
        .expect("unable to load credentials");
    let identity = credentials.into();
    let region = config.region().unwrap().to_string();

    let mut signing_settings = SigningSettings::default();
    signing_settings.expires_in = Some(Duration::from_secs(900));
    signing_settings.signature_location =
 aws_sigv4::http_request::SignatureLocation::QueryParams;

    let signing_params = v4::SigningParams::builder()
        .identity(&identity)
        .region(&region)
        .name("rds-db")
        .time(SystemTime::now())
        .settings(signing_settings)
        .build()?;

    let url = format!(
        "https://{db_hostname}:{port}/?Action=connect&DBUser={db_user}",
        db_hostname = db_hostname,
        port = port,
        db_user = db_username
    );

    let signable_request =
```

```
        SignableRequest::new("GET", &url, std::iter::empty(),
 SignableBody::Bytes(&[]))
            .expect("signable request");

    let (signing_instructions, _signature) =
        sign(signable_request, &signing_params.into())?.into_parts();

    let mut url = url::Url::parse(&url).unwrap();
    for (name, value) in signing_instructions.params() {
        url.query_pairs_mut().append_pair(name, &value);
    }

    let response = url.to_string().split_off("https://".len());

    Ok(response)
}

#[tokio::main]
async fn main() -> Result<(), Error> {
    run(service_fn(handler)).await
}

async fn handler(_event: LambdaEvent<Value>) -> Result<Value, Error> {
    let db_host = env::var("DB_HOSTNAME").expect("DB_HOSTNAME must be set");
    let db_port = env::var("DB_PORT")
        .expect("DB_PORT must be set")
        .parse::<u16>()
        .expect("PORT must be a valid number");
    let db_name = env::var("DB_NAME").expect("DB_NAME must be set");
    let db_user_name = env::var("DB_USERNAME").expect("DB_USERNAME must be set");

    let token = generate_rds_iam_token(&db_host, db_port, &db_user_name).await?;

    let opts = PgConnectOptions::new()
        .host(&db_host)
        .port(db_port)
        .username(&db_user_name)
        .password(&token)
        .database(&db_name)
        .ssl_root_cert_from_pem(RDS_CERTS.to_vec())
        .ssl_mode(sqlx::postgres::PgSslMode::Require);

    let pool = sqlx::postgres::PgPoolOptions::new()
        .connect_with(opts)
```

```
        .await?;

    let result: i32 = sqlx::query_scalar("SELECT $1 + $2")
        .bind(3)
        .bind(2)
        .fetch_one(&pool)
        .await?;

    println!("Result: {:?}", result);

    Ok(json!({
        "statusCode": 200,
        "content-type": "text/plain",
        "body": format!("The selected sum is: {result}")
    }))
}
```

有关 Amazon SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 Amazon SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

Amazon RDS 中的安全性

Amazon 十分重视云安全性。作为 Amazon 客户，您将从专为满足大多数安全敏感型组织的要求而打造的数据中心和网络架构中受益。

安全性是 Amazon 和您的共同责任。责任共担模式将其描述为云的安全性和云中的安全性：

- 云的安全性 – Amazon负责保护在Amazon云中运行Amazon服务的基础架构。Amazon还向您提供可安全使用的服务。作为 Amazon 合规性计划的一部分，第三方审核人员将定期测试和验证安全性的有效性。要了解适用于 Amazon RDS的合规性计划，请参阅 Amazon 按合规性计划提供的范围内服务。
- 云中的安全性：您的责任由您使用的 Amazon 服务决定。您还需要对其他因素负责，包括您的数据的敏感性、您组织的要求以及适用的法律法规。

此文档将帮助您了解如何在使用 Amazon RDS 时应用责任共担模式。以下主题说明如何配置 Amazon RDS 以实现您的安全性和合规性目标。您还将了解如何使用其他 Amazon 服务来帮助您监控和保护 Amazon RDS 资源。

您可以管理对数据库实例上的 Amazon RDS 资源和数据库的访问。用来管理访问的方法取决于用户需要对 Amazon RDS 执行的任务类型：

- 在基于 Amazon VPC 服务的虚拟私有云（VPC）中运行数据库实例以获得尽可能大的网络访问控制能力。有关在 VPC 中创建数据库实例的更多信息，请参阅 Amazon VPC 和 Amazon RDS。
- 使用 Amazon Identity and Access Management（IAM）策略分配权限，来决定谁可以管理 Amazon RDS 资源。例如，您可以使用 IAM 确定哪些用户可以创建、描述、修改和删除数据库实例、为资源添加标签或修改安全组。
- 使用安全组可以控制哪些 IP 地址或 Amazon EC2 实例可连接到数据库实例上的数据库。首次创建数据库实例时，除非通过关联安全组指定的规则进行访问，否则实例防火墙会阻止任何数据库访问。
- 将安全套接字层（SSL）或传输层安全性协议（TLS）连接用于运行 Db2、MySQL、MariaDB、PostgreSQL、Oracle 或 Microsoft SQL Server 数据库引擎的数据库实例。有关将 SSL/TLS 用于数据库实例的更多信息，请参阅使用 SSL/TLS 加密与数据库实例或集群的连接。
- 使用 Amazon RDS 加密来保护您的数据库实例和静态快照。Amazon RDS 加密使用行业标准 AES-256 加密算法，来对托管您的数据库实例的服务器上的数据进行加密。有关更多信息，请参阅加密 Amazon RDS 资源。

- 对 Oracle 数据库实例使用网络加密和透明数据加密；有关更多信息，请参阅 Oracle 本机网络加密和 Oracle 透明数据加密
- 使用数据库引擎的安全功能，控制哪些用户可以登录数据库实例上的数据库。这些功能就像本地网络上的数据库一样工作。

> **Note**
> 您必须仅为您的使用案例配置安全性。您无需为 Amazon RDS 管理的过程配置安全访问。这些过程包括创建备份、在主数据库实例和只读副本之间复制数据以及其他过程。

有关管理对 Amazon RDS 资源和您的数据库实例上的数据库的访问的更多信息，请参阅以下主题。

主题

- Amazon RDS 的数据库身份验证
- 使用 Amazon RDS 和 Amazon Secrets Manager 管理密码
- Amazon RDS 中的数据保护
- Amazon RDS 的 Identity and Access Management
- Amazon RDS 中的日志记录和监控
- Amazon RDS 的合规性验证
- Amazon RDS 中的弹性
- Amazon RDS 中的基础设施安全性
- Amazon RDS API 和接口 VPC 终端节点 (Amazon PrivateLink)
- Amazon RDS 的安全最佳实践
- 使用安全组控制访问权限
- 主用户账户权限
- 将服务相关角色用于 Amazon RDS
- Amazon VPC 和 Amazon RDS

Amazon RDS 的数据库身份验证

Amazon RDS 支持多种对数据库用户进行身份验证的方法。

密码、Kerberos 和 IAM 数据库身份验证使用不同的方法对数据库进行身份验证。因此，特定用户只能使用一种身份验证方法登录到数据库。

对于 PostgreSQL，仅为特定数据库的用户使用以下角色设置之一：

- 要使用 IAM 数据库身份验证，请为用户分配 `rds_iam` 角色。
- 要使用 Kerberos 数据库身份验证，请为用户分配 `rds_ad` 角色。
- 要使用密码身份验证，请勿为用户分配 `rds_iam` 或 `rds_ad` 角色。

不要通过嵌套的授予访问权限直接或间接地将 `rds_iam` 和 `rds_ad` 角色分配给 PostgreSQL 数据库的用户。如果将 `rds_iam` 角色添加到主用户，IAM 身份验证优先于密码身份验证，因此主用户必须以 IAM 用户身份登录。

> **Important**
>
> 我们强烈建议不要直接在应用程序中使用主用户。请遵守使用数据库用户的最佳实践，按照您的应用程序所需的最少权限创建用户。

主题

- 密码验证
- IAM 数据库身份验证
- Kerberos 身份验证

密码验证

使用密码身份验证，数据库将执行用户账户的所有管理。您可以使用 `CREATE USER` 等 SQL 语句创建用户，其中包含数据库引擎指定密码所需的适当子句。例如，在 MySQL 中，语句为 `CREATE USER` *##* `IDENTIFIED BY` *##*，而在 PostgreSQL 中，语句为 `CREATE USER` *##* `WITH PASSWORD` *##*。

使用密码身份验证，数据库可控制和验证用户账户。如果数据库引擎具有强大的密码管理功能，那么可以增强安全性。当您拥有较小的用户社区时，使用密码身份验证可能更易于管理数据库身份验证。因为在这种情况下会生成明文密码，因此与 Amazon Secrets Manager 集成可以增强安全性。

有关将 Secrets Manager 与 Amazon RDS 结合使用的信息，请参阅《Amazon Secrets Manager User Guide》中的 Creating a basic secret 和 Rotating secrets for supported Amazon RDS databases。

有关以编程方式在自定义应用程序中检索密钥的信息，请参阅 Amazon Secrets Manager 用户指南中的检索密钥值。

IAM 数据库身份验证

您可以使用 Amazon Identity and Access Management（IAM）数据库身份验证对数据库实例进行身份验证。利用此身份验证方法，您在连接到数据库实例时将无需使用密码。而是使用身份验证令牌。

有关 IAM 数据库身份验证的更多信息，包括特定数据库引擎的可用性信息，请参阅 适用于 MariaDB、MySQL 和 PostgreSQL 的 IAM 数据库身份验证。

Kerberos 身份验证

Amazon RDS 支持使用 Kerberos 和 Microsoft Active Directory 对数据库用户进行外部身份验证。Kerberos 是一种网络身份验证协议，它使用票证和对称密钥加密，而不再需要通过网络传输密码。Kerberos 已内置到 Active Directory 中，用于在网络资源（如数据库）中对用户进行身份验证。

Amazon RDS 支持 Kerberos 和 Active Directory，从而为数据库用户提供单一登录和集中身份验证的好处。您可以将用户凭证保存在 Active Directory 中。Active Directory 提供了一个集中位置，以存储和管理多个数据库实例的凭证。

要使用来自自行管理的 Active Directory 的凭证，您需要针对数据库实例加入的 Microsoft Active Directory 设置与 Amazon Directory Service 的信任关系。

RDS for PostgreSQL 和 RDS for MySQL 通过全林性身份验证或选择性身份验证，支持单向和双向林信任关系。

在某些情况下，可以通过外部信任关系配置 Kerberos 身份验证。这要求自行管理的 Active Directory 具有更多设置。这包括但不限于 Kerberos Forest Search Order。

Microsoft SQL Server 和 PostgreSQL 数据库实例支持单向和双向林信任关系。Oracle 数据库实例支持单向和双向外部和林信任关系。有关更多信息，请参阅 Amazon Directory Service 管理指南 中的何时创建信任关系。

有关针对特定数据库引擎的 Kerberos 身份验证的信息，请参阅以下内容：

- 将 Amazon Managed Active Directory 用于 RDS for SQL Server
- 对 Amazon RDS for MySQL 进行 Kerberos 身份验证
- 为 Amazon RDS for Oracle 配置 Kerberos 身份验证
- 在 Amazon RDS for PostgreSQL 中使用 Kerberos 身份验证

- 将 Kerberos 身份验证用于 Amazon RDS for Db2 .

> **Note**
>
> 目前，MariaDB 数据库实例不支持 Kerberos 身份验证。

使用 Amazon RDS 和 Amazon Secrets Manager 管理密码

Amazon RDS 集成了 Secrets Manager，以管理您的数据库实例和多可用区数据库集群的主用户密码。

主题

Secrets Manager 与 Amazon RDS 集成的限制

以下功能不支持使用 Secrets Manager 管理主用户密码：

- 在源数据库或数据库集群使用 Secrets Manager 管理凭证时创建只读副本。这适用于除 RDS for SQL Server 之外的所有数据库引擎。
- Amazon RDS 蓝绿部署
- Amazon RDS Custom
- Oracle Data Guard 切换

使用 Amazon Secrets Manager 管理主用户密码的概述

借助 Amazon Secrets Manager，您可以将代码中的硬编码凭证（包括数据库密码）替换为对 Secrets Manager 的 API 调用，从而以编程方式检索密钥。有关 Secrets Manager 的更多信息，请参阅《Amazon Secrets Manager 用户指南》https://docs.amazonaws.cn/secretsmanager/latest/userguide/。

当您在 Secrets Manager 中存储数据库密钥时，您的 Amazon Web Services 账户会产生费用。有关定价的信息，请参阅 Amazon Secrets Manager 定价。

在执行以下操作之一时，您可以指定 RDS 在 Secrets Manager 中管理 Amazon RDS 数据库实例或多可用区数据库集群的主用户密码：

- 创建数据库实例
- 创建多可用区数据库集群
- 在 RDS for Oracle CDB 中创建租户数据库
- 修改数据库实例
- 修改多可用区数据库集群
- 修改租户数据库（仅限 RDS for Oracle）
- 从 Amazon S3 还原数据库实例
- 将数据库实例从快照还原到某个时间点（仅限 RDS for Oracle）

当您指定 RDS 在 Secrets Manager 中管理主用户密码时，RDS 会生成密码并将其存储在 Secrets Manager 中。您可以直接与密钥交互以检索主用户的凭证。您还可以指定客户托管密钥来加密密钥，或者使用 Secrets Manager 提供的 KMS 密钥。

RDS 管理密钥的设置，默认情况下每七天轮换一次密钥。您可以修改某些设置，例如轮换计划。如果您删除在 Secrets Manager 中管理密钥的数据库实例，则该密钥及其关联的元数据也会被删除。

要使用密钥中的凭证连接到数据库实例或多可用区数据库集群，您可以从 Secrets Manager 检索密钥。有关更多信息，请参阅《Amazon Secrets Manager 用户指南》中的从 Amazon Secrets Manager 中检索密钥和使用 Amazon Secrets Manager 密钥中的凭证连接到 SQL 数据库。

使用 Secrets Manager 管理主用户密码的优势

使用 Secrets Manager 管理 RDS 主用户密码具有以下优势：

- RDS 会自动生成数据库凭证。

- RDS 会自动在 Amazon Secrets Manager 中存储和管理数据库凭证。
- RDS 定期轮换数据库凭证，而无需更改应用程序。
- Secrets Manager 保护数据库凭证免受人员访问和纯文本视图的影响。
- Secrets Manager 允许在密钥中检索数据库凭证以进行数据库连接。
- Secrets Manager 允许使用 IAM 细粒度控制对密钥中的数据库凭证的访问权限。
- 您可以选择使用不同的 KMS 密钥将数据库加密与凭证加密分开。
- 您可以省去手动管理和轮换数据库凭证。
- 您可以使用 Amazon CloudTrail 和 Amazon CloudWatch 轻松监控数据库凭证。

有关 Secrets Manager 的优势的更多信息，请参阅 Amazon Secrets Manager 用户指南。

Secrets Manager 集成所需的权限

用户必须拥有所需的权限才能执行与 Secrets Manager 集成相关的操作。您可以创建 IAM policy，以便授予权限对所需的指定资源执行特定的 API 操作。然后，可以将这些策略附加到需要这些权限的 IAM 权限集或角色。有关更多信息，请参阅 Amazon RDS 的 Identity and Access Management。

对于创建、修改或还原操作，指定 Amazon RDS 在 Secrets Manager 中管理主用户密码的用户必须具有执行以下操作的权限：

- `kms:DescribeKey`
- `secretsmanager:CreateSecret`
- `secretsmanager:TagResource`

需要 `kms:DescribeKey` 权限才能访问 `MasterUserSecretKmsKeyId` 的客户托管密钥并描述 `aws/secretsmanager`。

对于创建、修改或还原操作，指定用于在 Secrets Manager 中加密密钥的客户托管密钥的用户必须具有执行以下操作的权限：

- `kms:Decrypt`
- `kms:GenerateDataKey`
- `kms:CreateGrant`

对于修改操作，在 Secrets Manager 中轮换主用户密码的用户必须具有执行以下操作的权限：

- secretsmanager:RotateSecret

强制 RDS 在 Amazon Secrets Manager 中管理主用户密码

您可以使用 IAM 条件密钥强制 RDS 在 Amazon Secrets Manager 中管理主用户密码。除非主用户密码由 RDS 在 Secrets Manager 中进行管理，否则以下策略不支持用户创建或还原数据库实例或数据库集群，或者创建或修改租户数据库。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": ["rds:CreateDBInstance", "rds:CreateDBCluster",
 "rds:RestoreDBInstanceFromS3", "rds:RestoreDBClusterFromS3",
                       "rds:RestoreDBInstanceFromDBSnapshot",
 "rds:RestoreDBInstanceToPointInTime", "rds:CreateTenantDatabase",
                       "rds:ModifyTenantDatabase"],
            "Resource": "*",
            "Condition": {
                "Bool": {
                    "rds:ManageMasterUserPassword": false
                }
            }
        }
    ]
}
```

> **Note**
>
> 此策略在创建时在 Amazon Secrets Manager 中强制执行密码管理。但是，您仍然可以禁用 Secrets Manager 集成，并通过修改实例手动设置主密码。
> 为防止这种情况，请在策略的操作块中包括 `rds:ModifyDBInstance`、`rds:ModifyDBCluster`。请注意，这可以防止用户对未启用 Secrets Manager 集成的现有实例进行任何进一步修改。

有关在 IAM 策略中使用条件密钥的更多信息，请参阅 Amazon RDS 的策略条件键 和 示例策略：使用条件键。

使用 Secrets Manager 管理数据库实例的主用户密码

执行以下操作时，可以配置 RDS 在 Secrets Manager 中管理主用户密码：

- 创建 Amazon RDS 数据库实例
- 修改 Amazon RDS 数据库实例
- 将备份还原到 Amazon RDS for MySQL 数据库实例
- 还原到数据库实例（仅限 RDS for Oracle）
- 将 Amazon RDS 的数据库实例还原到指定时间（仅限 RDS for Oracle）

您可以使用 RDS 控制台、Amazon CLI 或 RDS API 来执行上述操作。

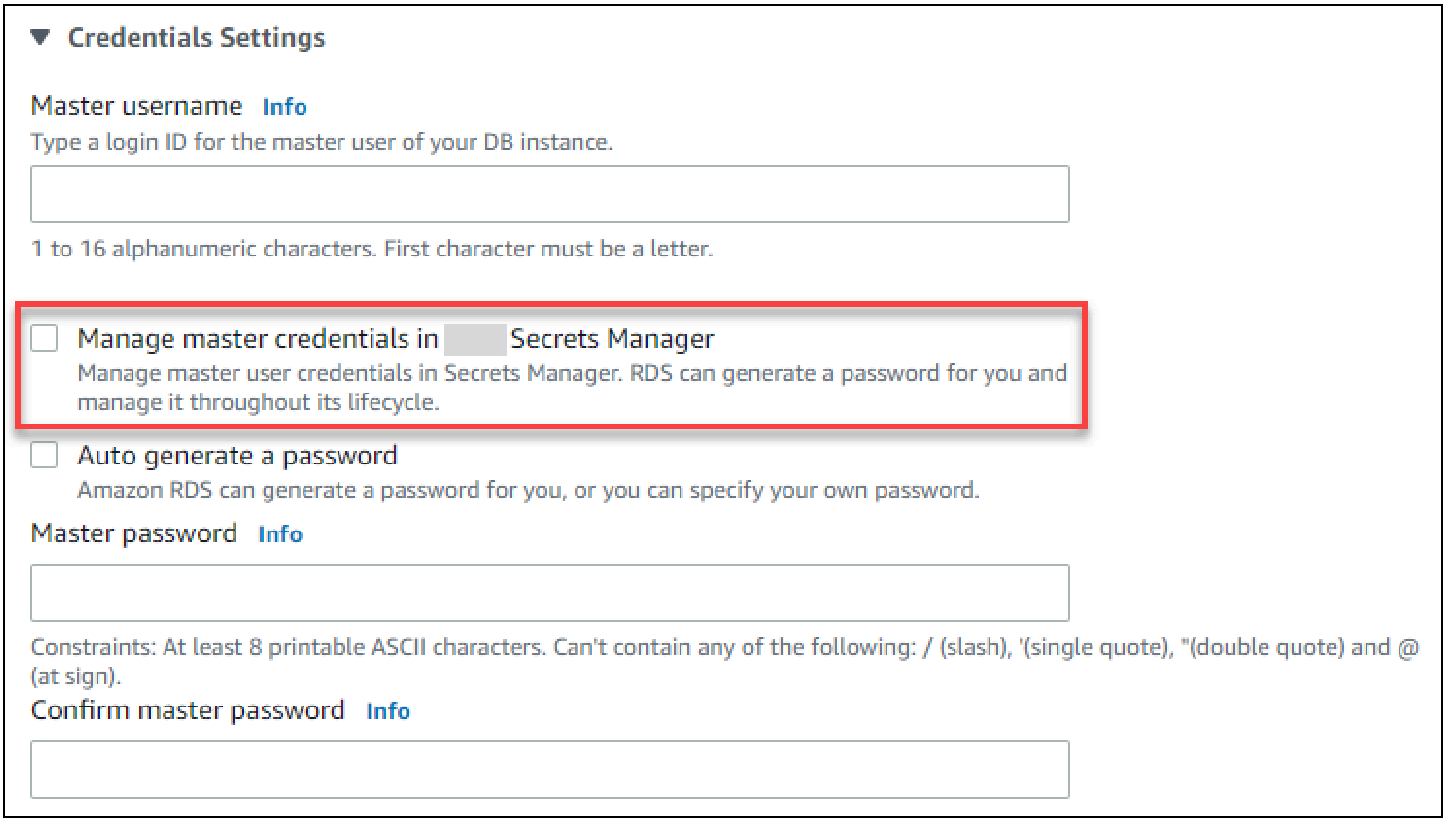
控制台

按照说明使用 RDS 控制台创建或修改数据库实例：

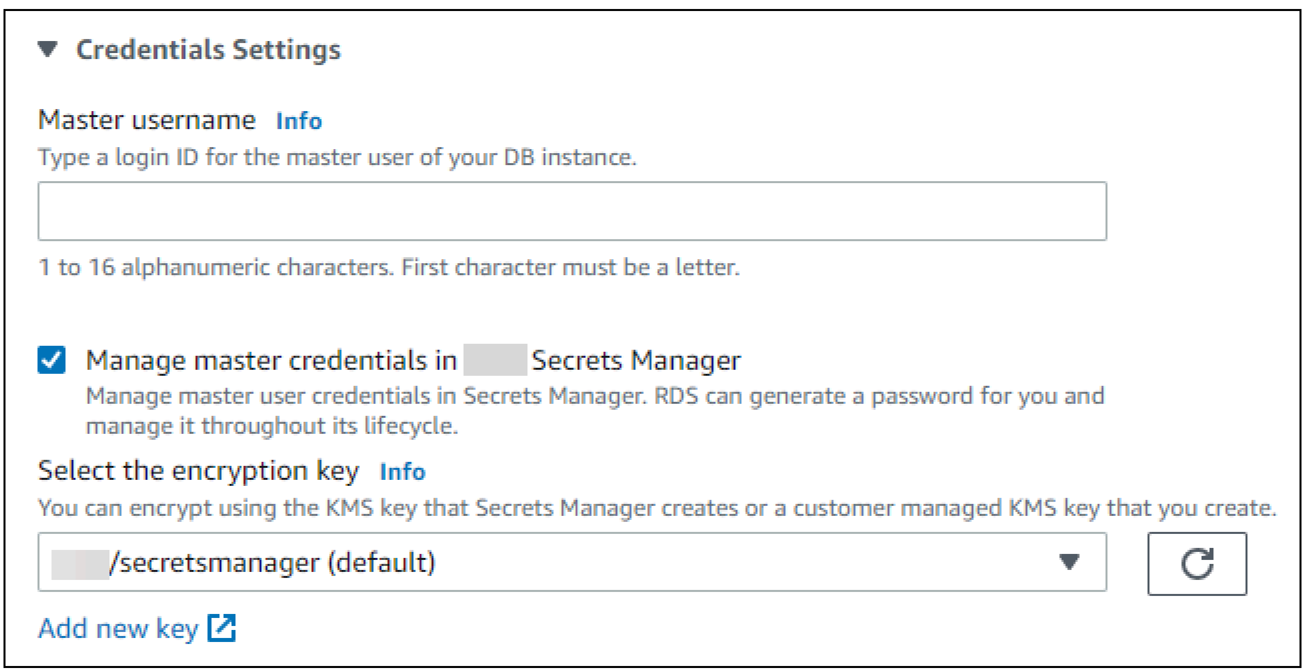
- 创建数据库实例
- 修改 Amazon RDS 数据库实例
- 将数据从 Amazon S3 导入到新的 MySQL 数据库实例

当您使用 RDS 控制台执行其中一项操作时，可以在 Secrets Manager 中指定主用户密码由 RDS 管理。当您创建或还原数据库实例时，请在凭证设置中选择在 Amazon Secrets Manager 中管理主凭证。修改数据库实例时，请在设置中选择在 Amazon Secrets Manager 中管理主凭证。

下图是在创建或还原数据库实例时在 Amazon Secrets Manager 中管理主凭证设置的示例。



当您选择此选项时，RDS 会生成主用户密码并在其整个生命周期中在 Secrets Manager 中对其进行管理。



您可以选择使用 Secrets Manager 提供的 KMS 密钥或您创建的客户托管密钥对密钥进行加密。在 RDS 管理数据库实例的数据库凭证后，您无法更改用于加密密钥的 KMS 密钥。

您可以选择其他设置来满足您的要求。有关创建数据库实例时的可用设置的更多信息，请参阅数据库实例的设置。有关修改数据库实例时的可用设置的更多信息，请参阅数据库实例的设置。

Amazon CLI

要在 Secrets Manager 中使用 RDS 管理主用户密码，请在以下 Amazon CLI 命令之一中指定 `--manage-master-user-password` 选项：

- create-db-instance
- modify-db-instance
- restore-db-instance-from-s3
- restore-db-instance-from-db-snapshot（仅限 RDS for Oracle）
- restore-db-instance-to-point-in-time（仅限 RDS for Oracle）

当您在这些命令中指定 `--manage-master-user-password` 选项时，RDS 会生成主用户密码，并在其整个生命周期中在 Secrets Manager 中对其进行管理。

要加密密钥，您可以指定客户托管式密钥或使用 Secrets Manager 提供的原定设置 KMS 密钥。使用 `--master-user-secret-kms-key-id` 选项指定客户托管密钥。Amazon KMS 密钥标识符是密钥 ARN、密钥 ID、别名 ARN 或者 KMS 密钥的别名。要使用不同 Amazon Web Services 账户中的密钥，请指定密钥 ARN 或别名 ARN。在 RDS 管理数据库实例的数据库凭证后，您无法更改用于加密密钥的 KMS 密钥。

您可以选择其他设置来满足您的要求。有关创建数据库实例时的可用设置的更多信息，请参阅 数据库实例的设置。有关修改数据库实例时的可用设置的更多信息，请参阅 数据库实例的设置。

以下示例创建一个数据库实例，并指定 RDS 在 Secrets Manager 中管理主用户密码。此密钥使用 Secrets Manager 提供的 KMS 密钥进行加密。

Example

对于 Linux、macOS 或 Unix：

```
aws rds create-db-instance \
    --db-instance-identifier mydbinstance \
    --engine mysql \
    --engine-version 8.0.39 \
    --db-instance-class db.r5b.large \
```

```
    --allocated-storage 200 \
    --master-username testUser \
    --manage-master-user-password
```

对于 Windows：

```
aws rds create-db-instance ^
    --db-instance-identifier mydbinstance ^
    --engine mysql ^
    --engine-version 8.0.39 ^
    --db-instance-class db.r5b.large ^
    --allocated-storage 200 ^
    --master-username testUser ^
    --manage-master-user-password
```

RDS API

要指定 RDS 在 Secrets Manager 中管理主用户密码，请在以下 RDS API 操作之一中将 `ManageMasterUserPassword` 参数设置为 `true`：

- CreateDBInstance
- ModifyDBInstance
- RestoreDBInstanceFromS3
- RestoreDBInstanceFromSnapshot（仅限 RDS for Oracle）
- RestoreDBInstanceToPointInTime（仅限 RDS for Oracle）

当您在其中一个操作中将 `ManageMasterUserPassword` 参数设置为 `true` 时，RDS 会生成主用户密码并在其整个生命周期中在 Secrets Manager 中对其进行管理。

要加密密钥，您可以指定客户托管式密钥或使用 Secrets Manager 提供的原定设置 KMS 密钥。使用 `MasterUserSecretKmsKeyId` 参数指定客户托管密钥。Amazon KMS 密钥标识符是密钥 ARN、密钥 ID、别名 ARN 或者 KMS 密钥的别名。要使用不同 Amazon Web Services 账户中的密钥，请指定密钥 ARN 或别名 ARN。在 RDS 管理数据库实例的数据库凭证后，您无法更改用于加密密钥的 KMS 密钥。

使用 Secrets Manager 管理 RDS for Oracle 租户数据库的主用户密码

执行以下操作时，可以配置 RDS 在 Secrets Manager 中管理主用户密码：

- [将 RDS for Oracle 租户数据库添加到您的 CDB 实例]
- [修改 RDS for Oracle 租户数据库]

您可以使用 RDS 控制台、Amazon CLI 或 RDS API 来执行前面的操作。

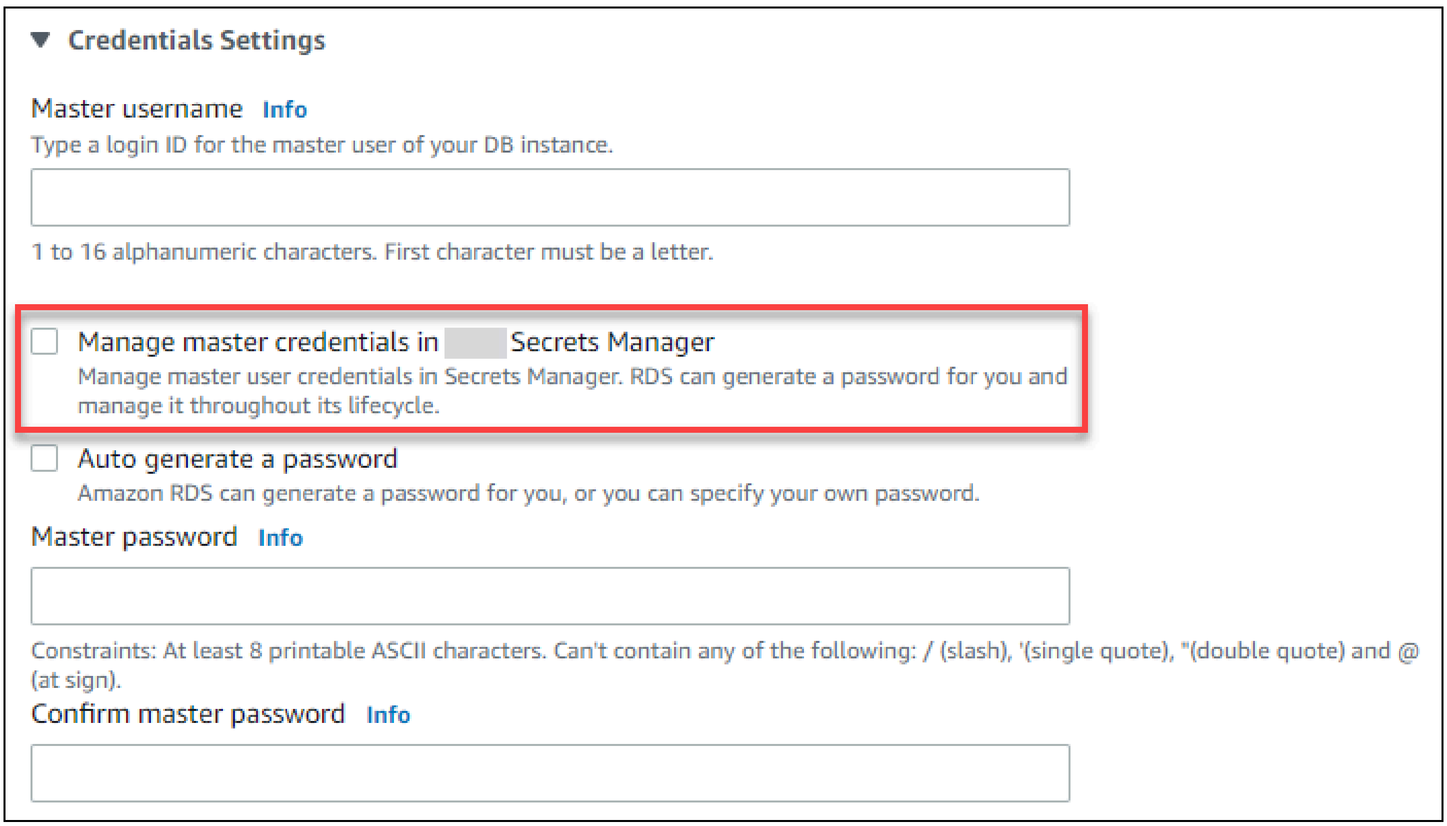
控制台

按照使用 RDS 控制台创建或修改 RDS for Oracle 租户数据库的说明进行操作：

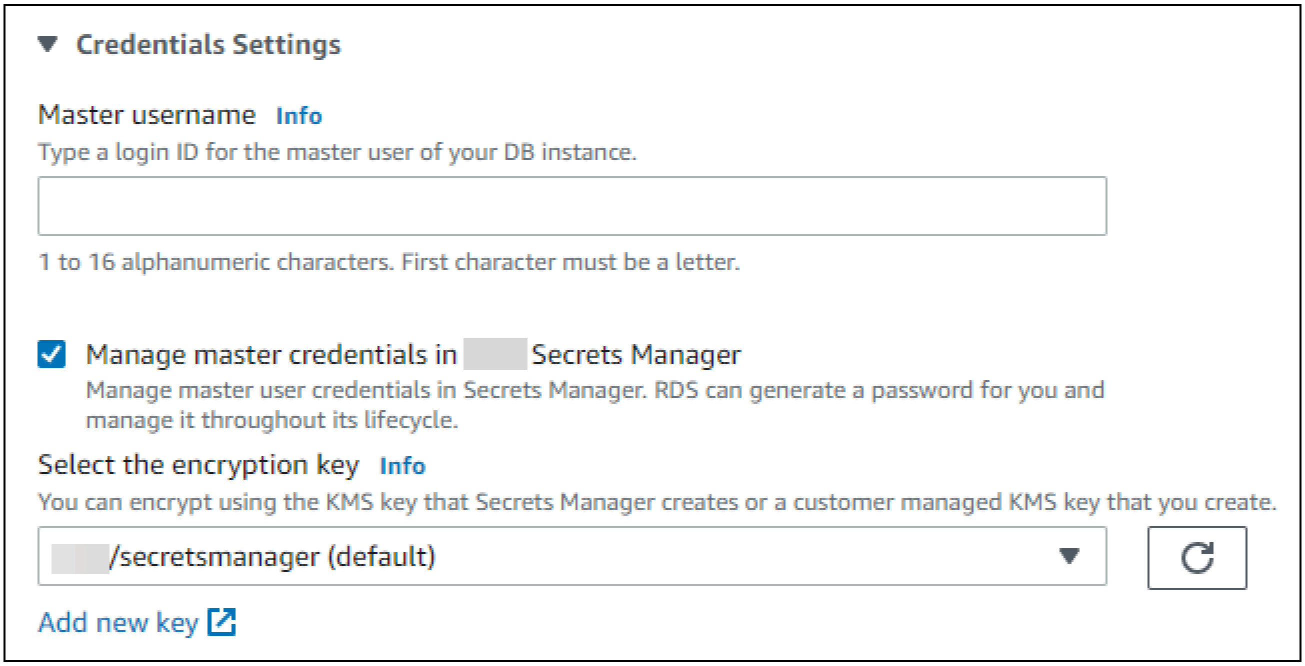
- [将 RDS for Oracle 租户数据库添加到您的 CDB 实例]
- [修改 RDS for Oracle 租户数据库]

当您使用 RDS 控制台执行上述操作之一时，您可以指定 RDS 在 Secrets Manager 中管理主密码。创建租户数据库时，请在凭证设置中选择在 Amazon Secrets Manager 中管理主凭证。修改租户数据库时，请在设置中选择在 Amazon Secrets Manager 中管理主凭证。

下图是在创建租户数据库时在 Amazon Secrets Manager 中管理主凭证设置的示例。



当您选择此选项时，RDS 会生成主用户密码并在其整个生命周期中在 Secrets Manager 中对其进行管理。



您可以选择使用 Secrets Manager 提供的 KMS 密钥或您创建的客户托管密钥对密钥进行加密。在 RDS 管理租户数据库的数据库凭证后，您无法更改用于加密密钥的 KMS 密钥。

您可以选择其他设置来满足您的要求。有关创建租户数据库时的可用设置的更多信息，请参阅数据库实例的设置。有关修改租户数据库时的可用设置的更多信息，请参阅数据库实例的设置。

Amazon CLI

要在 Secrets Manager 中使用 RDS 管理主用户密码，请在以下 Amazon CLI 命令之一中指定 `--manage-master-user-password` 选项：

- create-tenant-database
- modify-tenant-database

当您在前面的命令中指定 `--manage-master-user-password` 选项时，RDS 会生成主用户密码，并在其整个生命周期内在 Secrets Manager 中管理此密码。

要加密密钥，您可以指定客户托管密钥或使用 Secrets Manager 提供的默认 KMS 密钥。使用 `--master-user-secret-kms-key-id` 选项指定客户托管密钥。Amazon KMS 密钥标识符是密钥 ARN、密钥 ID、别名 ARN 或者 KMS 密钥的别名。要使用不同 Amazon Web Services 账户中的密钥，请指定密钥 ARN 或别名 ARN。在 RDS 管理租户数据库的数据库凭证后，您无法更改用于加密密钥的 KMS 密钥。

您可以选择其他设置来满足您的要求。有关创建租户数据库时的可用设置的更多信息，请参阅 create-tenant-database。有关修改租户数据库时的可用设置的更多信息，请参阅 modify-tenant-database。

以下示例创建一个 RDS for Oracle 租户数据库，并指定 RDS 在 Secrets Manager 中管理主用户密码。此密钥使用 Secrets Manager 提供的 KMS 密钥进行加密。

Example

对于 Linux、macOS 或 Unix：

```
aws rds create-tenant-database --region us-east-1 \
    --db-instance-identifier my-cdb-inst \
    --tenant-db-name mypdb2 \
    --master-username mypdb2-admin \
    --character-set-name UTF-16 \
    --manage-master-user-password
```

对于 Windows：

```
aws rds create-tenant-database --region us-east-1 ^
    --db-instance-identifier my-cdb-inst ^
    --tenant-db-name mypdb2 ^
    --master-username mypdb2-admin ^
    --character-set-name UTF-16 ^
    --manage-master-user-password
```

RDS API

要指定 RDS 在 Secrets Manager 中管理主用户密码，请在以下 RDS API 操作之一中将 `ManageMasterUserPassword` 参数设置为 `true`：

- CreateTenantDatabase
- ModifyTenantDatabase

当您在其中一个操作中将 `ManageMasterUserPassword` 参数设置为 `true` 时，RDS 会生成主用户密码并在其整个生命周期中在 Secrets Manager 中对其进行管理。

要加密密钥，您可以指定客户托管式密钥或使用 Secrets Manager 提供的原定设置 KMS 密钥。使用 `MasterUserSecretKmsKeyId` 参数指定客户托管密钥。Amazon KMS 密钥标识符是密钥 ARN、密钥 ID、别名 ARN 或者 KMS 密钥的别名。要使用不同 Amazon Web Services 账户中的密钥，请指定

密钥 ARN 或别名 ARN。在 RDS 管理租户数据库的数据库凭证后，您无法更改用于加密密钥的 KMS 密钥。

使用 Secrets Manager 管理多可用区数据库集群的主用户密码

执行以下操作时，可以配置 RDS 在 Secrets Manager 中管理主用户密码：

- 创建 Amazon RDS 的多可用区数据库集群
- 修改 Amazon RDS 的多可用区数据库集群

您可以使用 RDS 控制台、Amazon CLI 或 RDS API 来执行这些操作。

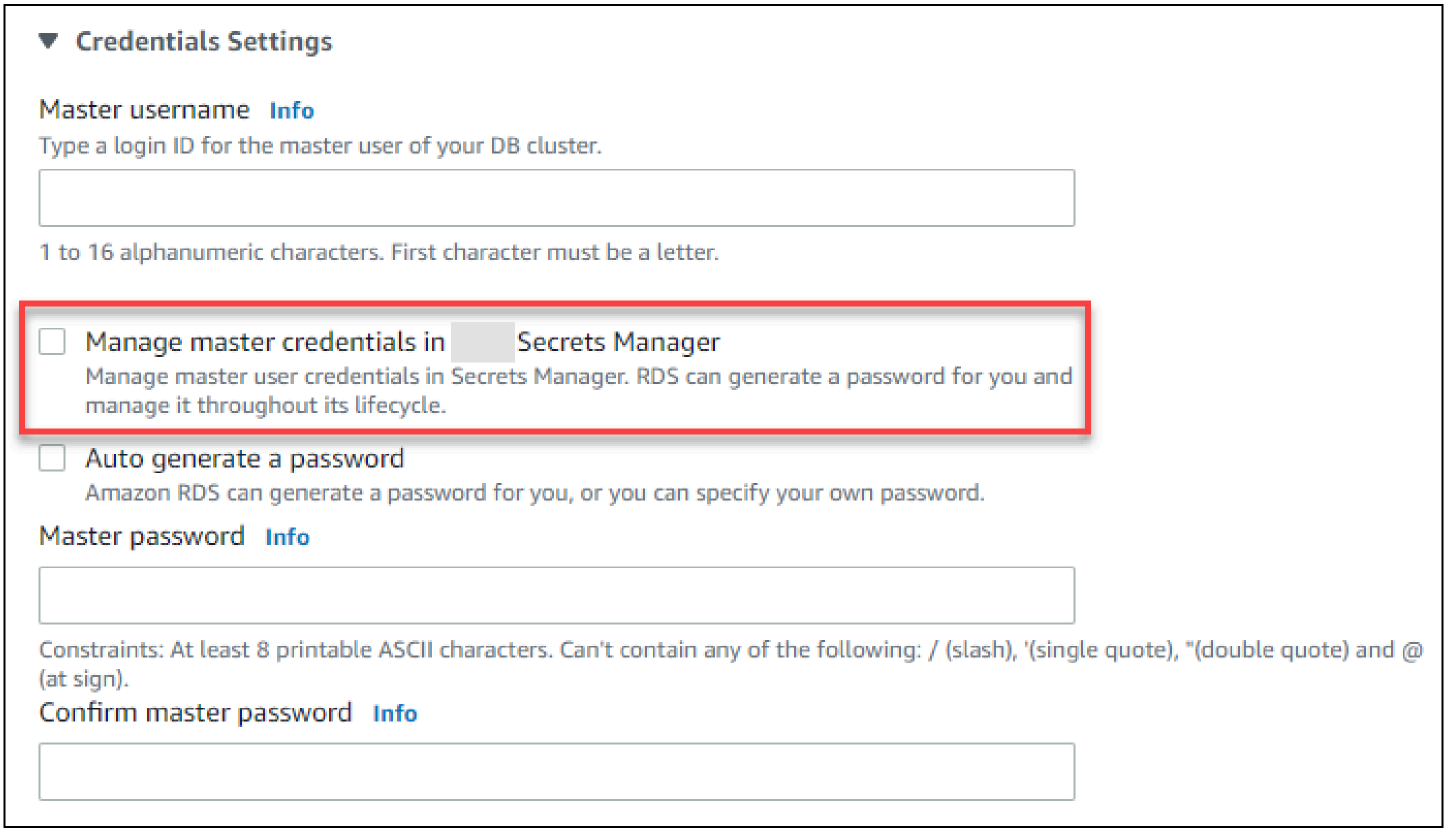
控制台

按照说明使用 RDS 控制台创建或修改多可用区数据库集群：

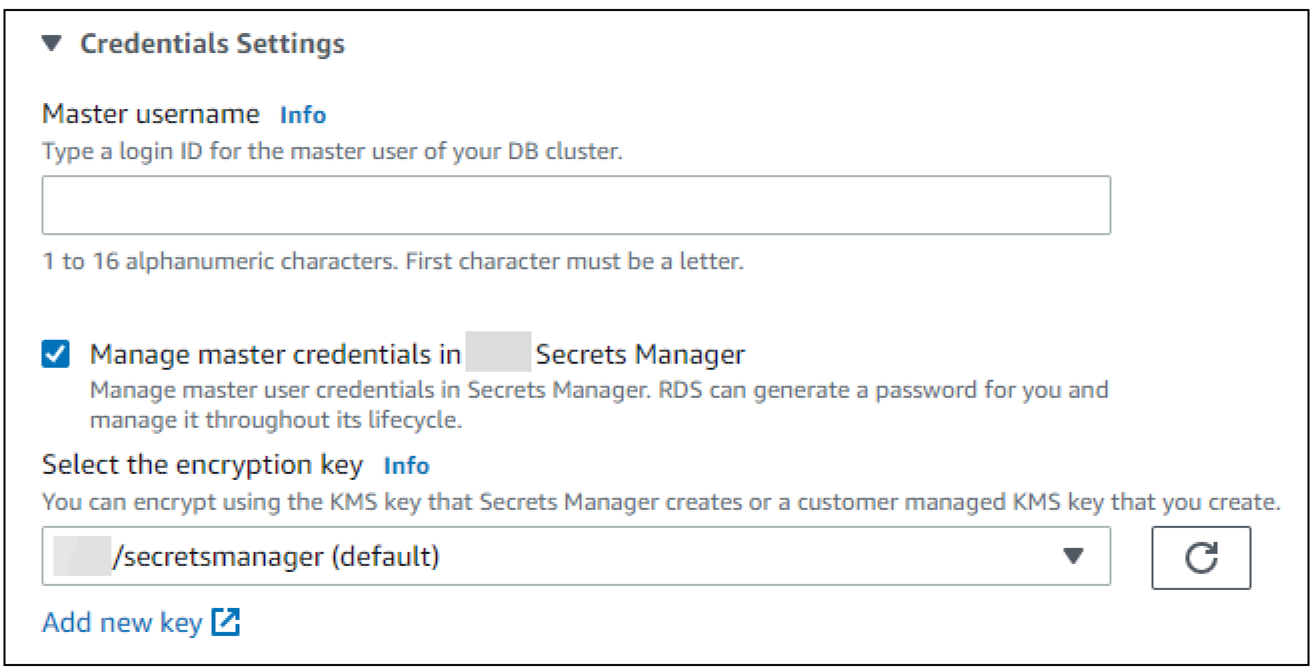
- 创建数据库集群
- 修改 Amazon RDS 的多可用区数据库集群

当您使用 RDS 控制台执行其中一项操作时，可以在 Secrets Manager 中指定主用户密码由 RDS 管理。要在创建数据库集群时执行此操作，请在凭证设置中选择在 Amazon Secrets Manager 中管理主凭证。修改数据库集群时，请在设置中选择在 Amazon Secrets Manager 中管理主凭证。

下图是在创建数据库集群时在 Amazon Secrets Manager 中管理主凭证的示例。



当您选择此选项时，RDS 会生成主用户密码并在其整个生命周期中在 Secrets Manager 中对其进行管理。



您可以选择使用 Secrets Manager 提供的 KMS 密钥或您创建的客户托管密钥对密钥进行加密。在 RDS 管理数据库集群的数据库凭证后，您无法更改用于加密密钥的 KMS 密钥。

您可以选择其他设置来满足您的要求。

有关创建多可用区数据库集群时的可用设置的更多信息，请参阅 创建多可用区数据库集群的设置。有关修改多可用区数据库集群时的可用设置的更多信息，请参阅 修改多可用区数据库集群的设置。

Amazon CLI

要指定 RDS 在 Secrets Manager 中管理主用户密码，请在以下命令之一中指定 `--manage-master-user-password` 选项：

- create-db-cluster
- modify-db-cluster

当您在这些命令中指定 `--manage-master-user-password` 选项时，RDS 会生成主用户密码，并在其整个生命周期中在 Secrets Manager 中对其进行管理。

要加密密钥，您可以指定客户托管密钥或使用 Secrets Manager 提供的默认 KMS 密钥。使用 `--master-user-secret-kms-key-id` 选项指定客户托管密钥。Amazon KMS 密钥标识符是密钥 ARN、密钥 ID、别名 ARN 或者 KMS 密钥的别名。要使用不同 Amazon Web Services 账户中的密钥，请指定密钥 ARN 或别名 ARN。在 RDS 管理数据库集群的数据库凭证后，您无法更改用于加密密钥的 KMS 密钥。

您可以选择其他设置来满足您的要求。

有关创建多可用区数据库集群时的可用设置的更多信息，请参阅 创建多可用区数据库集群的设置。有关修改多可用区数据库集群时的可用设置的更多信息，请参阅 修改多可用区数据库集群的设置。

此示例创建一个多可用区数据库集群，并指定 RDS 在 Secrets Manager 中管理密码。此密钥使用 Secrets Manager 提供的 KMS 密钥进行加密。

Example

对于 Linux、macOS 或 Unix：

```
aws rds create-db-cluster \
   --db-cluster-identifier mysql-multi-az-db-cluster \
   --engine mysql \
   --engine-version 8.0.39  \
   --backup-retention-period 1  \
   --allocated-storage 4000 \
   --storage-type io1 \
```

```
    --iops 10000 \
    --db-cluster-instance-class db.r6gd.xlarge \
    --master-username testUser \
    --manage-master-user-password
```

对于 Windows：

```
aws rds create-db-cluster ^
    --db-cluster-identifier mysql-multi-az-db-cluster ^
    --engine mysql ^
    --engine-version 8.0.39 ^
    --backup-retention-period 1 ^
    --allocated-storage 4000 ^
    --storage-type io1 ^
    --iops 10000 ^
    --db-cluster-instance-class db.r6gd.xlarge ^
    --master-username testUser ^
    --manage-master-user-password
```

RDS API

要指定 RDS 在 Secrets Manager 中管理主用户密码，请在以下操作之一中将 `ManageMasterUserPassword` 参数设置为 `true`：

- CreateDBCluster
- ModifyDBCluster

当您在其中一个操作中将 `ManageMasterUserPassword` 参数设置为 `true` 时，RDS 会生成主用户密码并在其整个生命周期中在 Secrets Manager 中对其进行管理。

要加密密钥，您可以指定客户托管式密钥或使用 Secrets Manager 提供的原定设置 KMS 密钥。使用 `MasterUserSecretKmsKeyId` 参数指定客户托管密钥。Amazon KMS 密钥标识符是密钥 ARN、密钥 ID、别名 ARN 或者 KMS 密钥的别名。要使用不同 Amazon Web Services 账户中的密钥，请指定密钥 ARN 或别名 ARN。在 RDS 管理数据库集群的数据库凭证后，您无法更改用于加密密钥的 KMS 密钥。

轮换数据库实例的主用户密码密钥

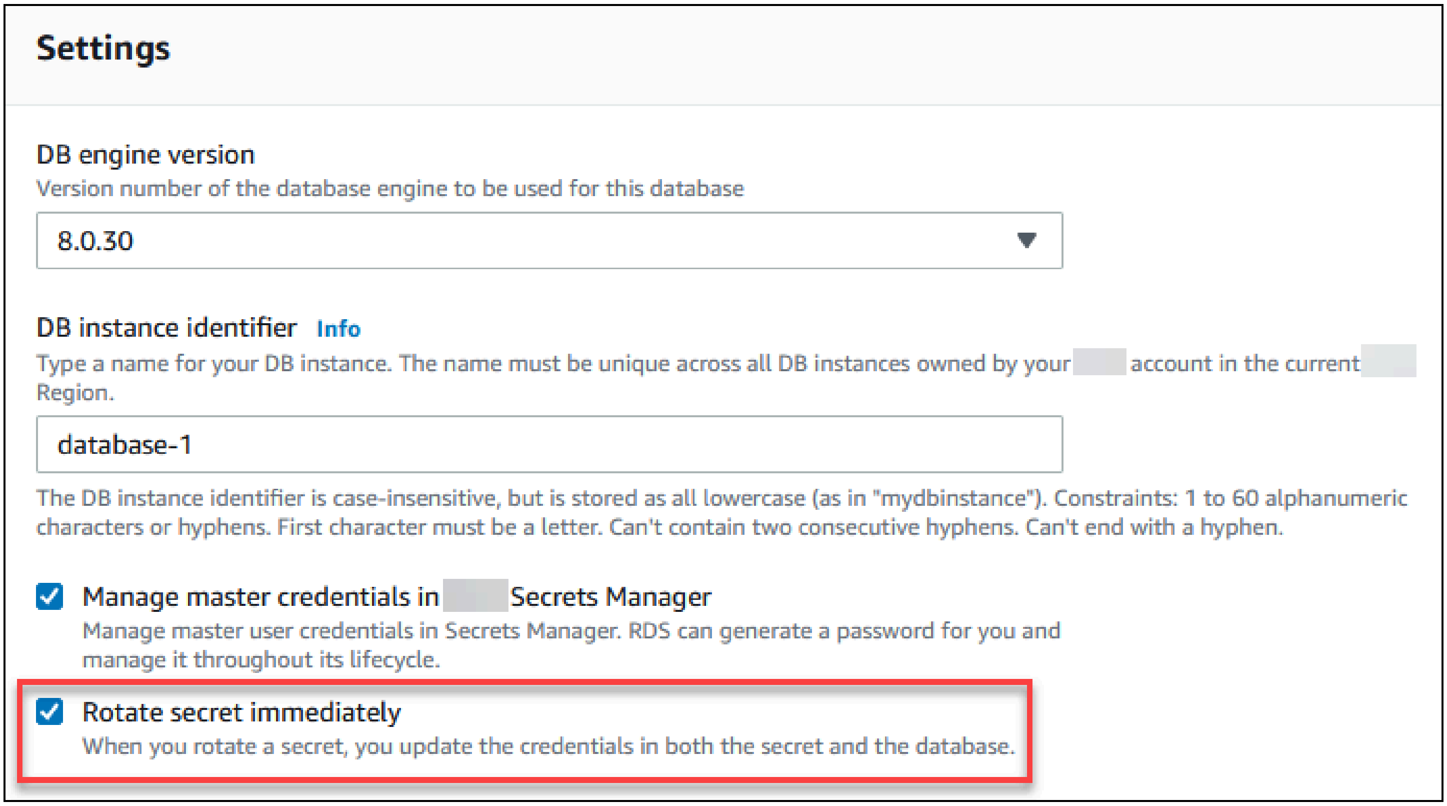
当 RDS 轮换主用户密码密钥时，Secrets Manager 会为现有密钥生成一个新的密钥版本。密钥的新版本包含新的主用户密码。Amazon RDS 更改数据库实例的主用户密码以匹配新密钥版本的密码。

您可以立即轮换密钥，而不必等待计划的轮换。要在 Secrets Manager 中轮换主用户密码密钥，请修改数据库实例。有关修改数据库实例的信息，请参阅[修改 Amazon RDS 数据库实例](#)。

您可以使用 RDS 控制台、Amazon CLI 或 RDS API 立即轮换主用户密码密钥。新密码长度始终为 28 个字符，包含至少一个大写和小写字母、一个数字和一个标点符号。

控制台

要使用 RDS 控制台轮换主用户密码密钥，请修改数据库实例并在 Settings（设置）中选择 Rotate secret immediately（立即轮换密钥）。



按照[修改 Amazon RDS 数据库实例](#)中的说明使用 RDS 控制台修改数据库实例。您必须在确认页面上选择 Apply immediately（立即应用）。

Amazon CLI

要使用 Amazon CLI 轮换主用户密码密钥，请使用 [modify-db-instance](#) 命令并指定 `--rotate-master-user-password` 选项。轮换主密码时必须指定 `--apply-immediately` 选项。

此示例轮换主用户密码密钥。

Example

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier mydbinstance \
    --rotate-master-user-password \
    --apply-immediately
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier mydbinstance ^
    --rotate-master-user-password ^
    --apply-immediately
```

RDS API

您可以使用 ModifyDBInstance 操作并将 `RotateMasterUserPassword` 参数设置为 `true` 来轮换主用户密码密钥。轮换主密码时，必须将 `ApplyImmediately` 参数设置为 `true`。

轮换多可用区数据库集群的主用户密码密钥

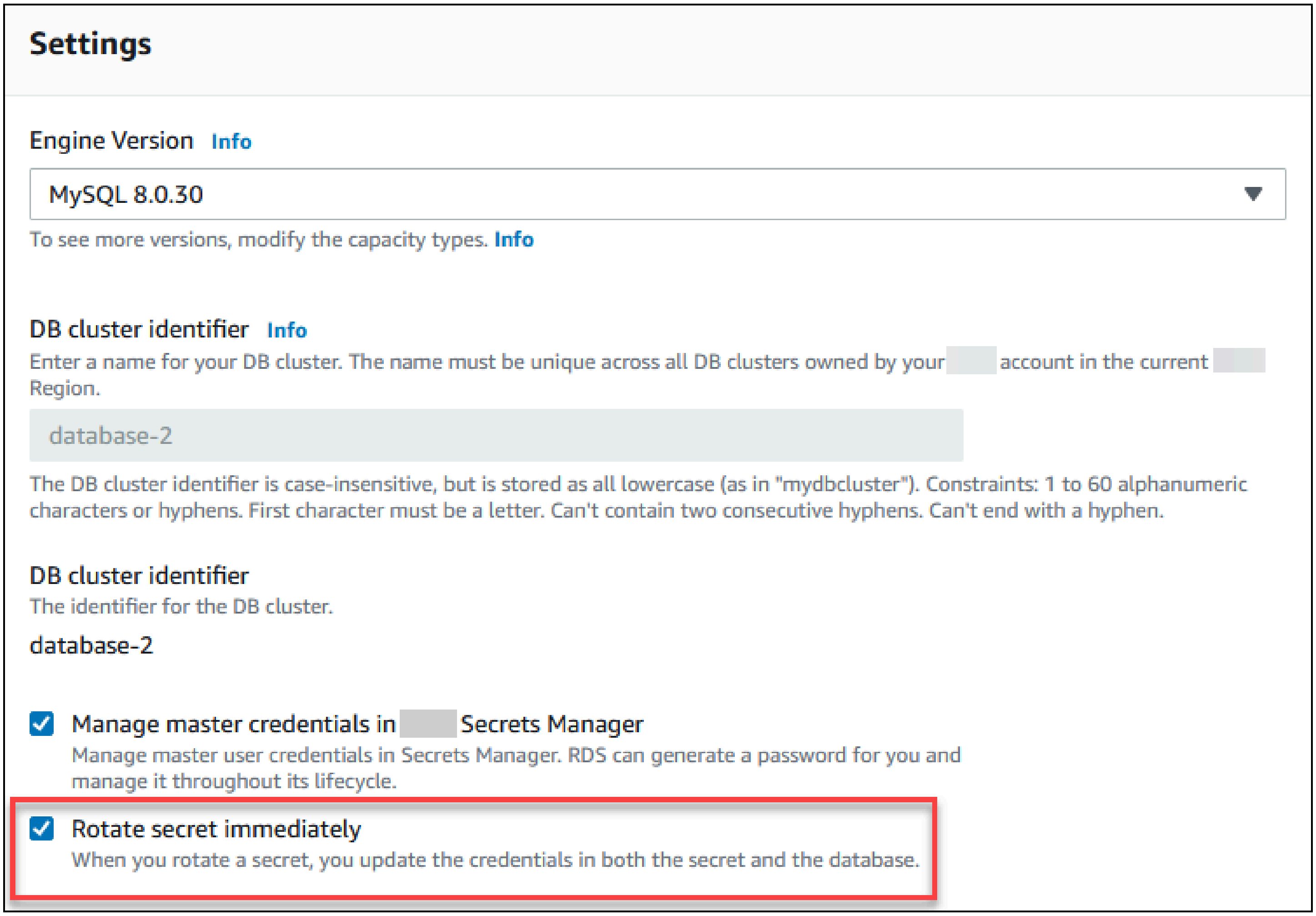
当 RDS 轮换主用户密码密钥时，Secrets Manager 会为现有密钥生成一个新的密钥版本。密钥的新版本包含新的主用户密码。Amazon RDS 更改多可用区数据库集群的主用户密码，以匹配新密钥版本的密码。

您可以立即轮换密钥，而不必等待计划的轮换。要在 Secrets Manager 中轮换主用户密码密钥，请修改多可用区数据库集群。有关修改多可用区数据库集群的信息，请参阅 修改 Amazon RDS 的多可用区数据库集群。

您可以使用 RDS 控制台、Amazon CLI 或 RDS API 立即轮换主用户密码密钥。新密码长度始终为 28 个字符，包含至少一个大写和小写字母、一个数字和一个标点符号。

控制台

要使用 RDS 控制台轮换主用户密码密钥，请修改多可用区数据库集群并在 Settings（设置）中选择 Rotate secret immediately（立即轮换密钥）。



按照修改 Amazon RDS 的多可用区数据库集群中的说明使用 RDS 控制台修改多可用区数据库集群。您必须在确认页面上选择 Apply immediately（立即应用）。

Amazon CLI

要使用 Amazon CLI 轮换主用户密码密钥，请使用 modify-db-cluster 命令并指定 `--rotate-master-user-password` 选项。轮换主密码时必须指定 `--apply-immediately` 选项。

此示例轮换主用户密码密钥。

Example

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-cluster \
    --db-cluster-identifier mydbcluster \
    --rotate-master-user-password \
```

```
    --apply-immediately
```

对于 Windows：

```
aws rds modify-db-cluster ^
    --db-cluster-identifier mydbcluster ^
    --rotate-master-user-password ^
    --apply-immediately
```

RDS API

您可以使用 ModifyDBCluster 操作并将 `RotateMasterUserPassword` 参数设置为 `true` 来轮换主用户密码密钥。轮换主密码时，必须将 `ApplyImmediately` 参数设置为 `true`。

查看有关数据库实例的密钥的详细信息

您可以使用控制台（https://console.aws.amazon.com/secretsmanager/）或 Amazon CLI（get-secret-value Secrets Manager 命令）检索您的密钥。

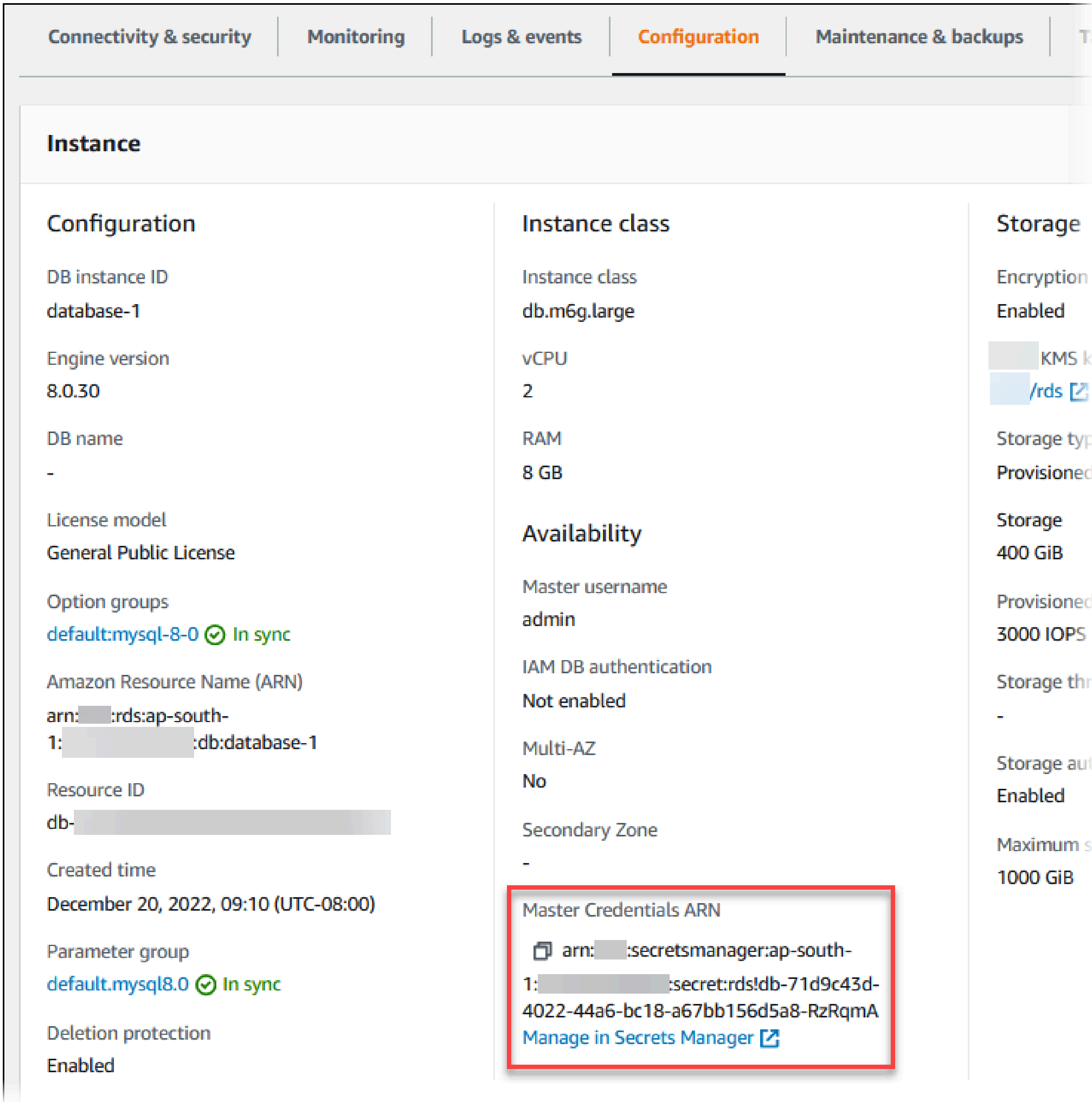
您可以使用 RDS 控制台、Amazon CLI 或 RDS API 找到 RDS 在 Secrets Manager 中管理的密钥的 Amazon 资源名称（ARN）。

控制台

查看有关 RDS 在 Secrets Manager 中管理的密钥的详细信息

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。
3. 选择数据库实例的名称以显示其详细信息。
4. 选择配置选项卡。

   在 Master Credentials ARN（主凭证 ARN）中，您可以查看密钥 ARN。



您可以单击 Manage in Secrets Manager（在 Secrets Manager 中管理）链接，以在 Secrets Manager 控制台中查看和管理密钥。

Amazon CLI

您可以使用 describe-db-instances RDS CLI 命令找到有关 RDS 在 Secrets Manager 中管理的密钥的以下信息：

- SecretArn – 密钥的 ARN
- SecretStatus – 密钥的状态

  可能的状态值包括：

  - creating – 密钥正在创建中。
  - active – 密钥可用于正常使用和轮换。
  - rotating – 密钥正在轮换。
  - impaired – 密钥可用于访问数据库凭证，但不能轮换。例如，如果更改权限以使 RDS 无法再访问密钥或密钥的 KMS 密钥，则密钥可能具有此状态。

    当密钥具有此状态时，您可以更正导致该状态的条件。如果您更正导致状态的条件，则状态会一直保持为 impaired，直至下一次轮换。或者，您可以修改数据库实例以关闭数据库凭证的自动管理，然后再次修改数据库实例以启用数据库凭证的自动管理。要修改数据库实例，请在 modify-db-instance 命令中使用 --manage-master-user-password 选项。
- KmsKeyId – 用于加密密钥的 KMS 密钥的 ARN

指定 --db-instance-identifier 选项可显示特定数据库实例的输出。此示例显示数据库实例使用的密钥的输出。

Example

```
aws rds describe-db-instances --db-instance-identifier mydbinstance
```

以下是密钥的示例输出：

```
"MasterUserSecret": {
                "SecretArn": "arn:aws:secretsmanager:eu-west-1:123456789012:secret:rds!
db-033d7456-2c96-450d-9d48-f5de3025e51c-xmJRDx",
                "SecretStatus": "active",
                "KmsKeyId": "arn:aws:kms:eu-
west-1:123456789012:key/0987dcba-09fe-87dc-65ba-ab0987654321"
            }
```

当您拥有密钥 ARN 时，您可以使用 get-secret-value Secrets Manager CLI 命令查看有关该密钥的详细信息。

此示例显示先前示例输出中的密钥的详细信息。

Example

对于 Linux、macOS 或 Unix：

```
aws secretsmanager get-secret-value \
    --secret-id 'arn:aws:secretsmanager:eu-west-1:123456789012:secret:rds!
db-033d7456-2c96-450d-9d48-f5de3025e51c-xmJRDx'
```

对于 Windows：

```
aws secretsmanager get-secret-value ^
    --secret-id 'arn:aws:secretsmanager:eu-west-1:123456789012:secret:rds!
db-033d7456-2c96-450d-9d48-f5de3025e51c-xmJRDx'
```

RDS API

您可以使用 DescribeDBInstances 操作并将 `DBInstanceIdentifier` 参数设置为数据库实例标识符，查看 RDS 在 Secrets Manager 中管理的密钥的 ARN、状态和 KMS 密钥。输出中包含有关密钥的详细信息。

当您拥有密钥 ARN 时，您可以使用 GetSecretValue Secrets Manager 操作查看有关该密钥的详细信息。

查看有关多可用区数据库集群的密钥的详细信息

您可以使用控制台（https://console.aws.amazon.com/secretsmanager/）或 Amazon CLI（get-secret-value Secrets Manager 命令）检索您的密钥。

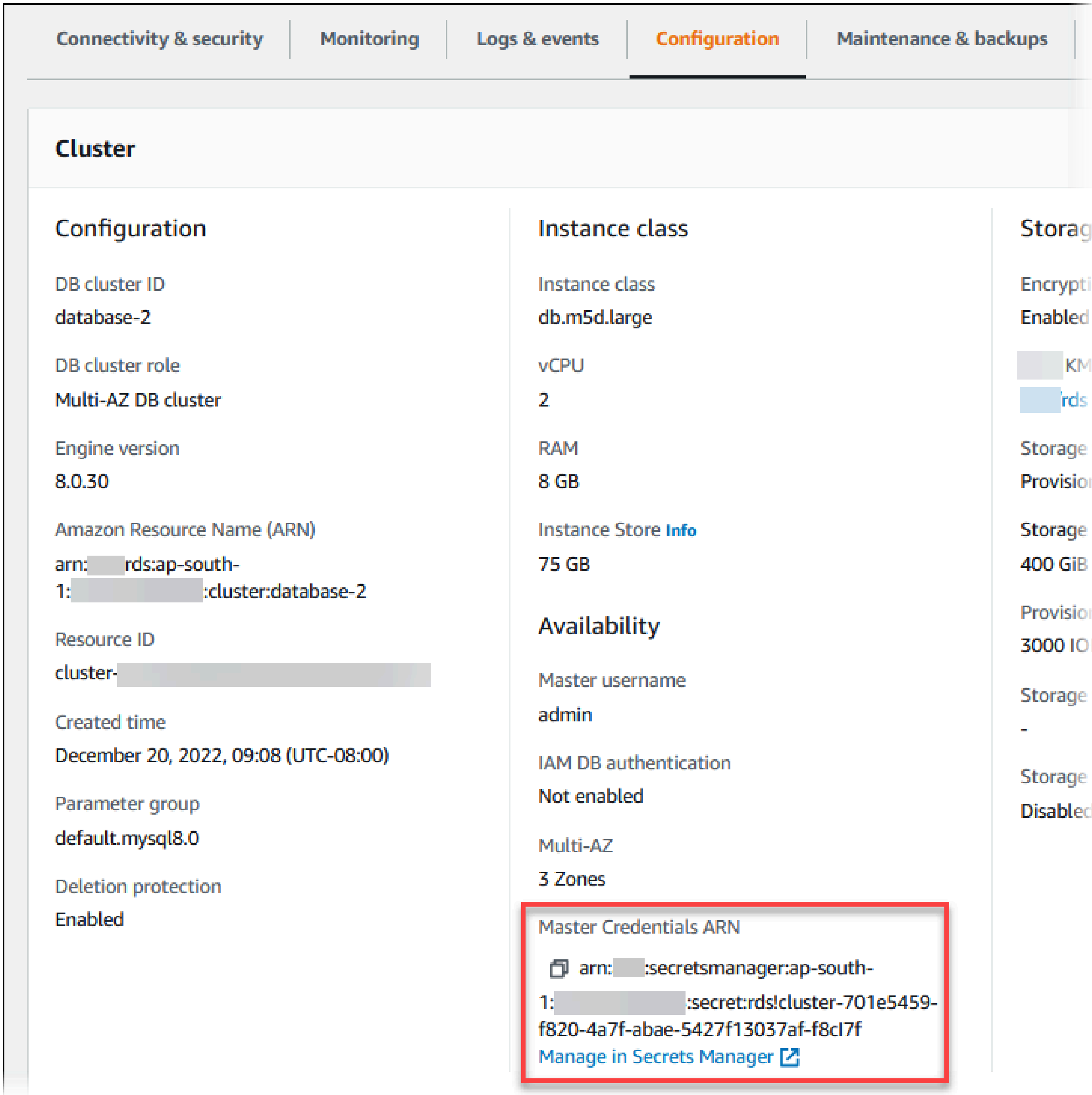
您可以使用 RDS 控制台、Amazon CLI 或 RDS API 找到 RDS 在 Secrets Manager 中管理的密钥的 Amazon 资源名称（ARN）。

控制台

查看有关 RDS 在 Secrets Manager 中管理的密钥的详细信息

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。
3. 选择多可用区数据库集群的名称以显示其详细信息。
4. 选择配置选项卡。

在 Master Credentials ARN（主凭证 ARN）中，您可以查看密钥 ARN。



您可以单击 Manage in Secrets Manager（在 Secrets Manager 中管理）链接，以在 Secrets Manager 控制台中查看和管理密钥。

Amazon CLI

您可以使用 RDS Amazon CLI [describe-db-clusters](https://docs.aws.amazon.com/cli/latest/reference/rds/describe-db-clusters.html) 命令查找有关 RDS 在 Secrets Manager 中管理的密钥的以下信息：

- `SecretArn` – 密钥的 ARN
- `SecretStatus` – 密钥的状态

  可能的状态值包括：

  - `creating` – 密钥正在创建中。
  - `active` – 密钥可用于正常使用和轮换。
  - `rotating` – 密钥正在轮换。
  - `impaired` – 密钥可用于访问数据库凭证，但不能轮换。例如，如果更改权限以使 RDS 无法再访问密钥或密钥的 KMS 密钥，则密钥可能具有此状态。

    当密钥具有此状态时，您可以更正导致该状态的条件。如果您更正导致状态的条件，则状态会一直保持为 `impaired`，直至下一次轮换。或者，您可以修改数据库集群以关闭数据库凭证的自动管理，然后再次修改数据库集群以开启数据库凭证的自动管理。要修改数据库集群，请在 [modify-db-cluster](https://docs.aws.amazon.com/cli/latest/reference/rds/modify-db-cluster.html) 命令中使用 `--manage-master-user-password` 选项。
- `KmsKeyId` – 用于加密密钥的 KMS 密钥的 ARN

指定 `--db-cluster-identifier` 选项可显示特定数据库集群的输出。此示例显示数据库集群使用的密钥的输出。

Example

```
aws rds describe-db-clusters --db-cluster-identifier mydbcluster
```

以下示例显示密钥的输出：

```
"MasterUserSecret": {
                "SecretArn": "arn:aws:secretsmanager:eu-west-1:123456789012:secret:rds!
cluster-033d7456-2c96-450d-9d48-f5de3025e51c-xmJRDx",
                "SecretStatus": "active",
                "KmsKeyId": "arn:aws:kms:eu-
west-1:123456789012:key/0987dcba-09fe-87dc-65ba-ab0987654321"
            }
```

当您拥有密钥 ARN 时，您可以使用 get-secret-value Secrets Manager CLI 命令查看有关该密钥的详细信息。

此示例显示先前示例输出中的密钥的详细信息。

Example

对于 Linux、macOS 或 Unix：

```
aws secretsmanager get-secret-value \
    --secret-id 'arn:aws:secretsmanager:eu-west-1:123456789012:secret:rds!
cluster-033d7456-2c96-450d-9d48-f5de3025e51c-xmJRDx'
```

对于 Windows：

```
aws secretsmanager get-secret-value ^
    --secret-id 'arn:aws:secretsmanager:eu-west-1:123456789012:secret:rds!
cluster-033d7456-2c96-450d-9d48-f5de3025e51c-xmJRDx'
```

RDS API

您可以使用 DescribeDBClusters RDS 操作并将 `DBClusterIdentifier` 参数设置为数据库集群标识符，查看 RDS 在 Secrets Manager 中管理的密钥的 ARN、状态和 KMS 密钥。输出中包含有关密钥的详细信息。

当您拥有密钥 ARN 时，您可以使用 GetSecretValue Secrets Manager 操作查看有关该密钥的详细信息。

查看有关租户数据库的密钥的详细信息

您可以使用控制台（https://console.aws.amazon.com/secretsmanager/）或 Amazon CLI（get-secret-value Secrets Manager 命令）检索您的密钥。

您可以使用 Amazon RDS 控制台、Amazon CLI 或 Amazon RDS API 找到 Amazon RDS 在 Amazon Secrets Manager 中管理的密钥的 Amazon 资源名称（ARN）。

控制台

查看有关 Amazon RDS 在 Amazon Secrets Manager 中管理的租户数据库密钥的详细信息

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。

2. 在导航窗格中，选择 Databases (数据库)。
3. 选择包含租户数据库的数据库实例的名称，即可显示其详细信息。
4. 选择配置选项卡。

   在租户数据库部分，找到租户数据库并查看其主凭证 ARN。

   您可以单击 Manage in Secrets Manager（在 Secrets Manager 中管理）链接，以在 Secrets Manager 控制台中查看和管理密钥。

Amazon CLI

您可以使用 describe-tenant-databases Amazon RDS Amazon CLI 命令找到有关 Amazon RDS 在 Amazon Secrets Manager 中管理的租户数据库密钥的以下信息：

- `SecretArn` – 密钥的 ARN
- `SecretStatus` – 密钥的状态

  可能的状态值包括：

  - `creating` – 密钥正在创建中。
  - `active` – 密钥可用于正常使用和轮换。
  - `rotating` – 密钥正在轮换。
  - `impaired` – 密钥可用于访问数据库凭证，但不能轮换。例如，如果权限发生了更改，使得 Amazon RDS 无法再访问密钥或密钥的 KMS 密钥，则密钥可能具有此状态。

    当密钥具有此状态时，您可以更正导致该状态的条件。如果您更正导致状态的条件，则状态会一直保持为 `impaired`，直至下一次轮换。或者，您可以修改租户数据库以关闭数据库凭证的自动管理，然后再次修改租户数据库以开启数据库凭证的自动管理。要修改租户数据库，请在 modify-tenant-database 命令中使用 `--manage-master-user-password` 选项。
- `KmsKeyId` – 用于加密密钥的 KMS 密钥的 ARN

指定 `--db-instance-identifier` 选项可显示特定数据库实例中的租户数据库输出。您也可以指定 `--tenant-db-name` 选项来显示特定租户数据库的输出。此示例显示租户数据库使用的密钥的输出。

Example

```
aws rds describe-tenant-databases \
    --db-instance-identifier database-3 \
```

```
    --query "TenantDatabases[0].MasterUserSecret"
```

以下是密钥的示例输出：

```
{
    "SecretArn": "arn:aws:secretsmanager:us-east-2:123456789012:secret:rds!db-ABC123",
    "SecretStatus": "active",
    "KmsKeyId": "arn:aws:kms:us-east-2:123456789012:key/aa11bb22-####-####-####-
fedcba123456"
}
```

当您拥有密钥 ARN 时，您可以使用 get-secret-value Secrets Manager Amazon CLI 命令查看有关该密钥的详细信息。

此示例显示先前示例输出中的密钥的详细信息。

Example

对于 Linux、macOS 或 Unix：

```
aws secretsmanager get-secret-value \
    --secret-id 'arn:aws:secretsmanager:us-east-2:123456789012:secret:rds!db-ABC123'
```

对于 Windows：

```
aws secretsmanager get-secret-value ^
    --secret-id 'arn:aws:secretsmanager:us-east-2:123456789012:secret:rds!db-ABC123'
```

Amazon RDS API

您可以使用 DescribeTenantDatabases 操作并将 `DBInstanceIdentifier` 参数设置为数据库实例标识符，查看 Amazon RDS 在 Amazon Secrets Manager 中管理的密钥的 ARN、状态和 KMS 密钥。您也可以将 `TenantDBName` 参数设置为特定的租户数据库名称。输出中包含有关密钥的详细信息。

当您拥有密钥 ARN 时，您可以使用 GetSecretValue Secrets Manager 操作查看有关该密钥的详细信息。

区域和版本可用性

功能可用性和支持因每个数据库引擎的特定版本以及 Amazon Web Services 区域而异。有关 Secrets Manager 与 Amazon RDS 集成的版本和区域可用性的更多信息，请参阅 支持将 Secrets Manager 与 Amazon RDS 进行集成的区域和数据库引擎。

Amazon RDS 中的数据保护

Amazon 责任共担模式适用于 Amazon Relational Database Service 中的数据保护。如该模式中所述，Amazon 负责保护运行所有 Amazon Web Services 云 的全球基础结构。您负责维护对托管在此基础结构上的内容的控制。您还负责您所使用的 Amazon Web Services 服务 的安全配置和管理任务。有关数据隐私的更多信息，请参阅数据隐私常见问题。

出于数据保护目的，建议您保护 Amazon Web Services 账户 凭证并使用 Amazon IAM Identity Center 或 Amazon Identity and Access Management（IAM）设置单个用户。这样，每个用户只获得履行其工作职责所需的权限。还建议您通过以下方式保护数据：

- 对每个账户使用多重身份验证（MFA）。
- 使用 SSL/TLS 与 Amazon 资源进行通信。我们要求使用 TLS 1.2，建议使用 TLS 1.3。
- 使用 Amazon CloudTrail 设置 API 和用户活动日记账记录。有关使用 CloudTrail 跟踪来捕获 Amazon 活动的信息，请参阅《Amazon CloudTrail 用户指南》中的使用 CloudTrail 跟踪。
- 使用 Amazon 加密解决方案以及 Amazon Web Services 服务中的所有默认安全控制。
- 使用高级托管安全服务（例如 Amazon Macie），它有助于发现和保护存储在 Amazon S3 中的敏感数据。
- 如果在通过命令行界面或 API 访问 Amazon 时需要经过 FIPS 140-3 验证的加密模块，请使用 FIPS 端点。有关可用的 FIPS 端点的更多信息，请参阅《美国联邦信息处理标准（FIPS）第 140-3 版》。

强烈建议您切勿将机密信息或敏感信息（如您客户的电子邮件地址）放入标签或自由格式文本字段（如名称字段）。这包括当您通过控制台、API、Amazon CLI 或 Amazon SDK 使用 Amazon RDS 或其他 Amazon Web Services 服务 时。在用于名称的标签或自由格式文本字段中输入的任何数据都可能会用于计费或诊断日志。如果您向外部服务器提供网址，强烈建议您不要在网址中包含凭证信息来验证对该服务器的请求。

主题

- 使用加密保护数据
- 互联网络流量隐私

使用加密保护数据

您可以为数据库资源启用加密。您也可以加密与数据库实例的连接。

主题

加密 Amazon RDS 资源

Amazon RDS 可以加密您的 Amazon RDS 数据库实例。静态加密的数据包括数据库实例的基础存储、其自动化备份、只读副本和快照。

Amazon RDS 加密的数据库实例使用行业标准 AES-256 加密算法来对托管 Amazon RDS 数据库实例的服务器上的数据进行加密。在加密数据后，Amazon RDS 将以透明方式处理访问的身份验证和数据的解密，并且对性能产生的影响最小。您无需修改数据库客户端应用程序来使用加密。

> **Note**
>
> 对于加密和未加密数据库实例，将对在源与只读副本之间发送的数据进行加密，即使在 Amazon 区域之间复制时也是如此。

主题

Amazon RDS 资源加密概览

Amazon RDS 加密的数据库实例通过保护您的数据免受未经授权的访问来为基础存储提供额外一层数据保护。您可以使用 Amazon RDS 加密来增强对云中部署的应用程序的数据保护，并满足静态数据加密的合规性要求。可为 Amazon RDS 加密的数据库实例加密所有日志、备份和快照。有关加密功能的可用性和限制的更多信息，请参阅 Amazon RDS 加密的可用性和 Amazon RDS 加密的数据库实例的限制。

Amazon RDS 使用 Amazon Key Management Service 密钥加密这些资源。Amazon KMS 将安全、高度可用的硬件和软件结合起来，提供面向云端扩展的密钥管理系统。您可以使用 Amazon 托管式密钥，也可以创建客户托管的密钥。

创建加密数据库实例时，您可以为 Amazon RDS 选择客户托管密钥或 Amazon 托管式密钥 来加密您的数据库实例。如果您没有为客户托管密钥指定密钥标识符，则 Amazon RDS 会将 Amazon 托管式密钥 用于您的新数据库实例。Amazon RDS 为您的 Amazon 账户创建用于 Amazon RDS 的 Amazon 托管式密钥。您的 Amazon 账户在每个 Amazon 区域都有用于 Amazon RDS 的不同 Amazon 托管式密钥。

要管理用于加密和解密 Amazon RDS 资源的客户自主管理型密钥，您可以使用 Amazon Key Management Service（Amazon KMS）。

利用 Amazon KMS，您可创建客户自主管理型密钥，并定义策略来控制这些客户自主管理型密钥的使用方式。Amazon KMS 支持 CloudTrail，因此，您可审核 KMS 密钥使用情况以验证客户自主管理型密钥是否使用得当。可以将客户托管密钥与 Amazon Aurora 和支持的 Amazon 服务（例如 Amazon S3、Amazon EBS 和 Amazon Redshift）结合使用。有关与 Amazon KMS 集成的服务的列表，请参阅 Amazon 服务集成。关于使用 KMS 密钥的一些注意事项：

- 创建加密的数据库实例后，您无法更改该数据库实例使用的 KMS 密钥。因此，请确保先确定您的 KMS 密钥要求，然后再创建加密的数据库实例。

  如果您必须更改数据库实例的加密密钥，请创建实例的手动快照，然后在复制快照时启用加密。有关更多信息，请参阅 re:Post 知识文章。
- 如果复制加密快照，则可以使用不同于用于加密源快照的 KMS 密钥来加密目标快照。
- Amazon RDS 加密实例的只读副本必须使用与主数据库实例相同的 KMS 密钥进行加密，前提是两者在相同的 Amazon 区域中。
- 如果主数据库实例和只读副本位于不同的 Amazon 区域中，则使用该 Amazon 区域的 KMS 密钥对只读副本进行加密。
- 如果快照已使用共享该快照的 Amazon 托管式密钥 账户的 Amazon 进行加密，则您无法共享该快照。
- Amazon RDS 还支持利用透明数据加密 (TDE) 对 Oracle 或 SQL Server 数据库实例进行加密。TDE 可以与 RDS 静态加密一起使用，但同时使用 TDE 和 RDS 静态加密可能会稍微影响数据库的性能。您必须为每种加密方法管理不同的密钥。有关 TDE 的更多信息，请参阅 Oracle 透明数据加密 或 SQL Server 中的透明数据加密支持。

> **Important**
>
> 禁用 KMS 密钥后，Amazon RDS 会失去对数据库实例的 KMS 密钥的访问权限。如果您失去了对 KMS 密钥的访问权限，则在启用备份的情况下，加密的数据库实例会在检测后的 2 小时进入 `inaccessible-encryption-credentials-recoverable` 状态。数据库实例将保持此状态七天时间，在此期间实例停止运行。在此期间对该数据库实例进行的 API 调用可能不会成功。要恢复数据库实例，请启用 KMS 密钥并重新启动该数据库实例。从 Amazon Web Services Management Console、Amazon CLI 或 RDS API 启用 KMS 密钥。使用 Amazon CLI 命令 [start-db-instance](https://docs.aws.amazon.com/cli/latest/reference/rds/start-db-instance.html) 或 Amazon Web Services Management Console重新启动数据库实例。
>
> `inaccessible-encryption-credentials-recoverable` 状态仅适用于可以停止的数据库实例。这种可恢复的状态不适用于无法停止的实例，例如只读副本和带有只读副本的实例。有关更多信息，请参阅 停止数据库实例的限制。
>
> 如果数据库实例未在七天内恢复，则该实例会进入终端 `inaccessible-encryption-credentials` 状态。在此状态下，数据库实例将无法再使用，您只能从备份还原数据库实例。强烈建议您始终对加密的数据库实例启用备份以防止数据库中的加密数据丢失。
>
> 在数据库实例创建期间，Amazon RDS 会检查发出调用的主体是否有权访问 KMS 密钥，并从 KMS 密钥中生成授权，该授权将在数据库实例的整个生命周期内使用。撤消发出调用的主体对 KMS 密钥的访问权限，不会影响正在运行的数据库。在跨账户场景中使用 KMS 密钥时，例如将快照复制到其他账户，需要与其他账户共享 KMS 密钥。如果您从快照创建数据库实例但未指定不同的 KMS 密钥，则新实例将使用源账户中的 KMS 密钥。创建数据库实例后，撤消密钥访问权限不会影响实例。但是，禁用某个密钥会影响使用该密钥加密的所有数据库实例。为防止出现这种情况，请在快照复制操作期间指定不同的密钥。
>
> 在实例修改或恢复期间，具有禁用备份的数据库实例将一直可用，直到卷与主机分离。如果适用，RDS 会将实例移至 `inaccessible-encryption-credentials-recoverable` 状态或 `inaccessible-encryption-credentials` 状态。

有关 KMS 密钥的更多信息，请参阅《Amazon Key Management Service 开发人员指南》中的 Amazon KMS keys 和Amazon KMS key 管理。

加密数据库实例

要加密新数据库实例，请在 Amazon RDS 控制台上，选择 Enable encryption（启用加密）。想要了解有关创建数据库实例的信息，请参阅创建 Amazon RDS 数据库实例。

如果使用 create-db-instance Amazon CLI 命令创建加密的数据库实例，请设置 `--storage-encrypted` 参数。如果使用 CreateDBInstance API 操作，请将 `StorageEncrypted` 参数设置为 true。

如果使用 Amazon CLI `create-db-instance` 命令创建带有客户托管密钥的加密数据库实例，请将 `--kms-key-id` 参数设置为 KMS 密钥的任何密钥标识符。如果您使用 Amazon RDS API `CreateDBInstance` 操作，请将 `KmsKeyId` 参数设置为 KMS 密钥的任何密钥标识符。要在其他 Amazon 账户中使用客户托管密钥，请指定密钥 ARN 或别名 ARN。

确定是否为数据库实例启用加密

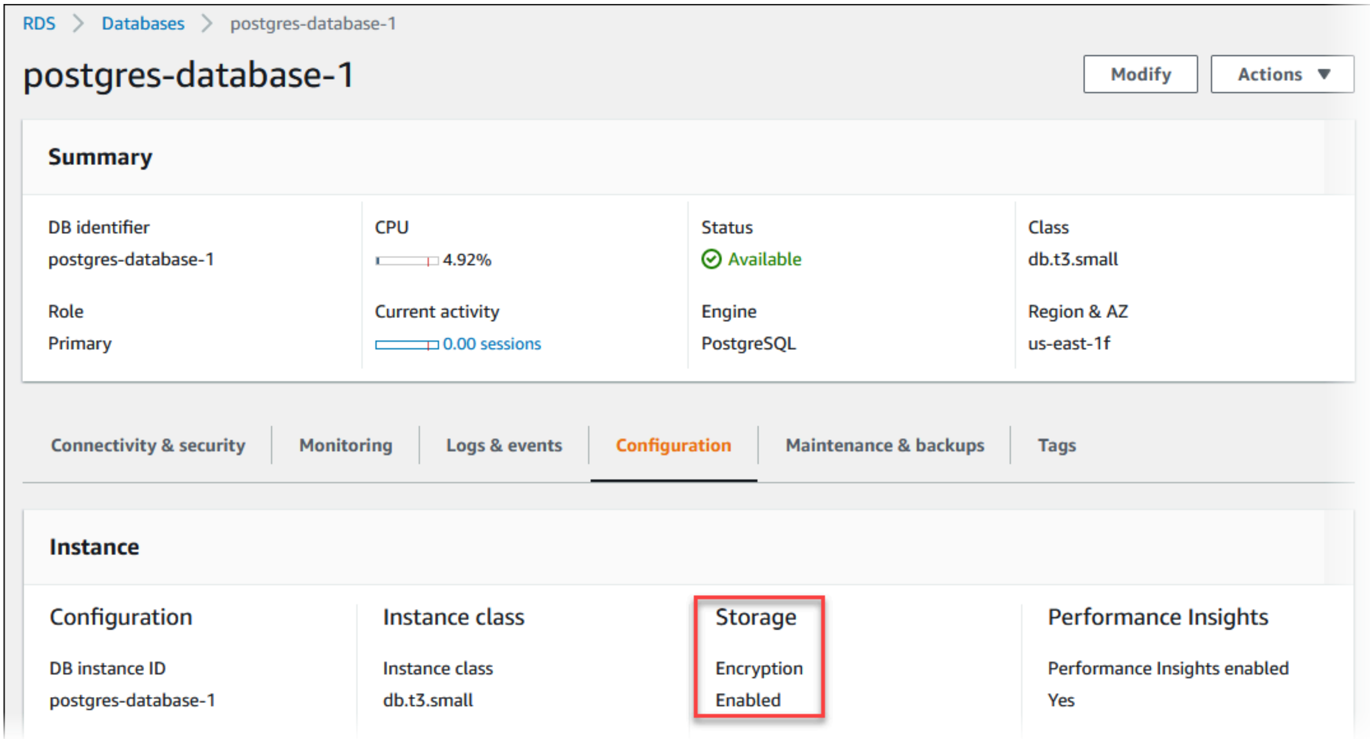
您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 确定是否为数据库实例开启静态加密。

控制台

要确定是否为数据库实例开启静态加密

1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。
3. 选择要检查的数据库实例的名称以显示其详细信息。
4. 选择 Configuration（配置）选项卡，然后检查 Storage（存储）下的 Encryption（加密）值。

   它显示 Enabled（已启用）或 Not enabled（未启用）。



Amazon CLI

要使用 Amazon CLI 确定是否已为数据库实例启用静态加密，请使用以下选项调用 [describe-db-instances](#) 命令：

- `--db-instance-identifier` – 数据库实例的名称。

下面的示例使用查询就 `mydb` 数据库实例的静态加密返回 `TRUE` 或 `FALSE`。

Example

```
aws rds describe-db-instances --db-instance-identifier mydb --query "*[].
{StorageEncrypted:StorageEncrypted}" --output text
```

RDS API

要使用 Amazon RDS API 确定是否已为数据库实例启用静态加密，请使用以下参数调用 [DescribeDBInstances](#) 操作：

- `DBInstanceIdentifier` – 数据库实例的名称。

Amazon RDS 加密的可用性

Amazon RDS 加密当前可用于所有数据库引擎和存储类型。

Amazon RDS 加密适用于大多数数据库实例类。下表列出了不支持 Amazon RDS 加密的数据库实例类：

实例类型	实例类
通用型（M1）	db.m1.small db.m1.medium db.m1.large db.m1.xlarge
内存优化（M2）	db.m2.xlarge db.m2.2xlarge db.m2.4xlarge
突增型（T2）	db.t2.micro

传输中加密

物理层加密

所有在 Amazon 全球网络上跨 Amazon Web Services 区域流动的数据在离开受 Amazon 保护的设施之前，都会在物理层自动加密。可用区之间的所有流量均已加密。其它加密层（包括本节中列出的加密层）可能会提供额外保护。

Amazon VPC 对等连接和 Transit Gateway 跨区域对等连接提供的加密

使用 Amazon VPC 和 Transit Gateway 对等连接的所有跨区域流量在退出区域时都会自动批量加密。所有流量在离开受 Amazon 保护的设施之前，均会在物理层自动提供额外一层加密。

实例之间的加密

Amazon 在所有类型的数据库实例之间提供安全的私有连接。此外，某些实例类型使用底层 Nitro 系统硬件的卸载功能，自动加密实例之间的传输中流量。此加密使用关联数据的身份验证加密

（AEAD）算法，采用 256 位加密。这对网络性能没有影响。要在实例之间支持这种额外的传输中流量加密，必须满足以下要求：

- 使用以下实例类型：
  - 通用型：M6i、M6id、M6in、M6idn、M7g
  - 内存优化型：R6i、R6id、R6in、R6idn、R7g、X2idn、X2iedn、X2iezn
- 这些实例位于同一 Amazon Web Services 区域。
- 这些实例位于相同 VPC 或对等的 VPC 中，并且流量不会通过虚拟网络设备或服务（如负载均衡器或中转网关）传输。

Amazon RDS 加密的数据库实例的限制

Amazon RDS 加密的数据库实例存在以下限制：

- 您只能在创建 Amazon RDS 数据库实例时而不是创建该数据库实例之后加密该数据库实例。

  不过，由于您可以加密未加密快照的副本，因此，您可以高效地为未加密的数据库实例添加加密。也就是说，您可以创建数据库实例快照，然后创建该快照的加密副本。然后，您可以从加密快照还原数据库实例，从而获得原始数据库实例的加密副本。有关更多信息，请参阅 复制 Amazon RDS 的数据库快照。
- 您无法在加密的数据库实例上关闭加密。
- 您无法创建未加密数据库实例的加密快照。
- 加密数据库实例的快照必须使用与数据库实例相同的 KMS 密钥进行加密。
- 您无法拥有未加密数据库实例的加密只读副本或加密数据库实例的未加密只读副本。
- 如果加密的只读副本位于同一 Amazon 区域，则它们的 KMS 密钥必须与源数据库实例的相同。
- 您不能将未加密的备份或快照还原到加密的数据库实例。
- 要将已加密快照从一个 Amazon 区域复制到另一个区域，您必须指定目标 Amazon 区域的 KMS 密钥。这是因为 KMS 密钥特定于在其中创建它们的 Amazon 区域。

  源快照在复制过程中保持加密状态。Amazon RDS 使用信封加密在复制过程中保护数据。有关信封加密的更多信息，请参阅Amazon Key Management Service开发人员指南中的信封加密。
- 您无法对加密数据库实例取消加密。但是，您可以从加密的数据库实例中导出数据，然后将数据导入未加密的数据库实例。

Amazon KMS key 管理

Amazon RDS 自动与 Amazon Key Management Service（Amazon KMS）集成以进行密钥管理。Amazon RDS 使用信封加密。有关信封加密的更多信息，请参阅 Amazon Key Management Service 开发人员指南中的信封加密。

您可以使用两种类型的 Amazon KMS 密钥来加密数据库实例。

- 要完全控制 KMS 密钥，您必须创建客户托管密钥。有关客户托管密钥的更多信息，请参阅 Amazon Key Management Service 开发人员指南中的客户托管密钥。
- Amazon 托管式密钥 是由与 Amazon KMS 集成的 Amazon 服务代表您在账户中创建、管理和使用的 KMS 密钥。默认情况下，RDS Amazon 托管式密钥（`aws/rds`）用于加密。您无法管理、轮换或删除 RDS Amazon 托管式密钥。有关 Amazon 托管式密钥 的更多信息，请参阅《Amazon Key Management Service 开发人员指南》中的 Amazon 托管式密钥。

要管理用于 Amazon RDS 加密数据库实例的 KMS 密钥，您可以使用 Amazon KMS 控制台中的 Amazon Key Management Service（Amazon KMS）、Amazon CLI 或 Amazon KMS API。要查看利用 Amazon 托管式或客户托管密钥执行的每个操作的审计日志，请使用 Amazon CloudTrail。有关密钥轮换的更多信息，请参阅轮换 Amazon KMS 密钥。

授权使用客户托管密钥

RDS 在加密操作中使用客户自主管理型密钥时，其代表创建或更改 RDS 资源的用户执行操作。

要使用客户自主管理型密钥创建 RDS 资源，用户必须具有对客户自主管理型密钥调用以下操作的权限：

- `kms:CreateGrant`
- `kms:DescribeKey`

您可以在密钥策略中指定这些所需权限，或者在密钥策略允许的情况下在 IAM 策略中指定这些权限。

> **Tip**
> 为遵循最小特权原则，请不要允许对 `kms:CreateGrant` 拥有完全访问权限。而是仅当 Amazon 服务代表用户创建授权时，才使用 kms:ViaService 条件密钥来支持用户对 KMS 密钥创建授权。

您可以通过各种方式使 IAM 策略更严格。例如，如果您要支持将客户自主管理型密钥仅用于源自 RDS 的请求，请将 kms:ViaService 条件密钥与 `rds.<region>.amazonaws.com` 值结合使用。此外，您可以使用Amazon RDS 加密上下文中的密钥或值作为使用客户托管密钥进行加密的条件。

有关更多信息，请参阅《Amazon Key Management Service 开发人员指南》中的允许其他账户中的用户使用 KMS 密钥和 Amazon KMS 中的密钥策略。

Amazon RDS 加密上下文

当 RDS 使用您的 KMS 密钥时，或者当 Amazon EBS 代表 RDS 使用 KMS 密钥时，服务会指定加密上下文。加密上下文是 Amazon KMS 用于确保数据完整性而使用的额外的身份验证数据 (AAD)。在为加密操作指定加密上下文时，该服务必须为解密操作指定同一加密上下文。否则，解密将失败。加密上下文还将写至您的 Amazon CloudTrail 日志中，以帮助您了解为什么使用给定的 KMS 密钥。您的 CloudTrail 日志可能包含多个描述 KMS 密钥使用情况的条目，但每个日志条目中的加密上下文可以帮助您确定该特定使用的原因。

至少，Amazon RDS 始终将数据库实例 ID 用于加密上下文，如以下 JSON 格式的示例所示：

```
{ "aws:rds:db-id": "db-CQYSMDPBRZ7BPMH7Y3RTDG5QY" }
```

此加密上下文可以帮助您确定使用您的 KMS 密钥的数据库实例。

当您的 KMS 密钥用于特定的数据库实例和特定的 Amazon EBS 卷时，数据库实例 ID 和 Amazon EBS 卷 ID 将用于加密上下文，如以下 JSON 格式的示例所示：

```
{
  "aws:rds:db-id": "db-BRG7VYS3SVIFQW7234EJQOM5RQ",
  "aws:ebs:id": "vol-ad8c6542"
}
```

使用 SSL/TLS 加密与数据库实例或集群的连接

您可以使用应用程序中的安全套接字层（SSL）或传输层安全性协议（TLS）来加密与运行 Db2、MariaDB、Microsoft SQL Server、MySQL、Oracle 或 PostgreSQL 的数据库的连接。

SSL/TLS 连接对在客户端和数据库实例或集群之间移动的数据进行加密，从而提供一层额外的安全性。（可选）您的 SSL/TLS 连接可以通过验证安装在数据库上的服务器证书来执行服务器身份验证。要请求服务器身份验证，请遵循以下一般流程：

1. 为数据库选择对数据库服务器证书进行签名的证书颁发机构（CA）。有关证书颁发机构的更多信息，请参阅证书颁发机构。

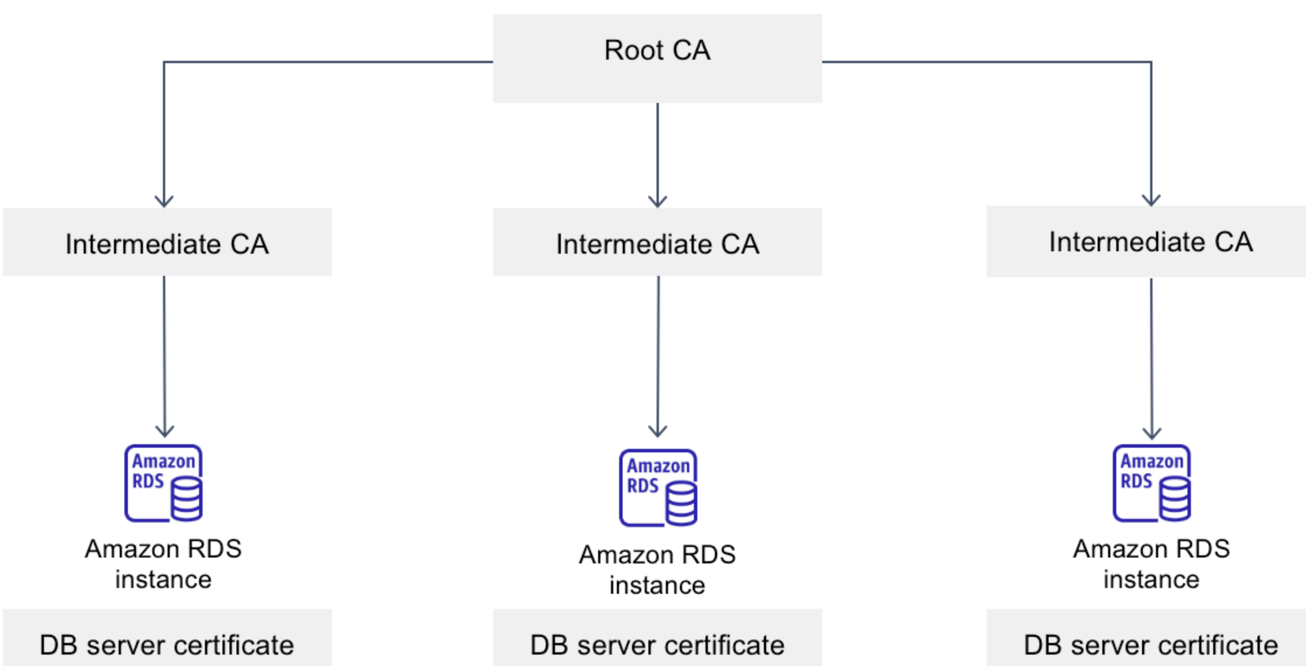
2. 下载证书包，以便在连接到数据库时使用。要下载证书捆绑包，请参阅。特定 Amazon Web Services 区域的证书捆绑包。。

> **Note**
>
> 只能使用 SSL/TLS 连接下载所有证书。

3. 使用数据库引擎用于实现 SSL/TLS 连接的过程连接到数据库。每个数据库引擎都有自己的用于实施 SSL/TLS 的过程。要了解如何为您的数据库实施 SSL/TLS，请使用对应于您的数据库引擎的链接：
   - 对 Amazon RDS for Db2 数据库实例使用 SSL/TLS
   - Amazon RDS 上 MariaDB 数据库实例的 SSL/TLS 支持
   - 将 SSL 与 Microsoft SQL Server 数据库实例结合使用
   - Amazon RDS 上 MySQL 数据库实例的 SSL/TLS 支持
   - 对 RDS for Oracle 数据库实例使用 SSL
   - 将 SSL 与 PostgreSQL 数据库实例结合使用

证书颁发机构

证书颁发机构（CA）是用于标识证书链顶部的根 CA 的证书。CA 签署安装在每个数据库实例上的数据库服务器证书。数据库服务器证书将数据库实例标识为可信服务器。



Amazon RDS 提供以下 CA 来签署数据库的数据库服务器证书。

证书颁发机构（CA）	描述	公用名（CN）
rds-ca-rsa2048-g1	在大多数 Amazon Web Services 区域中，使用具有 RSA 2048 私有密钥算法和 SHA256 签名算法的证书颁发机构。 在 Amazon GovCloud (US) Regions中，此 CA 使用具有 RSA 2048 私有密钥算法和 SHA384 签名算法的证书颁发机构。 此 CA 支持服务器证书自动轮换。	Amazon RDS *region-identifier* RSA2048 G1

> **Note**
>
> 如果您使用的是 Amazon CLI，则可以使用 describe-certificates 查看上面列出的证书颁发机构的有效性。

这些 CA 证书包含在区域和全球证书捆绑包中。当您将 rds-ca-rsa2048-g1 CA 用于数据库时，RDS 会管理数据库上的数据库服务器证书。RDS 会在数据库服务器证书过期之前自动轮换此证书。

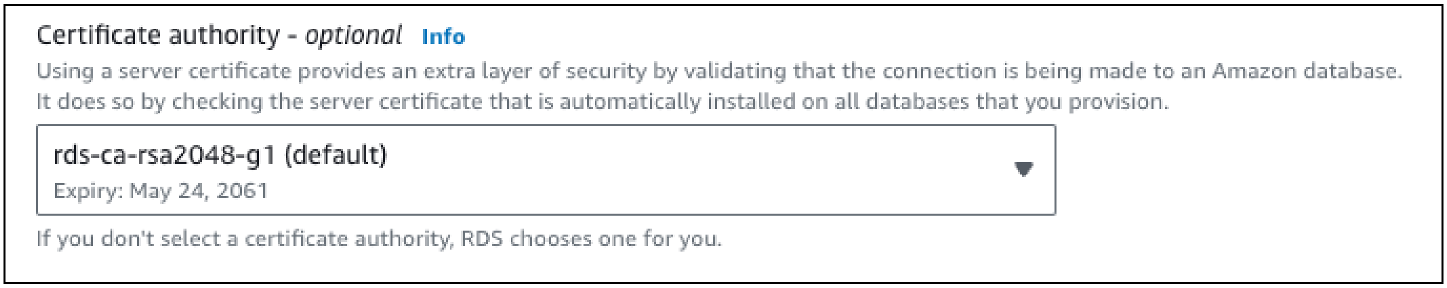
为数据库设置 CA

您可以在执行以下任务时为数据库设置 CA：

- 创建数据库实例或多可用区数据库集群 – 您可以在创建数据库实例或集群时设置 CA。有关说明，请参阅the section called “创建数据库实例”或the section called “创建多可用区数据库集群”。
- 修改数据库实例或多可用区数据库集群 - 您可以通过修改数据库实例或集群为其设置 CA。有关说明，请参阅the section called “修改数据库实例”或the section called “修改多可用区数据库集群”。

> **Note**
>
> 默认 CA 设置为 rds-ca-rsa2048-g1。您可以使用 modify-certificates 命令为您的 Amazon Web Services 账户覆盖默认 CA。

可用的 CA 取决于数据库引擎和数据库引擎版本。使用 Amazon Web Services Management Console 时，您可以使用 Certificate authority（证书颁发机构）设置选择 CA，如下图所示。



控制台仅显示可用于数据库引擎和数据库引擎版本的 CA。如果您使用的是 Amazon CLI，则可以使用 create-db-instance 或 modify-db-instance 命令为数据库实例设置 CA。您可以使用 create-db-cluster 或 modify-db-cluster 命令为多可用区数据库集群设置 CA。

如果您使用的是 Amazon CLI，则可以使用 describe-certificates 命令查看您账户的可用 CA。此命令还在输出的 `ValidTill` 中显示每个 CA 的到期日期。您可以使用 describe-db-engine-versions 命令找到适用于特定数据库引擎和数据库引擎版本的 CA。

以下示例显示了可用于默认 RDS for PostgreSQL 数据库引擎版本的 CA。

```
aws rds describe-db-engine-versions --default-only --engine postgres
```

输出类似于以下内容。`SupportedCACertificateIdentifiers` 中列出了可用的 CA。输出还显示数据库引擎版本是否支持在 `SupportsCertificateRotationWithoutRestart` 中不重新启动的情况下轮换证书。

```
{
    "DBEngineVersions": [
        {
            "Engine": "postgres",
            "MajorEngineVersion": "13",
            "EngineVersion": "13.4",
            "DBParameterGroupFamily": "postgres13",
            "DBEngineDescription": "PostgreSQL",
            "DBEngineVersionDescription": "PostgreSQL 13.4-R1",
            "ValidUpgradeTarget": [],
            "SupportsLogExportsToCloudwatchLogs": false,
            "SupportsReadReplica": true,
            "SupportedFeatureNames": [
                "Lambda"
            ],
```

```
                "Status": "available",
                "SupportsParallelQuery": false,
                "SupportsGlobalDatabases": false,
                "SupportsBabelfish": false,
                "SupportsCertificateRotationWithoutRestart": true,
                "SupportedCACertificateIdentifiers": [
                    "rds-ca-rsa2048-g1",
                    "rds-ca-ecc384-g1",
                    "rds-ca-rsa4096-g1"
                ]
            }
        ]
}
```

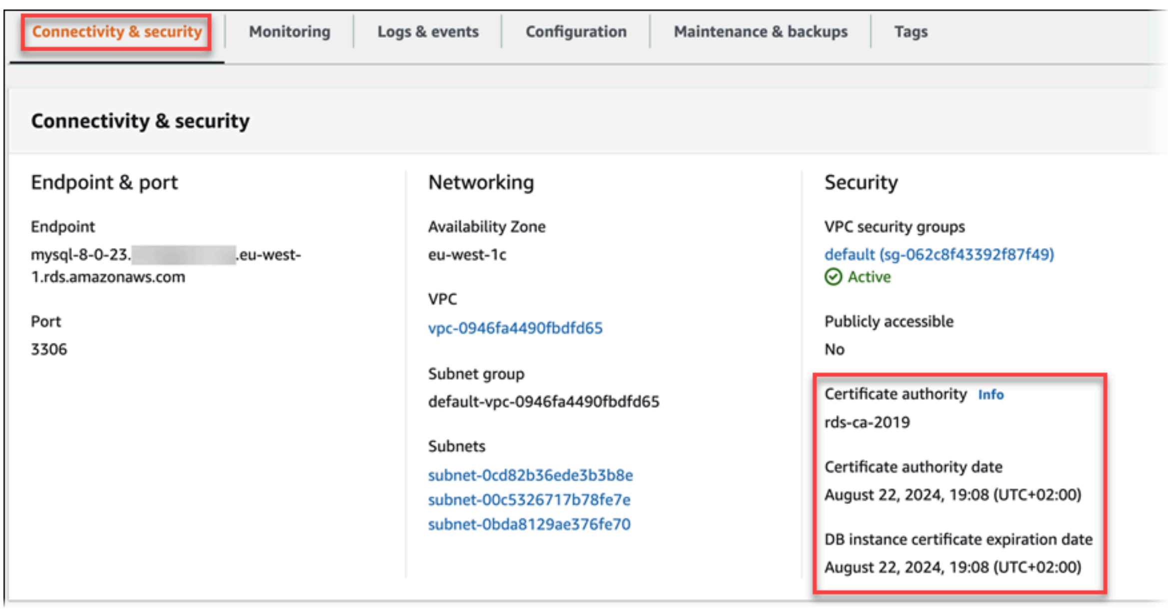
数据库服务器证书有效期

数据库服务器证书的有效期取决于数据库引擎和数据库引擎版本。如果数据库引擎版本支持在不重启的情况下轮换证书，则数据库服务器证书的有效期为 1 年。否则，有效期为 3 年。

有关数据库服务器证书轮换的更多信息，请参阅[自动服务器证书轮换](#)。

查看数据库实例的 CA

您可以通过查看控制台中的连接性和安全性选项卡来查看有关数据库的 CA 的详细信息，如下图所示。



如果您使用的是 Amazon CLI，您可以使用 [describe-db-instances](#) 命令，查看有关数据库实例的 CA 的详细信息。您可以使用 [describe-db-clusters](#) 命令查看有关多可用区数据库集群的 CA 的详细信息。

下载 Amazon RDS 的证书捆绑包

当您使用 SSL 或 TLS 连接到数据库时，数据库实例需要来自 Amazon RDS 的信任证书。在下表中选择相应的链接，来下载与托管数据库的 Amazon Web Services 区域相对应的捆绑包。

特定 Amazon Web Services 区域的证书捆绑包

要获取中国（北京）Amazon 区域或中国（宁夏）Amazon 区域的证书捆绑包，可从 https://rds-truststore.s3.cn-north-1.amazonaws.com.cn/global/global-bundle.pem 下载。如果应用程序位于 Microsoft Windows 中并需要 PKCS7 文件，可以从 https://rds-truststore.s3.cn-north-1.amazonaws.com.cn/global/global-bundle.p7b 下载 PKCS7 证书捆绑包。

该捆绑包包含 `rds-ca-rsa2048-g1`、`rds-ca-rsa4096-g1` 和 `rds-ca-ecc384-g1` 根 CA 证书。您的应用程序信任存储只需要注册根 CA 证书。

要获取证书捆绑包，可通过下表中与 Amazon 区域对应的链接下载。

Amazon 区域	证书捆绑包 (PEM)	证书捆绑包 (PKCS7)
China (Beijing)	cn-north-1-bundle.pem	cn-north-1-bundle.p7b
China (Ningxia)	cn-northwest-1-bundle.pem	cn-northwest-1-bundle.p7b

查看 CA 证书的内容

要检查您的 CA 证书包的内容，请使用以下命令：

```
keytool -printcert -v -file global-bundle.pem
```

轮换 SSL/TLS 证书

Amazon RDS 证书颁发机构证书 rds-ca-2019 于 2024 年 8 月到期。如果您使用或计划使用带有证书验证的安全套接字层（SSL）或传输层安全性协议（TLS）来连接您的 RDS 数据库实例或多可用区数据库集群，则考虑使用新的 CA 证书 rds-ca-rsa2048-g1。如果您当前未将 SSL/TLS 用于证书验证，则可能仍有过期的 CA 证书，如果您计划为证书验证使用 SSL/TLS 连接到 RDS 数据库，则必须将它们更新为新的 CA 证书。

Amazon RDS 提供新 CA 证书作为Amazon最佳安全实践。有关新证书和受支持的 Amazon 区域的信息，请参阅使用 SSL/TLS 加密与数据库实例或集群的连接。

要更新数据库的 CA 证书，请使用以下方法：

- 通过修改数据库实例或集群来更新 CA 证书
- 通过应用维护来更新 CA 证书

更新数据库实例或多可用区数据库集群来使用新的 CA 证书之前，请确保更新连接到 RDS 数据库的客户端或应用程序。

轮换证书的注意事项

在轮换证书之前，请考虑以下情况：

- Amazon RDS 代理使用来自 Amazon Certificate Manager（ACM）的证书。如果您使用的是 RDS 代理，当您轮换 SSL/TLS 证书时，您不需要更新使用 RDS 代理连接的应用程序。有关更多信息，请参阅 将 TLS/SSL 与 RDS 代理结合使用。
- 如果您将 Go 版本 1.15 应用程序与在 2020 年 7 月 28 日之前创建或更新到 rds-ca-2019 证书的数据库实例或多可用区数据库集群一起使用，则必须再次更新证书。将证书更新为 rds-ca-rsa2048-g1。

  使用新的 CA 证书标识符，对数据库实例运行 `modify-db-instance` 命令，或对多可用区数据库集群运行 `modify-db-cluster` 命令。您可以使用 `describe-db-engine-versions` 命令找到适用于特定数据库引擎和数据库引擎版本的 CA。

  如果您在 2020 年 7 月 28 日之后创建了数据库或更新了其证书，则无需执行任何操作。有关更多信息，请参阅 Go GitHub 问题 #39568。

通过修改数据库实例或集群来更新 CA 证书

以下示例将您的 CA 证书从 rds-ca-2019 更新为 rds-ca-rsa2048-g1。。

更新您的应用程序信任存储，以减少任何与更新 CA 证书相关的停机时间。有关与 CA 证书轮换相关的重启的更多信息，请参阅自动服务器证书轮换。

通过修改数据库实例或集群来更新 CA 证书

1. 下载新的 SSL/TLS 证书，如使用 SSL/TLS 加密与数据库实例或集群的连接中所述。
2. 更新应用程序以使用新的 SSL/TLS 证书。

   更新应用程序以使用新 SSL/TLS 证书的方法取决于特定的应用程序。请与应用程序开发人员一起更新应用程序的 SSL/TLS 证书。

   有关检查 SSL/TLS 连接和更新每个数据库引擎的应用程序的信息，请参阅以下主题：

- 更新应用程序以使用新的 SSL/TLS 证书连接到 MariaDB 实例
- 更新应用程序以使用新的 SSL/TLS 证书连接到 Microsoft SQL Server 数据库实例
- 更新应用程序以使用新的 SSL/TLS 证书连接到 MySQL 数据库实例
- 更新应用程序以使用新的 SSL/TLS 证书连接到 Oracle 数据库实例
- 更新应用程序以使用新的 SSL/TLS 证书连接到 PostgreSQL 数据库实例

有关更新 Linux 操作系统信任存储的示例脚本，请参阅将证书导入信任存储的示例脚本。

> **Note**
>
> 证书捆绑包包含新旧 CA 证书，因此您可以安全地升级应用程序并在转换期间保持连接。如果您正在使用 Amazon Database Migration Service 将数据库迁移到数据库实例或集群，我们建议您使用证书捆绑包来确保迁移期间的连接性。

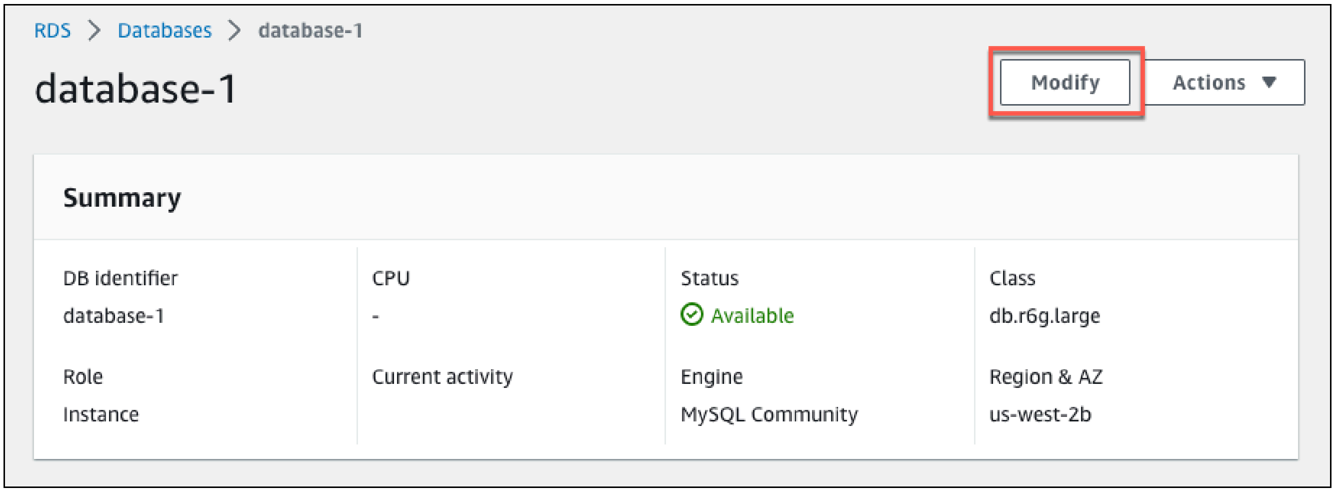
3. 修改数据库实例或多可用区数据库集群，以便将 CA 从 rds-ca-2019 更改为 rds-ca-rsa2048-g1。要检查您的数据库是否需要重启才能更新 CA 证书，请使用 describe-db-engine-versions 命令并检查 `SupportsCertificateRotationWithoutRestart` 标志。

> **Important**
>
> 如果您在证书到期后遇到连接问题，请通过在控制台中指定 Apply immediately (立即应用) 或者 使用 `--apply-immediately` 指定 Amazon CLI 选项来使用“立即应用”选项。默认情况下，此操作安排在您的下个维护时段运行。
> 对于 RDS for Oracle 数据库实例，建议您重新启动 Oracle 数据库以防止出现任何连接错误。
> 对于启用了 AlwaysOn 或镜像选项的 RDS for SQL Server 多可用区实例，在证书轮换后重启实例时，预计会进行失效转移。
> 要为与默认 RDS CA 不同的实例 CA 设置覆盖，请使用 modify-certificates CLI 命令。

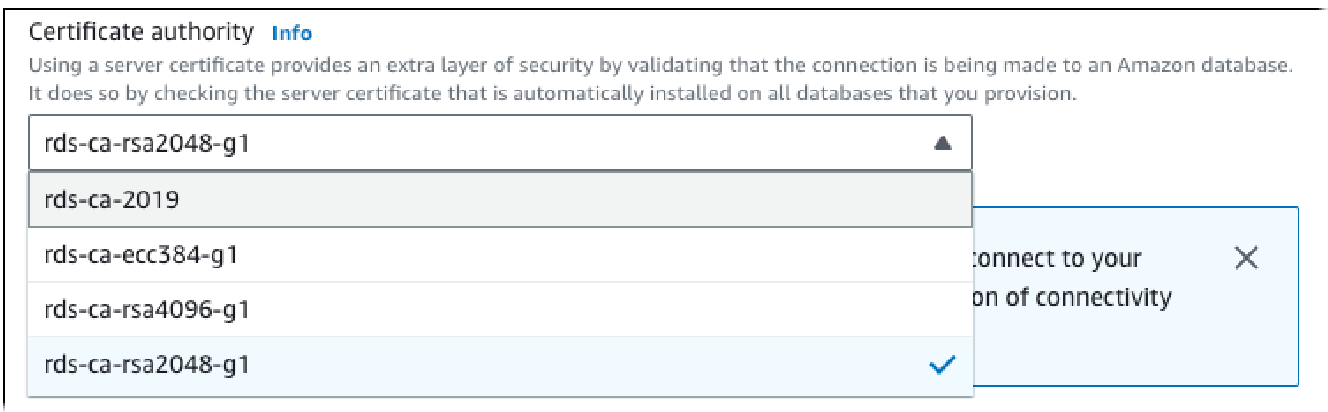
您可以使用 Amazon Web Services Management Console或 Amazon CLI，对于数据库实例或多可用区数据库集群，将 CA 证书从 rds-ca-2019 更改为 rds-ca-rsa2048-g1。

Console

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择数据库，然后选择要修改的数据库实例或多可用区数据库集群。
3. 选择 Modify(修改)。



4. 在连接部分中，选择 rds-ca-rsa2048-g1。



5. 选择继续，查看修改摘要。
6. 要立即应用更改，请选择立即应用。
7. 在确认页面上，检查您的更改。如果更改正确无误，请选择修改数据库实例或修改集群以保存更改。

> **Important**
>
> 安排此操作时，请确保已预先更新客户端信任存储。

也可以选择 Back 编辑您的更改，或选择 Cancel 取消更改。

Amazon CLI

要使用 Amazon CLI 将数据库实例或多可用区数据库集群的 CA 从 rds-ca-2019 更改为 rds-ca-rsa2048-g1，请调用 modify-db-instance 或 modify-db-cluster 命令。指定数据库实例或集群标识符和 `--ca-certificate-identifier` 选项。

使用 `--apply-immediately` 参数可立即应用更新。默认情况下，此操作安排在您的下个维护时段运行。

> **Important**
>
> 安排此操作时，请确保已预先更新客户端信任存储。

Example

数据库实例

以下示例通过将 CA 证书设置为 `rds-ca-rsa2048-g1` 来修改 `mydbinstance`。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier mydbinstance \
    --ca-certificate-identifier rds-ca-rsa2048-g1
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier mydbinstance ^
    --ca-certificate-identifier rds-ca-rsa2048-g1
```

> **Note**
>
> 如果您的实例需要重启，可以使用 modify-db-instance CLI 命令并指定 `--no-certificate-rotation-restart` 选项。

Example

多可用区数据库集群

以下示例通过将 CA 证书设置为 `rds-ca-rsa2048-g1` 来修改 `mydbcluster`。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-cluster \
    --db-cluster-identifier mydbcluster \
    --ca-certificate-identifier rds-ca-rsa2048-g1
```

对于 Windows：

```
aws rds modify-db-cluster ^
    --db-cluster-identifier mydbcluster ^
    --ca-certificate-identifier rds-ca-rsa2048-g1
```

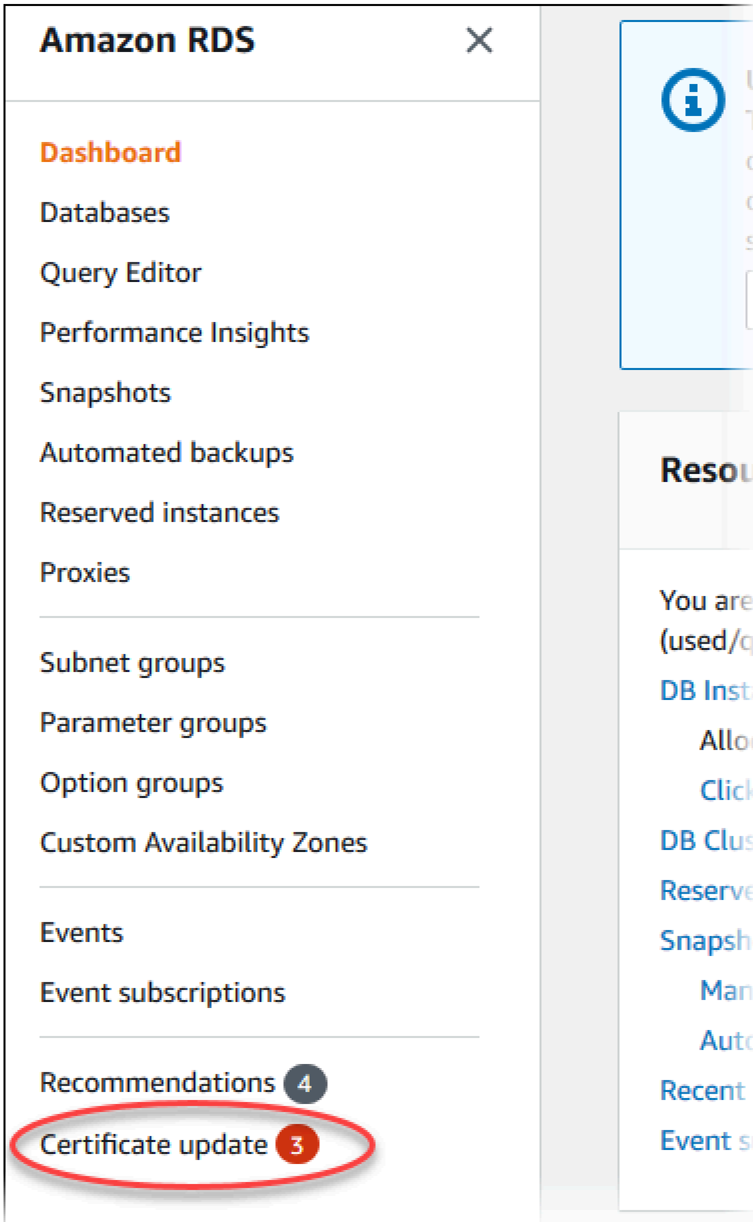
通过应用维护来更新 CA 证书

完成以下步骤，通过应用维护来更新 CA 证书。

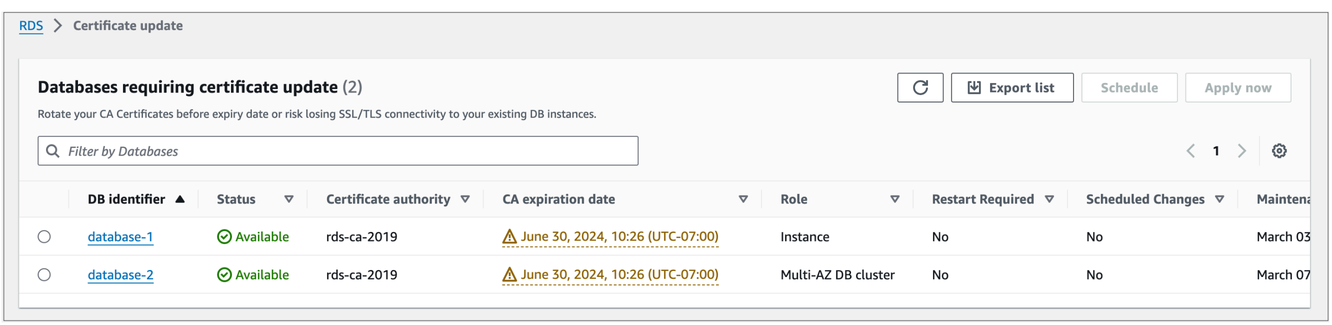
Console

通过应用维护来更新 CA 证书

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择证书更新。



将显示需要更新证书的数据库页面。



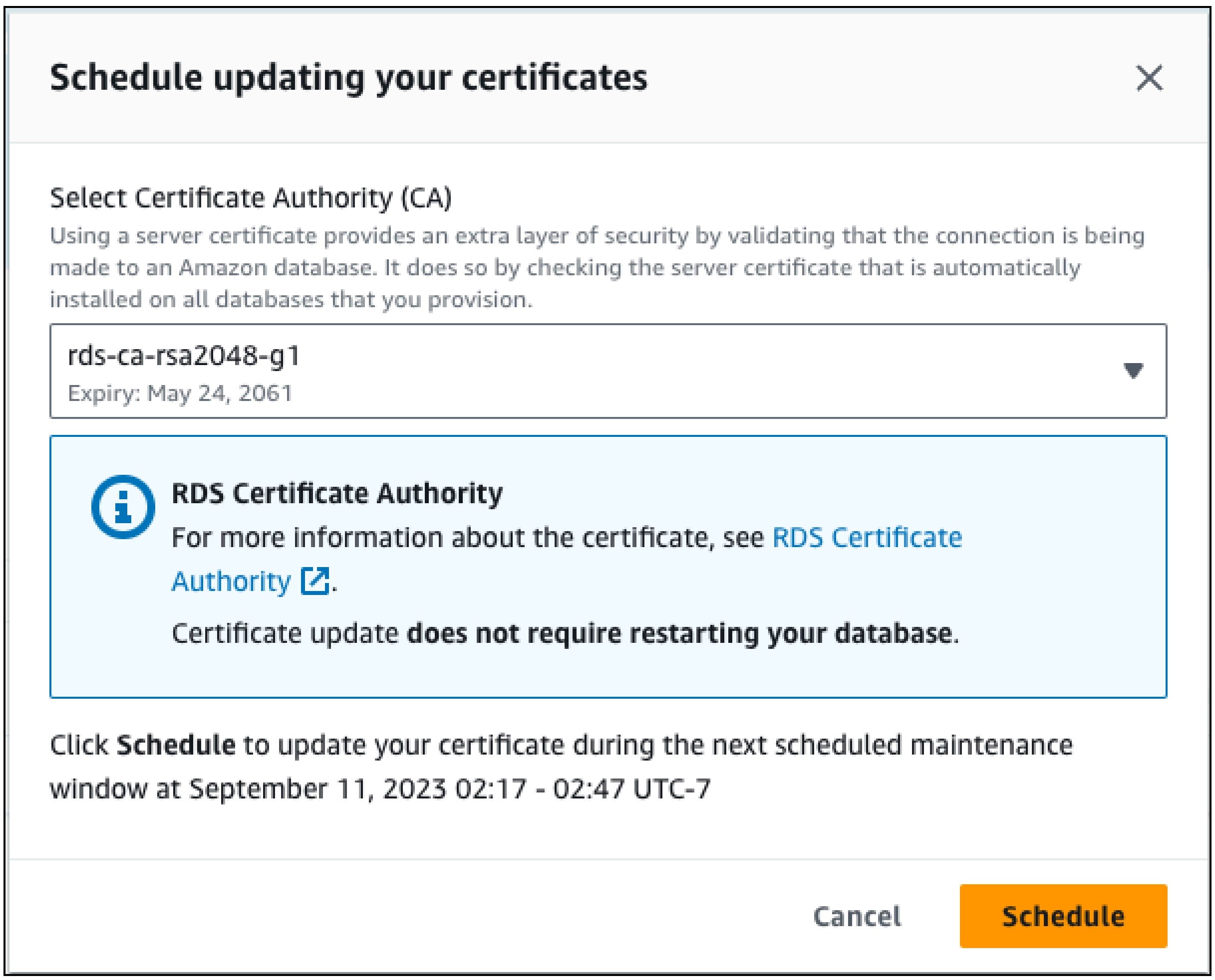
> **Note**
>
> 该页面仅显示当前 Amazon Web Services 区域的数据库实例和集群。如果您在多个 Amazon Web Services 区域中有数据库，请检查每个 Amazon Web Services 区域中的该页面，以查看具有旧 SSL/TLS 证书的所有数据库实例。

3. 选择要更新的数据库实例或多可用区数据库集群。

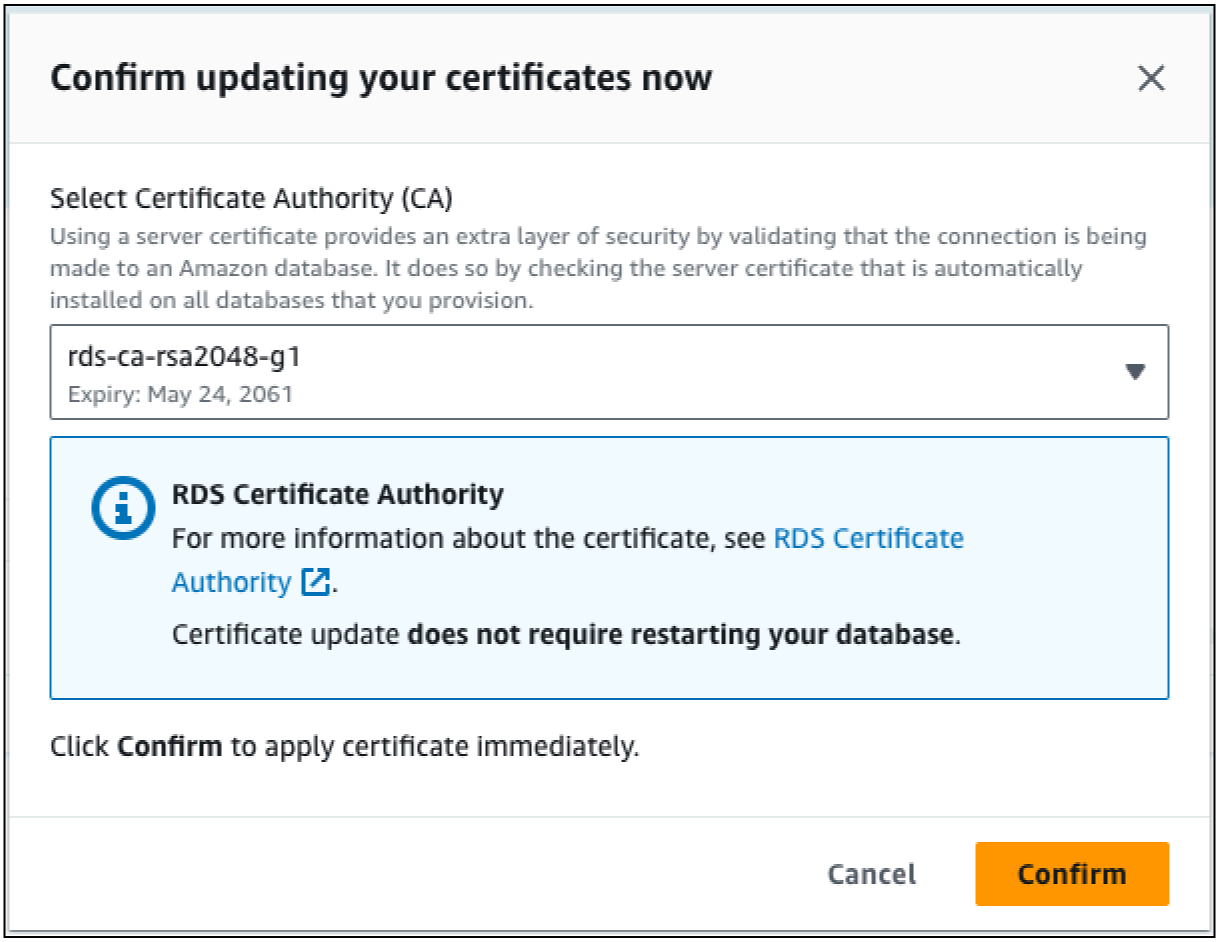
   您可以通过选择计划来计划下一维护时段的证书轮换。通过选择立即应用来立即应用轮换。

> **Important**
>
> 如果您在证书到期后遇到连接问题，请使用立即应用选项。

4. a. 如果您选择计划，系统会提示您确认 CA 证书轮换。此提示还说明更新的计划时段。



   b. 如果您选择立即应用，系统会提示您确认 CA 证书轮换。



> **Important**
>
> 在计划数据库上的 CA 证书轮换之前，请更新使用 SSL/TLS 和服务器证书进行连接的所有客户端应用程序。这些更新特定于您的数据库引擎。更新这些客户端应用程序后，可以确认 CA 证书轮换。

要继续，请选中该复选框，然后选择 Confirm (确认)。

5. 对要更新的每个数据库实例和集群重复步骤 3 和 4。

自动服务器证书轮换

如果根 CA 支持自动服务器证书轮换，RDS 会自动处理数据库服务器证书的轮换。RDS 使用相同的根 CA 进行自动轮换，因此您无需下载新的 CA 服务包。请参阅证书颁发机构。

数据库服务器证书的轮换和有效期取决于您的数据库引擎：

- 如果您的数据库引擎支持无需重启即可轮换，则 RDS 会自动轮换数据库服务器证书，而无需您执行任何操作。RDS 尝试在您首选的维护时段中，在数据库服务器证书的半生命周期轮换您的数据库服务器证书。新的数据库服务器证书的有效期为 12 个月。
- 如果您的数据库引擎不支持无需重启即可轮换，则 RDS 会在数据库服务器证书到期前至少 6 个月通知您有关维护事件的信息。新的数据库服务器证书的有效期为 36 个月。

使用 describe-db-engine-versions 命令并检查 `SupportsCertificateRotationWithoutRestart`标志，以确定数据库引擎版本是否支持无需重启即可轮换证书。有关更多信息，请参阅 为数据库设置 CA。

将证书导入信任存储的示例脚本

以下是将证书捆绑包导入信任存储的示例 shell 脚本。

每个示例 Shell 脚本都使用 keytool，它是 Java 开发工具包 (JDK) 的一部分。有关安装 JDK 的信息，请参阅 JDK 安装指南。

Linux

下面是一个示例 Shell 脚本，它将证书捆绑包导入 Linux 操作系统上的信任存储。

```
mydir=tmp/certs
if [ ! -e "${mydir}" ]
then
mkdir -p "${mydir}"
fi truststore=${mydir}/rds-truststore.jks storepassword=changeit

curl -sS "https://rds-truststore.s3.cn-north-1.amazonaws.com.cn/global/global-bundle.pem"> ${mydir}/global-bundle.pem
awk 'split_after == 1 {n++;split_after=0} /-----END CERTIFICATE-----/
 {split_after=1}{print > "rds-ca-" n+1 ".pem"}' < ${mydir}/global-bundle.pem

for CERT in rds-ca-*; do alias=$(openssl x509 -noout -text -in $CERT | perl -ne
 'next unless /Subject:/; s/.*(CN=|CN = )//; print')
  echo "Importing $alias"
  keytool -import -file ${CERT} -alias "${alias}" -storepass ${storepassword} -keystore ${truststore} -noprompt
  rm $CERT
done

rm ${mydir}/global-bundle.pem
```

```
echo "Trust store content is: "

keytool -list -v -keystore "$truststore" -storepass ${storepassword} | grep Alias |
 cut -d " " -f3- | while read alias
do expiry=`keytool -list -v -keystore "$truststore" -storepass ${storepassword} -
alias "${alias}" | grep Valid | perl -ne 'if(/until: (.*?)\n/) { print "$1\n"; }'`
   echo " Certificate ${alias} expires in '$expiry'"
done
```

macOS

下面是一个示例 Shell 脚本，它将证书捆绑包导入 macOS 上的信任存储。

```
mydir=tmp/certs
if [ ! -e "${mydir}" ]
then
mkdir -p "${mydir}"
fi truststore=${mydir}/rds-truststore.jks storepassword=changeit

curl -sS "https://rds-truststore.s3.cn-north-1.amazonaws.com.cn/global/global-
bundle.pem"> ${mydir}/global-bundle.pem
split -p "-----BEGIN CERTIFICATE-----" ${mydir}/global-bundle.pem rds-ca-

for CERT in rds-ca-*; do alias=$(openssl x509 -noout -text -in $CERT | perl -ne
 'next unless /Subject:/; s/.*(CN=|CN = )//; print')
  echo "Importing $alias"
  keytool -import -file ${CERT} -alias "${alias}" -storepass ${storepassword} -
keystore ${truststore} -noprompt
  rm $CERT
done

rm ${mydir}/global-bundle.pem

echo "Trust store content is: "

keytool -list -v -keystore "$truststore" -storepass ${storepassword} | grep Alias |
 cut -d " " -f3- | while read alias
do expiry=`keytool -list -v -keystore "$truststore" -storepass ${storepassword} -
alias "${alias}" | grep Valid | perl -ne 'if(/until: (.*?)\n/) { print "$1\n"; }'`
   echo " Certificate ${alias} expires in '$expiry'"
done
```

互联网络流量隐私

Amazon RDS 与本地应用程序之间以及 Amazon RDS 与同一 Amazon 区域内的其他 Amazon 资源之间的连接都受到保护。

服务与本地客户端和应用之间的流量

在您的私有网络和 Amazon 之间有两个连接选项：

- 一个Amazon Site-to-Site VPN 连接。有关更多信息，请参阅什么是 Amazon Site-to-Site VPN？
- Amazon Direct Connect 连接。有关更多信息，请参阅什么是 Amazon Direct Connect？

您可使用 Amazon 发布的 API 操作，通过网络获取对 Amazon RDS 的访问权限。客户端必须支持以下内容：

- 传输层安全性协议（TLS）。我们要求使用 TLS 1.2，建议使用 TLS 1.3。
- 具有完全向前保密（PFS）的密码套件，例如 DHE（临时 Diffie-Hellman）或 ECDHE（临时椭圆曲线 Diffie-Hellman）。大多数现代系统（如 Java 7 及更高版本）都支持这些模式。

此外，必须使用访问密钥 ID 和与 IAM 主体关联的秘密访问密钥来对请求进行签名。或者，您可以使用 Amazon Security Token Service（Amazon STS）生成临时安全凭证来对请求进行签名。

Amazon RDS 的 Identity and Access Management

Amazon Identity and Access Management（IAM）是一项 Amazon Web Services 服务，可以帮助管理员安全地控制对 Amazon 资源的访问。IAM 管理员控制谁可以通过身份验证（登录）和获得授权（具有权限）以使用 Amazon RDS 资源。IAM 是一项无需额外费用即可使用的 Amazon Web Services 服务。

主题

受众

如何使用 Amazon Identity and Access Management (IAM) 因您在 Amazon RDS 中执行的操作而异。

服务用户 – 如果您使用 Amazon RDS 服务来完成作业，则您的管理员会为您提供所需的凭证和权限。当您使用更多 Amazon RDS 功能来完成工作时，您可能需要额外权限。了解如何管理访问权限可帮助您向管理员请求适合的权限。如果您无法访问 Amazon RDS 中的一项功能，请参阅对 Amazon RDS 身份和访问权限问题进行故障排除。

服务管理员 – 如果您在公司负责管理 Amazon RDS 资源，则您可能具有 Amazon RDS 的完全访问权限。您有责任确定您的员工应访问哪些 Amazon RDS 功能和资源。然后，您必须向 管理员提交请求，这样才能更改您的服务用户的权限。请查看该页面上的信息以了解 IAM 的基本概念。要了解有关您的公司如何将 IAM 与 Amazon RDS 搭配使用的更多信息，请参阅Amazon RDS 如何与 IAM 协同工作。

管理员 – 如果您是管理员，您可能希望了解有关您可以如何编写策略以管理 Amazon RDS 的访问权限的详细信息。要查看您可在 IAM 中使用的基于身份的 Amazon RDS 示例策略，请参阅Amazon RDS 的基于身份的策略示例。

使用身份进行身份验证

身份验证是您使用身份凭证登录 Amazon 的方法。您必须作为 Amazon Web Services 账户根用户、IAM 用户或通过代入 IAM 角色进行身份验证（登录到 Amazon）。

如果您以编程方式访问 Amazon，则 Amazon 将提供软件开发工具包（SDK）和命令行界面（CLI），以便使用您的凭证以加密方式签署您的请求。如果您不使用 Amazon 工具，则必须自行对请求签名。有关使用推荐的方法自行签署请求的更多信息，请参阅《IAM 用户指南》中的用于签署 API 请求的 Amazon 签名版本 4。

无论使用何种身份验证方法，您都可能需要提供其他安全信息。例如，Amazon 建议您使用多重身份验证（MFA）来提高账户的安全性。要了解更多信息，请参阅《IAM 用户指南》中的 IAM 中的 Amazon 多重身份验证。

Amazon 账户根用户

当您创建 Amazon Web Services 账户时，最初使用的是一个对账户中所有 Amazon Web Services 服务和资源拥有完全访问权限的登录身份。此身份称为 Amazon Web Services 账户根用户，使用您创建账户时所用的电子邮件地址和密码登录，即可获得该身份。强烈建议您不要使用根用户执行日常任务。保护好根用户凭证，并使用这些凭证来执行仅根用户可以执行的任务。有关要求您以根用户身份登录的任务的完整列表，请参阅 IAM 用户指南中的需要根用户凭证的任务。

联合身份

作为最佳实践，要求人类用户（包括需要管理员访问权限的用户）结合使用联合身份验证和身份提供程序，以使用临时凭证来访问 Amazon Web Services 服务。

联合身份是来自企业用户目录、Web 身份提供程序、Amazon Directory Service、的用户，或任何使用通过身份源提供的凭证来访问 Amazon Web Services 服务的用户。当联合身份访问 Amazon Web Services 账户时，他们代入角色，而角色提供临时凭证。

IAM 用户和群组

IAM 用户是 Amazon Web Services 账户内对某个人员或应用程序具有特定权限的一个身份。在可能的情况下，我们建议使用临时凭证，而不是创建具有长期凭证（如密码和访问密钥）的 IAM 用户。

但是，如果您有一些特定的使用场景需要长期凭证以及 IAM 用户，建议您轮换访问密钥。有关更多信息，请参阅《IAM 用户指南》中的对于需要长期凭证的用例，应在需要时更新访问密钥。

IAM 组是一个指定一组 IAM 用户的身份。您不能使用组的身份登录。您可以使用组来一次性为多个用户指定权限。如果有大量用户，使用组可以更轻松地管理用户权限。例如，您可能具有一个名为 IAMAdmins 的组，并为该组授予权限以管理 IAM 资源。

用户与角色不同。用户唯一地与某个人员或应用程序关联，而角色旨在让需要它的任何人代入。用户具有永久的长期凭证，而角色提供临时凭证。要了解更多信息，请参阅《IAM 用户指南》中的 IAM 用户使用案例。

可以使用 IAM 数据库身份验证对数据库实例进行身份验证。

IAM 数据库身份验证使用以下数据库引擎：

- RDS for MariaDB
- RDS for MySQL
- RDS for PostgreSQL

有关使用 IAM 对数据库实例进行身份验证的更多信息，请参阅 适用于 MariaDB、MySQL 和 PostgreSQL 的 IAM 数据库身份验证。

IAM 角色

IAM 角色是 Amazon Web Services 账户中具有特定权限的身份。它类似于用户，但未与特定人员关联。您可以通过切换角色，在 Amazon Web Services Management Console 中暂时代入 IAM 角色。您可以调用 Amazon CLI 或 Amazon API 操作或使用自定义网址以担任角色。有关使用角色的方法的更多信息，请参阅《IAM 用户指南》中的使用 IAM 角色。

具有临时凭证的 IAM 角色在以下情况下很有用：

- 临时用户权限 – 用户可代入 IAM 角色，暂时获得针对特定任务的不同权限。
- 联合用户访问：要向联合身份分配权限，请创建角色并为角色定义权限。当联合身份进行身份验证时，该身份将与角色相关联并被授予由此角色定义的权限。有关用于联合身份验证的角色的信息，请参阅《IAM 用户指南》中的针对第三方身份提供商创建角色（联合身份验证）。
- 跨账户存取：您可以使用 IAM 角色以允许不同账户中的某个人（可信主体）访问您的账户中的资源。角色是授予跨账户访问权限的主要方式。但是，对于某些 Amazon Web Services 服务，您可以将策略直接附加到资源（而不是使用角色作为代理）。要了解用于跨账户访问的角色和基于资源的策略之间的差别，请参阅《IAM 用户指南》中的 IAM 角色与基于资源的策略有何不同。

- 跨服务访问：某些 Amazon Web Services 服务使用其它 Amazon Web Services 服务中的特征。例如，当您在某个服务中进行调用时，该服务通常会在 Amazon EC2 中运行应用程序或在 Simple Storage Service（Amazon S3）中存储对象。服务可能会使用发出调用的主体的权限、使用服务角色或使用服务相关角色来执行此操作。
  - 转发访问会话：当您使用 IAM 用户或角色在 Amazon 中执行操作时，您将被视为主体。使用某些服务时，您可能会执行一个操作，然后此操作在其他服务中启动另一个操作。FAS 使用主体调用 Amazon Web Services 服务的权限，结合请求的 Amazon Web Services 服务，向下游服务发出请求。只有在服务收到需要与其他 Amazon Web Services 服务或资源交互才能完成的请求时，才会发出 FAS 请求。在这种情况下，您必须具有执行这两项操作的权限。有关发出 FAS 请求时的策略详情，请参阅转发访问会话。
  - 服务角色 - 服务角色是服务代表您在您的账户中执行操作而分派的 IAM 角色。IAM 管理员可以在 IAM 中创建、修改和删除服务角色。有关更多信息，请参阅《IAM 用户指南》中的创建向 Amazon Web Services 服务委派权限的角色。
  - 服务相关角色：服务相关角色是与 Amazon Web Services 服务 关联的一种服务角色。服务可以代入代表您执行操作的角色。服务相关角色显示在您的 Amazon Web Services 账户 中，并由该服务拥有。IAM 管理员可以查看但不能编辑服务相关角色的权限。
- 在 Amazon EC2 上运行的应用程序：您可以使用 IAM 角色管理在 EC2 实例上运行并发出 Amazon CLI 或 Amazon API 请求的应用程序的临时凭证。这优先于在 EC2 实例中存储访问密钥。要将 Amazon 角色分配给 EC2 实例并使其对该实例的所有应用程序可用，您可以创建一个附加到实例的实例配置文件。实例配置文件包含角色，并使 EC2 实例上运行的程序能够获得临时凭证。有关更多信息，请参阅《IAM 用户指南》中的使用 IAM 角色向在 Amazon EC2 实例上运行的应用程序授予权限。

要了解是否使用 IAM 角色，请参阅 IAM 用户指南 中的何时创建 IAM 角色（而不是用户）。

使用策略管理访问

您将创建策略并将其附加到 IAM 身份或Amazon资源，以便控制Amazon中的访问。策略是 Amazon 中的对象；在与身份或资源相关联时，策略定义其权限。在某个实体（根用户、用户或 IAM 角色）发出请求时，Amazon 将评估这些策略。策略中的权限确定是允许还是拒绝请求。大多数策略在 Amazon 中存储为 JSON 文档。有关 JSON 策略文档的结构和内容的更多信息，请参阅 IAM 用户指南中的 JSON 策略概览。

管理员可以使用策略来指定哪些用户有权访问 Amazon 资源，以及他们可以对这些资源执行哪些操作。每个 IAM 实体（权限集或角色）最初没有任何权限。换言之，预设情况下，用户什么都不能做，甚至不能更改他们自己的密码。要为用户授予执行某些操作的权限，管理员必须将权限策略附加到用

户。或者，管理员可以将用户添加到具有预期权限的组中。当管理员为某个组授予访问权限时，该组内的全部用户都会获得这些访问权限。

IAM 策略定义操作的权限，无关乎您使用哪种方法执行操作。例如，假设您有一个允许 `iam:GetRole` 操作的策略。具有该策略的用户可以从 Amazon Web Services Management Console、Amazon CLI 或 Amazon API 获取角色信息。

基于身份的策略

基于身份的策略是可附加到身份（如权限集或角色）的 JSON 权限策略文档。这些策略控制身份可在何种条件下对哪些资源执行哪些操作。要了解如何创建基于身份的策略，请参阅 IAM 用户指南中的创建 IAM policy。

基于身份的策略可以进一步归类为内联策略或托管策略。内联策略直接嵌入到单个权限集或角色中。托管式策略是可以附加到 Amazon 账户中的多个权限集和角色的独立策略。托管式策略包括 Amazon 托管式策略和客户管理型策略。要了解如何在托管式策略和内联策略之间进行选择，请参阅 IAM 用户指南中的在托管式策略与内联策略之间进行选择。

有关特定于 Amazon RDS 的 Amazon 托管式策略的信息，请参阅适用于 Amazon RDS 的 Amazon 托管式策略。

其他策略类型

Amazon 支持额外的、不太常用的策略类型。这些策略类型可以设置更常用的策略类型向您授予的最大权限。

- 权限边界 – 权限边界是一项高级功能，借助该功能，您可以设置基于身份的策略可以授予 IAM 实体（权限集或角色）的最大权限。您可为实体设置权限边界。这些结果权限是实体的基于身份的策略及其权限边界的交集。在 `Principal` 字段中指定权限集或角色的基于资源的策略不受权限边界限制。任一项策略中的显式拒绝将覆盖允许。有关权限边界的更多信息，请参阅 IAM 用户指南中的 IAM 实体的权限边界。
- 服务控制策略 (SCP) – SCP 是 JSON 策略，指定了组织或组织单位 (OU) 在 Amazon Organizations 中的最大权限。Amazon Organizations 是一项服务，用于分组和集中管理您的企业拥有的多个 Amazon 账户。如果在组织内启用了所有特征，则可对任意或全部账户应用服务控制策略（SCP）。SCP 限制成员账户中实体（包括每个 Amazon Web Services 账户根用户）的权限。有关 Organizations 和 SCP 的更多信息，请参阅《Amazon Organizations 用户指南》中的 SCP 的工作原理。
- 会话策略 – 会话策略是当您以编程方式为角色或联合用户创建临时会话时作为参数传递的高级策略。结果会话的权限是权限集或角色的基于身份的策略和会话策略的交集。权限也可以来自基于资

源的策略。任一项策略中的显式拒绝将覆盖允许。有关更多信息，请参阅 IAM 用户指南中的会话策略。

多个策略类型

当多个类型的策略应用于一个请求时，生成的权限更加复杂和难以理解。要了解 Amazon 如何确定在涉及多种策略类型时是否允许请求，请参阅 IAM 用户指南中的策略评测逻辑。

Amazon RDS 如何与 IAM 协同工作

在使用 IAM 管理对 Amazon RDS 的访问之前，您应了解哪些 IAM 功能可与 Amazon RDS 协同工作。

下表列出了可与 Amazon RDS 结合使用的 IAM 功能：

IAM 特征	Amazon RDS 支持
基于身份的策略	是
基于资源的策略	否
策略操作	是
策略资源	是
策略条件键（特定于服务）	是
ACL	否
基于属性的访问控制 (ABAC)（策略中的标签）	是
临时凭证	是
转发访问会话	是
服务角色	是
服务相关角色	是

要大致了解 Amazon RDS 和其他 Amazon 服务如何与 IAM 一起使用，请参阅《IAM 用户指南》中的与 IAM 一起使用的 Amazon 服务。

主题

- [Amazon RDS 基于身份的策略]
- [Amazon RDS 内基于资源的策略]
- [Amazon RDS 的策略操作]
- [Amazon RDS 的策略资源]
- [Amazon RDS 的策略条件键]
- [Amazon RDS 中的访问控制列表 (ACL)]
- [策略中具有 Amazon RDS 标签的基于属性的访问控制 (ABAC)]
- [将临时凭证用于 Amazon RDS]
- [Amazon RDS 的转发访问会话]
- [Amazon RDS 的服务角色]
- [Amazon RDS 的服务相关角色]

Amazon RDS 基于身份的策略

支持基于身份的策略：是。

基于身份的策略是可附加到身份（如 IAM 用户、用户组或角色）的 JSON 权限策略文档。这些策略控制用户和角色可在何种条件下对哪些资源执行哪些操作。要了解如何创建基于身份的策略，请参阅《IAM 用户指南》中的使用客户托管策略定义自定义 IAM 权限。

通过使用 IAM 基于身份的策略，您可以指定允许或拒绝的操作和资源以及允许或拒绝操作的条件。您无法在基于身份的策略中指定主体，因为它适用于其附加的用户或角色。要了解可在 JSON 策略中使用的所有元素，请参阅《IAM 用户指南》中的 IAM JSON 策略元素引用。

Amazon RDS 基于身份的策略示例

要查看 Amazon RDS 基于身份的策略的示例，请参阅 Amazon RDS 的基于身份的策略示例。

Amazon RDS 内基于资源的策略

支持基于资源的策略：否。

基于资源的策略是附加到资源的 JSON 策略文档。基于资源的策略的示例包括 IAM 角色信任策略和 Amazon S3 存储桶策略。在支持基于资源的策略的服务中，服务管理员可以使用它们来控制对特定资源的访问。对于在其中附加策略的资源，策略定义指定主体可以对该资源执行哪些操作以及在什么条件

下执行。您必须在基于资源的策略中指定主体。主体可以包括账户、用户、角色、联合用户或 Amazon Web Services 服务。

要启用跨账户访问，您可以将整个账户或其他账户中的 IAM 实体指定为基于资源的策略中的主体。将跨账户主体添加到基于资源的策略只是建立信任关系工作的一半而已。当主体和资源处于不同的 Amazon Web Services 账户 中时，则信任账户中的 IAM 管理员还必须授予主体实体（用户或角色）对资源的访问权限。他们通过将基于身份的策略附加到实体以授予权限。但是，如果基于资源的策略向同一个账户中的主体授予访问权限，则不需要额外的基于身份的策略。有关更多信息，请参阅《IAM 用户指南》中的 IAM 中的跨账户资源访问。

Amazon RDS 的策略操作

支持策略操作：是。

管理员可以使用 Amazon JSON 策略来指定谁有权访问什么内容。也就是说，哪个主体可以对什么资源执行操作，以及在什么条件下执行。

JSON 策略的 `Action` 元素描述可用于在策略中允许或拒绝访问的操作。策略操作通常与关联的 Amazon API 操作同名。有一些例外情况，例如没有匹配 API 操作的仅限权限 操作。还有一些操作需要在策略中执行多个操作。这些附加操作称为相关操作。

在策略中包含操作以授予执行关联操作的权限。

Amazon RDS 中的策略操作在操作前使用以下前缀：`rds:`。例如，要授予某人使用 Amazon RDS `DescribeDBInstances` API 操作妙手数据库实例的权限，您应将 `rds:DescribeDBInstances` 操作纳入其策略中。策略语句必须包含 `Action` 或 `NotAction` 元素。Amazon RDS 定义了一组自己的操作，以描述您可以使用该服务执行的任务。

要在单个语句中指定多项 操作，请使用逗号将它们隔开，如下所示。

```
"Action": [
      "rds:action1",
      "rds:action2"
```

您也可以使用通配符 (*) 指定多个操作。例如，要指定以单词 `Describe` 开头的所有操作，请包括以下操作。

```
"Action": "rds:Describe*"
```

有关 Amazon RDS 操作的列表，请参阅服务授权参考中的 Amazon RDS 定义的操作。

Amazon RDS 的策略资源

支持策略资源：是。

管理员可以使用 Amazon JSON 策略来指定谁有权访问什么内容。也就是说，哪个主体可以对什么资源执行操作，以及在什么条件下执行。

`Resource` JSON 策略元素指定要向其应用操作的一个或多个对象。语句必须包含 `Resource` 或 `NotResource` 元素。作为最佳实践，请使用其 [Amazon 资源名称（ARN）](#)指定资源。对于支持特定资源类型（称为资源级权限）的操作，您可以执行此操作。

对于不支持资源级权限的操作（如列出操作），请使用通配符（*）指示语句应用于所有资源。

```
"Resource": "*"
```

数据库实例资源具有以下 Amazon 资源名称 (ARN)。

```
arn:${Partition}:rds:${Region}:${Account}:{ResourceType}/${Resource}
```

有关 ARN 格式的更多信息，请参阅 [Amazon Resource Name (ARN) 和 Amazon 服务命名空间](#)。

例如，要在语句中指定 `dbtest` 数据库实例，请使用以下 ARN。

```
"Resource": "arn:aws:rds:us-west-2:123456789012:db:dbtest"
```

要指定属于特定账户的所有数据库实例，请使用通配符 (*)。

```
"Resource": "arn:aws:rds:us-east-1:123456789012:db:*"
```

某些 RDS API 操作（例如，用于创建资源的那些操作）无法在特定资源上执行。在这些情况下，请使用通配符 (*)。

```
"Resource": "*"
```

许多 Amazon RDS API 操作涉及多种资源。例如，`CreateDBInstance` 创建数据库实例。您可以指定，在创建数据库实例时， 用户必须使用特定的安全组和参数组。要在单个语句中指定多个资源，请使用逗号分隔 ARN。

```
"Resource": [
```

```
        "resource1",
        "resource2"
```

有关 Amazon RDS 资源类型及其 ARN 的列表，请参阅服务授权参考中的 Amazon RDS 定义的操作。要了解您可以在哪些操作中指定每个资源的 ARN，请参阅 Amazon RDS 定义的操作。

Amazon RDS 的策略条件键

支持特定于服务的策略条件键：是。

管理员可以使用 Amazon JSON 策略来指定谁有权访问什么内容。也就是说，哪个主体可以对什么资源执行操作，以及在什么条件下执行。

在 `Condition` 元素（或 `Condition` 块）中，可以指定语句生效的条件。`Condition` 元素是可选的。您可以创建使用条件运算符（例如，等于或小于）的条件表达式，以使策略中的条件与请求中的值相匹配。

如果您在一个语句中指定多个 `Condition` 元素，或在单个 `Condition` 元素中指定多个键，则 Amazon 使用逻辑 `AND` 运算评估它们。如果您为单个条件键指定多个值，则 Amazon 使用逻辑 `OR` 运算来评估条件。在授予语句的权限之前必须满足所有的条件。

在指定条件时，您也可以使用占位符变量。例如，只有在使用 IAM 用户名标记 IAM 用户时，您才能为其授予访问资源的权限。有关更多信息，请参阅《IAM 用户指南》中的 IAM 策略元素：变量和标签。

Amazon 支持全局条件键和特定于服务的条件键。要查看所有 Amazon 全局条件键，请参阅《IAM 用户指南》中的 Amazon 全局条件上下文键。

Amazon RDS 定义了自己的一组条件键，还支持使用一些全局条件键。要查看所有 Amazon 全局条件键，请参阅 IAM 用户指南中的 Amazon 全局条件上下文键 。

所有 RDS API 操作都支持 `aws:RequestedRegion` 条件键。

有关 Amazon RDS 条件键的列表，请参阅服务授权参考中的 Amazon RDS 的条件键。要了解您可以对哪些操作和资源使用条件键，请参阅 Amazon RDS 定义的操作。

Amazon RDS 中的访问控制列表 (ACL)

支持访问控制列表（ACL）：否

访问控制列表（ACL）控制哪些主体（账户成员、用户或角色）有权访问资源。ACL 与基于资源的策略类似，但它们不使用 JSON 策略文档格式。

策略中具有 Amazon RDS 标签的基于属性的访问控制 (ABAC)

在策略中支持基于属性的访问权限控制（ABAC）标签：是

基于属性的访问控制（ABAC）是一种授权策略，该策略基于属性来定义权限。在 Amazon 中，这些属性称为标签。您可以将标签附加到 IAM 实体（用户或角色）以及 Amazon 资源。标记实体和资源是 ABAC 的第一步。然后设计 ABAC 策略，以在主体的标签与他们尝试访问的资源标签匹配时允许操作。

ABAC 在快速增长的环境中非常有用，并在策略管理变得繁琐的情况下可以提供帮助。

要基于标签控制访问，您需要使用 `aws:ResourceTag/key-name`、`aws:RequestTag/key-name` 或 `aws:TagKeys` 条件键在策略的条件元素中提供标签信息。

如果某个服务对于每种资源类型都支持所有这三个条件键，则对于该服务，该值为是。如果某个服务仅对于部分资源类型支持所有这三个条件键，则该值为部分。

有关 ABAC 的更多信息，请参阅《IAM 用户指南》中的使用 ABAC 授权定义权限。要查看设置 ABAC 步骤的教程，请参阅《IAM 用户指南》中的使用基于属性的访问权限控制（ABAC）。

有关标记 Amazon RDS 资源的更多信息，请参阅指定条件：使用自定义标签。要查看基于身份的策略（用于根据资源上的标签来限制对该资源的访问）的示例，请参阅授予权限以允许对在特定标签中包含两个不同值的资源执行操作。

将临时凭证用于 Amazon RDS

支持临时凭证：是。

某些 Amazon Web Services 服务 在您使用临时凭证登录时无法正常工作。有关更多信息，包括 Amazon Web Services 服务 与临时凭证配合使用，请参阅 IAM 用户指南中的使用 IAM 的 Amazon Web Services 服务。

如果您不使用用户名和密码而用其他方法登录到 Amazon Web Services Management Console，则使用临时凭证。例如，当您使用贵公司的单点登录（SSO）链接访问 Amazon 时，该过程将自动创建临时凭证。当您以用户身份登录控制台，然后切换角色时，您还会自动创建临时凭证。有关切换角色的更多信息，请参阅《IAM 用户指南》中的从用户切换到 IAM 角色（控制台）。

您可以使用 Amazon CLI 或者 Amazon API 创建临时凭证。之后，您可以使用这些临时凭证访问 Amazon。Amazon 建议您动态生成临时凭证，而不是使用长期访问密钥。有关更多信息，请参阅 IAM 中的临时安全凭证。

Amazon RDS 的转发访问会话

支持转发访问会话：是。

当您使用 IAM 用户或角色在 Amazon 中执行操作时，您将被视为主体。使用某些服务时，您可能会执行一个操作，然后此操作在其他服务中启动另一个操作。FAS 使用主体调用 Amazon Web Services 服务的权限，结合请求的 Amazon Web Services 服务，向下游服务发出请求。只有在服务收到需要与其他 Amazon Web Services 服务或资源交互才能完成的请求时，才会发出 FAS 请求。在这种情况下，您必须具有执行这两项操作的权限。有关发出 FAS 请求时的策略详情，请参阅转发访问会话。

Amazon RDS 的服务角色

支持服务角色：是。

服务角色是由一项服务担任、代表您执行操作的 IAM 角色。IAM 管理员可以在 IAM 中创建、修改和删除服务角色。有关更多信息，请参阅《IAM 用户指南》中的创建向 Amazon Web Services 服务委派权限的角色。

> **Warning**
>
> 更改服务角色的权限可能会破坏 Amazon RDS 功能。仅当 Amazon RDS 提供相关指导时才编辑服务角色。

Amazon RDS 的服务相关角色

支持服务相关角色：是。

服务相关角色是一种与 Amazon Web Services 服务 相关的服务角色。服务可以代入代表您执行操作的角色。服务相关角色显示在您的 Amazon Web Services 账户 中，并由该服务拥有。IAM 管理员可以查看但不能编辑服务相关角色的权限。

有关使用 Amazon RDS 服务相关角色的详细信息，请参阅将服务相关角色用于 Amazon RDS。

Amazon RDS 的基于身份的策略示例

原定设置情况下，权限集和角色没有创建或修改 Amazon RDS 资源的权限。它们还无法使用 Amazon Web Services Management Console、Amazon CLI 或 Amazon API 执行任务。管理员必须创建 IAM policy，以便为权限集和角色授予权限以对所需的指定资源执行特定的 API 操作。然后，管理员必须将这些策略附加到需要这些权限的权限集或角色。

要了解如何使用这些示例 JSON 策略文档创建 IAM 基于身份的策略，请参阅《IAM 用户指南》中的在 JSON 选项卡上创建策略。

主题

策略最佳实践

基于身份的策略确定某个人是否可以创建、访问或删除您账户中的 Amazon RDS 资源。这些操作可能会使 Amazon Web Services 账户 产生成本。创建或编辑基于身份的策略时，请遵循以下指南和建议：

- Amazon 托管式策略及转向最低权限许可入门：要开始向用户和工作负载授予权限，请使用 Amazon 托管式策略来为许多常见使用场景授予权限。您可以在 Amazon Web Services 账户 中找到这些策略。我们建议通过定义特定于您的使用场景的 Amazon 客户托管式策略来进一步减少权限。有关更多信息，请参阅《IAM 用户指南》中的 Amazon 托管式策略或工作职能的 Amazon 托管式策略。
- 应用最低权限：在使用 IAM 策略设置权限时，请仅授予执行任务所需的权限。为此，您可以定义在特定条件下可以对特定资源执行的操作，也称为最低权限许可。有关使用 IAM 应用权限的更多信息，请参阅《IAM 用户指南》中的 IAM 中的策略和权限。
- 使用 IAM 策略中的条件进一步限制访问权限：您可以向策略添加条件来限制对操作和资源的访问。例如，您可以编写策略条件来指定必须使用 SSL 发送所有请求。如果通过特定 Amazon Web Services 服务（例如 Amazon CloudFormation）使用服务操作，您还可以使用条件来授予对服务操作的访问权限。有关更多信息，请参阅《IAM 用户指南》中的 IAM JSON 策略元素：条件。
- 使用 IAM Access Analyzer 验证您的 IAM 策略，以确保权限的安全性和功能性：IAM Access Analyzer 会验证新策略和现有策略，以确保策略符合 IAM 策略语言 (JSON) 和 IAM 最佳实践。IAM Access Analyzer 提供 100 多项策略检查和可操作的建议，以帮助您制定安全且功能性强的策略。有关更多信息，请参阅《IAM 用户指南》中的使用 IAM Access Analyzer 验证策略。
- 需要多重身份验证（MFA）：如果您所处的场景要求您的 Amazon Web Services 账户 中有 IAM 用户或根用户，请启用 MFA 来提高安全性。若要在调用 API 操作时需要 MFA，请将 MFA 条件添加到您的策略中。有关更多信息，请参阅《IAM 用户指南》中的使用 MFA 保护 API 访问。

有关 IAM 中的最佳实操的更多信息，请参阅《IAM 用户指南》中的 IAM 中的安全最佳实践。

使用 Amazon RDS 控制台

要访问 Amazon RDS 控制台，您必须拥有一组最低的权限。这些权限必须允许您列出和查看有关您的 Amazon Web Services 账户中的 Amazon RDS 资源的详细信息。如果创建比必需的最低权限更为严格的基于身份的策略，对于附加了该策略的实体（用户或角色），控制台将无法按预期正常运行。

对于只需要调用 Amazon CLI 或 Amazon API 的用户，无需为其提供最低控制台权限。相反，只允许访问与您尝试执行的 API 操作相匹配的操作。

要确保这些实体仍可使用 Amazon RDS 控制台，也可向实体附加以下Amazon托管策略。

```
AmazonRDSReadOnlyAccess
```

有关更多信息，请参阅 IAM 用户指南 中的为用户添加权限。

使用控制台所需的权限

对于要使用控制台的用户，该用户必须拥有一组最小权限。这些权限允许用户描述其Amazon账户的 Amazon RDS 资源并提供其他相关信息（包括 Amazon EC2 安全和网络信息）。

如果创建比必需的最低权限更为严格的 IAM 策略，对于附加了该 IAM 策略的用户，控制台无法按预期正常运行。要确保这些用户仍可使用控制台，也可向用户附加 `AmazonRDSReadOnlyAccess` 托管策略，如使用策略管理访问中所述。

对于只需要调用 Amazon CLI 或 Amazon RDS API 的用户，无需为其提供最低限度的控制台权限。

以下策略授予对Amazon根账户的所有 Amazon RDS 资源的完全访问权：

```
AmazonRDSFullAccess
```

允许用户查看他们自己的权限

该示例说明了您如何创建策略，以允许 IAM 用户查看附加到其用户身份的内联和托管式策略。此策略包括在控制台上完成此操作或者以编程方式使用 Amazon CLI 或 Amazon API 所需的权限。

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

在 Amazon RDS 中创建、修改和删除资源的权限策略

以下各节展示了授予和限制资源访问权限的权限策略示例：

允许用户在 Amazon 账户中创建数据库实例

以下是支持 ID 为 123456789012 的账户为您的 Amazon 账户创建数据库实例的示例策略。该策略要求新数据库实例的名称以 test 开头。新数据库实例还必须使用 MySQL 数据库引擎和 db.t2.micro 数据库实例类。此外，新数据库实例必须使用以 default 开头的选项组和数据库参数组，并且它必须使用 default 子网组。

```
{
   "Version": "2012-10-17",
```

```
    "Statement": [
       {
          "Sid": "AllowCreateDBInstanceOnly",
          "Effect": "Allow",
          "Action": [
             "rds:CreateDBInstance"
          ],
          "Resource": [
             "arn:aws:rds:*:123456789012:db:test*",
             "arn:aws:rds:*:123456789012:og:default*",
             "arn:aws:rds:*:123456789012:pg:default*",
             "arn:aws:rds:*:123456789012:subgrp:default"
          ],
          "Condition": {
             "StringEquals": {
                "rds:DatabaseEngine": "mysql",
                "rds:DatabaseClass": "db.t2.micro"
             }
          }
       }
    ]
 }
```

此策略包含一个为 用户指定以下权限的语句：

- 该策略可让该账户使用 CreateDBInstance API 操作创建数据库实例（这还适用于 create-db-instance Amazon CLI 命令和 Amazon Web Services Management Console）。
- Resource 元素指定用户可以执行操作的资源。使用 Amazon Resource Name (ARN) 指定资源。此 ARN 包括资源所属服务的名称（rds）、Amazon 区域（在该示例中，* 指示任何区域）、Amazon 账号（在该示例中，123456789012 为账号）以及资源的类型。有关创建 ARN 的更多信息，请参阅Amazon RDS 中的 Amazon 资源名称（ARN）。

  该示例中的 Resource 元素为用户指定有关资源的以下策略限制：
  - 新数据库实例的数据库实例标识符必须以 test 开头 (例如，testCustomerData1、test-region2-data)。
  - 新数据库实例的选项组必须以 default 开头。
  - 新数据库实例的数据库参数组必须以 default 开头。
  - 新数据库实例的子网组必须是 default 子网组。
- Condition 元素指定数据库引擎必须是 MySQL 并且数据库实例类必须是 db.t2.micro。Condition 元素指定策略生效的条件。您可以通过使用 Condition 元素添加

其他权限或限制。有关指定条件的更多信息，请参阅Amazon RDS 的策略条件键。此示例指定 `rds:DatabaseEngine` 和 `rds:DatabaseClass` 条件。有关 `rds:DatabaseEngine` 的有效条件值的信息，请参阅 CreateDBInstance 中的 `Engine` 参数下的列表。有关 `rds:DatabaseClass` 的有效条件值的信息，请参阅 数据库实例类支持的数据库引擎 。

该策略不指定 `Principal` 元素，因为在基于身份的策略中，您未指定获取权限的委托人。附加了策略的用户是隐式委托人。向 IAM 角色附加权限策略后，该角色的信任策略中标识的委托人将获取权限。

有关 Amazon RDS 操作的列表，请参阅服务授权参考中的 Amazon RDS 定义的操作。

允许用户对任何 RDS 资源执行任何 Describe 操作

以下权限策略对用户授予权限以运行以 `Describe` 开头的所有操作。这些操作显示有关 RDS 资源 (如数据库实例) 的信息。`Resource` 元素中的通配符 (*) 表示可对账户拥有的所有 Amazon RDS 资源执行操作。

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "AllowRDSDescribe",
         "Effect": "Allow",
         "Action": "rds:Describe*",
         "Resource": "*"
      }
   ]
}
```

允许用户创建使用指定数据库参数组和子网组的数据库实例

以下权限策略授予权限以允许用户仅创建必须使用 `mydbpg` 数据库参数组和 `mydbsubnetgroup` 数据库子网组的数据库实例。

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "VisualEditor0",
         "Effect": "Allow",
         "Action": "rds:CreateDBInstance",
```

```
        "Resource": [
           "arn:aws:rds:*:*:pg:mydbpg",
           "arn:aws:rds:*:*:subgrp:mydbsubnetgroup"
        ]
      }
   ]
}
```

授予权限以允许对在特定标签中包含两个不同值的资源执行操作

您可以在基于身份的策略中使用条件，以便基于标签控制对 Amazon RDS 资源的访问。以下策略所授予的权限允许对 `stage` 标签设置为 `development` 或 `test` 的实例执行 `CreateDBSnapshot` API 操作。

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"AllowAnySnapshotName",
         "Effect":"Allow",
         "Action":[
            "rds:CreateDBSnapshot"
         ],
         "Resource":"arn:aws:rds:*:123456789012:snapshot:*"
      },
      {
         "Sid":"AllowDevTestToCreateSnapshot",
         "Effect":"Allow",
         "Action":[
            "rds:CreateDBSnapshot"
         ],
         "Resource":"arn:aws:rds:*:123456789012:db:*",
         "Condition":{
            "StringEquals":{
                "rds:db-tag/stage":[
                  "development",
                  "test"
               ]
            }
         }
      }
   ]
}
```

以下策略所授予的权限允许对 `stage` 标签设置为 `development` 或 `test` 的实例执行 `ModifyDBInstance` API 操作。

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"AllowChangingParameterOptionSecurityGroups",
         "Effect":"Allow",
         "Action":[
            "rds:ModifyDBInstance"
         ],
         "Resource": [
            "arn:aws:rds:*:123456789012:pg:*",
            "arn:aws:rds:*:123456789012:secgrp:*",
            "arn:aws:rds:*:123456789012:og:*"
         ]
      },
      {
         "Sid":"AllowDevTestToModifyInstance",
         "Effect":"Allow",
         "Action":[
            "rds:ModifyDBInstance"
         ],
         "Resource":"arn:aws:rds:*:123456789012:db:*",
         "Condition":{
            "StringEquals":{
                "rds:db-tag/stage":[
                  "development",
                  "test"
               ]
            }
         }
      }
   ]
}
```

防止用户删除数据库实例

以下权限策略授予权限以防止用户删除特定数据库实例。例如，您可能想禁止任何非管理员用户删除您的生产数据库实例。

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "DenyDelete1",
         "Effect": "Deny",
         "Action": "rds:DeleteDBInstance",
         "Resource": "arn:aws:rds:us-west-2:123456789012:db:my-mysql-instance"
      }
   ]
}
```

拒绝对资源的所有访问

您可以明确拒绝对资源的访问。拒绝策略优先于允许策略。以下策略明确拒绝用户管理资源的能力：

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Deny",
         "Action": "rds:*",
         "Resource": "arn:aws:rds:us-east-1:123456789012:db:mydb"
      }
   ]
}
```

示例策略：使用条件键

以下示例说明了如何在 Amazon RDS IAM 权限策略中使用条件键。

示例 1：授予权限以创建使用特定数据库引擎的非多可用区数据库实例

以下策略使用 RDS 条件键，并仅允许用户创建采用 MySQL 数据库引擎且不使用多可用区的数据库实例。`Condition` 元素指示数据库引擎须为 MySQL 的要求。

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "AllowMySQLCreate",
```

```
            "Effect": "Allow",
            "Action": "rds:CreateDBInstance",
            "Resource": "*",
            "Condition": {
               "StringEquals": {
                  "rds:DatabaseEngine": "mysql"
               },
               "Bool": {
                  "rds:MultiAz": false
               }
            }
         }
      ]
}
```

示例 2：明确拒绝权限，以禁止创建特定数据库实例类的数据库实例和使用预置 IOPS 的数据库实例

以下策略显式拒绝创建使用数据库实例类 `r3.8xlarge` 和 `m4.10xlarge`（最大、最贵的数据库实例类）的数据库实例的权限。此策略还禁止用户创建使用预置的 IOPS (这会带来额外成本) 的数据库实例。

显式拒绝权限会取代授予的任何其他权限。这可确保用户身份不会无意中获得您绝不希望授予的权限。

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "DenyLargeCreate",
         "Effect": "Deny",
         "Action": "rds:CreateDBInstance",
         "Resource": "*",
         "Condition": {
            "StringEquals": {
               "rds:DatabaseClass": [
                  "db.r3.8xlarge",
                  "db.m4.10xlarge"
               ]
            }
         }
      },
      {
         "Sid": "DenyPIOPSCreate",
         "Effect": "Deny",
```

```
            "Action": "rds:CreateDBInstance",
            "Resource": "*",
            "Condition": {
               "NumericNotEquals": {
                  "rds:Piops": "0"
               }
            }
         }
      ]
}
```

示例 3：限制可用于对资源进行标记的一组标签键和值的值

下面的策略使用 RDS 条件键，并允许将键为 `stage` 的标签添加到值为 `test`、`qa` 和 `production` 的资源。

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": [
            "rds:AddTagsToResource",
            "rds:RemoveTagsFromResource"
         ],
         "Resource": "*",
         "Condition": {
            "streq": {
               "rds:req-tag/stage": [
                  "test",
                  "qa",
                  "production"
               ]
            }
         }
      }
   ]
}
```

指定条件：使用自定义标签

Amazon RDS 支持在 IAM 策略中使用自定义标签指定条件。

例如，假定您将一个名为 environment 的标签添加到具有 beta、staging、production 等值的数据库实例。如果您这样做，则可创建一个策略来根据 environment 标签值以仅允许某些用户使用数据库实例。

> **Note**
>
> 自定义标签标识符区分大小写。

下表列出了可以在 Condition 元素中使用的 RDS 标签标识符。

RDS 标签标识符	适用于
db-tag	数据库实例，包括只读副本
snapshot-tag	数据库快照
ri-tag	预留数据库实例
og-tag	数据库选项组
pg-tag	数据库参数组
subgrp-tag	数据库子网组
es-tag	事件订阅
cluster-tag	数据库集群
cluster-pg-tag	数据库集群参数组
cluster-snapshot-tag	数据库集群快照

自定义标签条件的语法如下：

```
"Condition":{"StringEquals":{"rds:rds-tag-identifier/tag-name":
["value"]} }
```

例如，以下 Condition 元素适用于具有名为 environment 的标签且标签值为 production 的数据库实例。

```
"Condition":{"StringEquals":{"rds:db-tag/environment": ["production"]} }
```

有关创建标签的信息，请参阅为 Amazon RDS 资源添加标签。

> **Important**
>
> 如果您使用标签管理对 RDS 资源的访问，建议您保护对 RDS 资源的标签的访问。您可通过为 `AddTagsToResource` 和 `RemoveTagsFromResource` 操作创建策略来管理对标签的访问。例如，以下策略不允许用户为所有资源添加或删除标签。之后，您可创建策略来允许特定用户添加或删除标签。
>
> ```
> {
>    "Version":"2012-10-17",
>    "Statement":[
>       {
>          "Sid":"DenyTagUpdates",
>          "Effect":"Deny",
>          "Action":[
>             "rds:AddTagsToResource",
>             "rds:RemoveTagsFromResource"
>          ],
>          "Resource":"*"
>       }
>    ]
> }
> ```

有关 Amazon RDS 操作的列表，请参阅服务授权参考中的 Amazon RDS 定义的操作。

示例策略：使用自定义标签

以下示例说明了如何在 Amazon RDS IAM 权限策略中使用自定义标签。有关向 Amazon RDS 资源添加标签的更多信息，请参阅Amazon RDS 中的 Amazon 资源名称（ARN）。

> **Note**
>
> 所有示例都使用 us-west-2 区域和虚构的账户 ID。

示例 1：授予权限以允许对在特定标签中包含两个不同值的资源执行操作

以下策略所授予的权限允许对 stage 标签设置为 development 或 test 的实例执行 CreateDBSnapshot API 操作。

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"AllowAnySnapshotName",
         "Effect":"Allow",
         "Action":[
            "rds:CreateDBSnapshot"
         ],
         "Resource":"arn:aws:rds:*:123456789012:snapshot:*"
      },
      {
         "Sid":"AllowDevTestToCreateSnapshot",
         "Effect":"Allow",
         "Action":[
            "rds:CreateDBSnapshot"
         ],
         "Resource":"arn:aws:rds:*:123456789012:db:*",
         "Condition":{
            "StringEquals":{
                "rds:db-tag/stage":[
                  "development",
                  "test"
               ]
            }
         }
      }
   ]
}
```

以下策略所授予的权限允许对 stage 标签设置为 development 或 test 的实例执行 ModifyDBInstance API 操作。

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"AllowChangingParameterOptionSecurityGroups",
```

```
         "Effect":"Allow",
         "Action":[
            "rds:ModifyDBInstance"
         ],
         "Resource":" [
            "arn:aws:rds:*:123456789012:pg:*",
            "arn:aws:rds:*:123456789012:secgrp:*",
            "arn:aws:rds:*:123456789012:og:*"
         ]
      },
      {
         "Sid":"AllowDevTestToModifyInstance",
         "Effect":"Allow",
         "Action":[
            "rds:ModifyDBInstance"
         ],
         "Resource":"arn:aws:rds:*:123456789012:db:*",
         "Condition":{
            "StringEquals":{
                "rds:db-tag/stage":[
                  "development",
                  "test"
               ]
            }
         }
      }
   ]
}
```

示例 2：明确拒绝权限，以禁止创建使用指定数据库参数组的数据库实例

以下策略通过显式拒绝权限，禁止创建在数据库参数组中包含特定标签值的数据库实例。如果您需要在创建数据库实例时始终使用特定客户创建的数据库参数组，则可以应用此策略。使用 Deny 的策略最常用于限制由更宽泛的策略所授予的访问权限。

显式拒绝权限会取代授予的任何其他权限。这可确保用户身份不会无意中获得您绝不希望授予的权限。

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
```

```
            "Sid":"DenyProductionCreate",
            "Effect":"Deny",
            "Action":"rds:CreateDBInstance",
            "Resource":"arn:aws:rds:*:123456789012:pg:*",
            "Condition":{
               "StringEquals":{
                  "rds:pg-tag/usage":"prod"
               }
            }
         }
      ]
}
```

示例 3：授予权限以允许对实例名称以用户名为前缀的数据库实例执行操作

以下策略授予的权限允许对具有如下性质的数据库实例调用除 AddTagsToResource 和 RemoveTagsFromResource 以外的任何 API：该数据库实例的实例名称以用户名称作为前缀，并且具有值为 stage 的 devo 标签或没有名为 stage 的标签。

策略中的 Resource 行通过 Amazon Resource Name (ARN) 标识资源。有关对 Amazon RDS 资源使用 ARN 的更多信息，请参阅[Amazon RDS 中的 Amazon 资源名称（ARN）](#)。

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"AllowFullDevAccessNoTags",
         "Effect":"Allow",
         "NotAction":[
            "rds:AddTagsToResource",
            "rds:RemoveTagsFromResource"
         ],
         "Resource":"arn:aws:rds:*:123456789012:db:${aws:username}*",
         "Condition":{
            "StringEqualsIfExists":{
               "rds:db-tag/stage":"devo"
            }
         }
      }
   ]
}
```

适用于 Amazon RDS 的 Amazon 托管式策略

要向权限集和角色添加权限，与自己编写策略相比，使用 Amazon 托管式策略更简单。创建仅为团队提供所需权限的 IAM 客户管理型策略需要时间和专业知识。要快速入门，您可以使用我们的 Amazon 托管策略。这些策略涵盖常见使用案例，可在您的 Amazon Web Services 账户 中使用。有关 Amazon 托管策略的更多信息，请参阅《IAM 用户指南》中的Amazon 托管策略。

Amazon Web Services 服务 负责维护和更新 Amazon 托管策略。您无法更改 Amazon 托管策略中的权限。服务偶尔会向 Amazon 管理型策略添加额外权限以支持新特征。此类型的更新会影响附加了策略的所有身份（权限集和角色）。当启动新特征或新操作可用时，服务最有可能会更新 Amazon 管理型策略。服务不会从 Amazon 托管式策略中删除权限，因此策略更新不会破坏您的现有权限。

此外，Amazon还支持跨多种服务的工作职能的托管式策略。例如，`ReadOnlyAccess` Amazon 托管式策略提供对许多 Amazon Web Services 服务和资源的只读访问权限。当服务启动新特征时，Amazon 会为新操作和资源添加只读权限。有关工作职能策略的列表和说明，请参阅《IAM 用户指南》中的适用于工作职能的 Amazon 托管式策略。

主题

Amazon 托管式策略：AmazonRDSReadOnlyAccess

此策略允许通过 Amazon Web Services Management Console 对 Amazon RDS 进行只读访问。

权限详细信息

该策略包含以下权限：

- `rds` – 允许主体描述 Amazon RDS 资源并列出 Amazon RDS 资源的标签。
- `cloudwatch` – 允许主体获取 Amazon CloudWatch 指标统计数据。
- `ec2` – 可让主体描述可用区和网络资源。
- `logs` – 允许主体描述日志组的 CloudWatch Logs 日志流，并获取 CloudWatch Logs 日志事件。
- `devops-guru` – 允许主体描述具有 Amazon DevOps Guru 覆盖范围的资源，该覆盖范围由 CloudFormation 堆栈名称或资源标签指定。

有关此策略的更多信息，包括 JSON 策略文档，请参阅《Amazon 托管式策略参考指南》中的 AmazonRDSReadOnlyAccess。

Amazon 托管式策略：AmazonRDSFullAccess

此策略通过 Amazon Web Services Management Console 提供了对 Amazon RDS 的完全访问权限。

权限详细信息

该策略包含以下权限：

- `rds` – 允许主体完全访问 Amazon RDS。
- `application-autoscaling` – 允许主体描述和管理 Application Auto Scaling 扩展目标和策略。
- `cloudwatch` – 允许主体获取 CloudWatch 指标统计数据并管理 CloudWatch 警报。
- `ec2` – 可让主体描述可用区和网络资源。
- `logs` – 允许主体描述日志组的 CloudWatch Logs 日志流，并获取 CloudWatch Logs 日志事件。
- `outposts` – 允许主体获取 Amazon Outposts 实例类型。
- `pi` – 允许主体获取 Performance Insights 指标。
- `sns` – 允许主体访问 Amazon Simple Notification Service (Amazon SNS) 订阅和主题，并发布 Amazon SNS 消息。
- `devops-guru` – 允许主体描述具有 Amazon DevOps Guru 覆盖范围的资源，该覆盖范围由 CloudFormation 堆栈名称或资源标签指定。

有关此策略的更多信息，包括 JSON 策略文档，请参阅《Amazon 托管式策略参考指南》中的 AmazonRDSFullAccess。

Amazon 托管式策略：AmazonRDSDataFullAccess

此策略提供完全访问权限，允许在特定 Amazon Web Services 账户中的 Aurora Serverless 集群上使用 Data API 和查询编辑器。此策略允许 Amazon Web Services 账户从 Amazon Secrets Manager 获取密钥的值。

您可以将 `AmazonRDSDataFullAccess` 策略附加到 IAM 身份。

权限详细信息

该策略包含以下权限：

- `dbqms` – 允许主体访问、创建、删除、描述和更新查询。Database Query Metadata Service (dbqms) 是一项仅限内部使用的服务。它为 Amazon Web Services Management Console 上多项 Amazon Web Services 服务（包括 Amazon RDS）提供查询编辑器最近的和保存的查询。
- `rds-data` – 允许主体在 Aurora Serverless 数据库上运行 SQL 语句。
- `secretsmanager` – 允许主体从 Amazon Secrets Manager 获取密钥的值。

有关此策略的更多信息，包括 JSON 策略文档，请参阅《Amazon 托管式策略参考指南》中的 AmazonRDSDataFullAccess。

Amazon 托管式策略：AmazonRDSEnhancedMonitoringRole

此策略提供了对 Amazon CloudWatch Logs 的访问权限，支持 Amazon RDS 增强监控。

权限详细信息

该策略包含以下权限：

- `logs` – 允许主体创建 CloudWatch Logs 日志组和保留策略，并创建和描述日志组的 CloudWatch Logs 日志流。它还允许主体放置和获取 CloudWatch Logs 日志事件。

有关此策略的更多信息，包括 JSON 策略文档，请参阅《Amazon 托管式策略参考指南》中的 AmazonRDSEnhancedMonitoringRole。

Amazon 托管式策略：AmazonRDSPerformanceInsightsReadOnly

此策略提供了对 Amazon RDS Performance Insights 的只读访问权限，用于处理 Amazon RDS 数据库实例和 Amazon Aurora 数据库集群。

此策略现在包含 `Sid`（语句 ID）作为策略语句的标识符。

权限详细信息

该策略包含以下权限：

- `rds` – 允许主体描述 Amazon RDS 数据库实例和 Amazon Aurora 数据库集群。
- `pi` – 允许主体调用 Amazon RDS Performance Insights API 并访问 Performance Insights 指标。

有关此策略的更多信息，包括 JSON 策略文档，请参阅《Amazon 托管式策略参考指南》中的 AmazonRDSPerformanceInsightsReadOnly。

Amazon 托管式策略：AmazonRDSPerformanceInsightsFullAccess

此策略提供了对 Amazon RDS 性能详情的完全访问权限，用于处理 Amazon RDS 数据库实例和 Amazon Aurora 数据库集群。

此策略现在包含 `Sid`（语句 ID）作为策略语句的标识符。

权限详细信息

该策略包含以下权限：

- `rds` – 允许主体描述 Amazon RDS 数据库实例和 Amazon Aurora 数据库集群。
- `pi` – 允许主体调用 Amazon RDS 性能详情 API，以及创建、查看和删除性能分析报告。
- `cloudwatch` – 允许主体列出所有 Amazon CloudWatch 指标，并获取指标数据和统计数据。

有关此策略的更多信息，包括 JSON 策略文档，请参阅《Amazon 托管式策略参考指南》中的 AmazonRDSPerformanceInsightsFullAccess。

Amazon 托管式策略：AmazonRDSDirectoryServiceAccess

此策略允许 Amazon RDS 调用 Amazon Directory Service。

权限详细信息

此策略包含以下权限：

- `ds` – 允许主体描述 Amazon Directory Service 目录并控制对 Amazon Directory Service 目录的授权。

有关此策略的更多信息，包括 JSON 策略文档，请参阅《Amazon 托管式策略参考指南》中的 AmazonRDSDirectoryServiceAccess。

Amazon 托管式策略：AmazonRDSServiceRolePolicy

您不能将 `AmazonRDSServiceRolePolicy` 策略附加到您的 IAM 实体。此附加到服务相关角色的策略允许 Amazon RDS 代表您执行操作。有关更多信息，请参阅 Amazon RDS 的服务相关角色权限。

Amazon 托管式策略：AmazonRDSCustomServiceRolePolicy

您不能将 `AmazonRDSCustomServiceRolePolicy` 策略附加到您的 IAM 实体。此策略附加到一个服务相关角色，该角色支持 Amazon RDS 代表 RDS 数据库资源调用 Amazon 服务。

该策略包含以下权限：

- `ec2` - 支持 RDS Custom 在提供时间点还原功能的数据库实例上执行备份操作。
- `secretsmanager` - 允许 RDS Custom 管理由 RDS Custom 创建的数据库实例特定密钥。
- `cloudwatch` - 支持 RDS Custom 通过 CloudWatch 代理将数据库实例指标和日志上传到 CloudWatch。
- `events`、`sqs` - 允许 RDS Custom 发送和接收有关数据库实例的状态信息。

有关此策略的更多信息，包括 JSON 策略文档，请参阅《Amazon 托管式策略参考指南》中的 AmazonRDSCustomServiceRolePolicy。

Amazon 托管式策略：AmazonRDSCustom实例ProfileRolePolicy

您不应将 `AmazonRDSCustomInstanceProfileRolePolicy` 附加到自己的 IAM 实体。只能将其附加到实例配置文件角色，该角色用于向您的 Amazon RDS 自定义数据库实例授予执行各种自动化操作和数据库管理任务的权限。在创建 RDS 自定义实例期间将实例配置文件作为 `custom-iam-instance-profile` 参数传递，并且 RDS Custom 会将此实例配置文件关联到您的数据库实例。

权限详细信息

该策略包含以下权限：

- `ssm`、`ssmmessages`、`ec2messages` - 允许 RDS Custom 通过 Systems Manager 在数据库实例上进行通信、执行自动化和维护代理任务。
- `ec2`、`s3` - 允许 RDS Custom 在提供时间点还原功能的数据库实例上执行备份操作。
- `secretsmanager` - 允许 RDS Custom 管理由 RDS Custom 创建的数据库实例特定密钥。

- `cloudwatch`、`logs` - 允许 RDS Custom 通过 CloudWatch 代理将数据库实例指标和日志上传到 CloudWatch。
- `events`、`sqs` - 允许 RDS Custom 发送和接收有关数据库实例的状态信息。
- `kms` - 允许 RDS Custom 使用特定于实例的 KMS 密钥对 RDS Custom 管理的密钥和 S3 对象进行加密。

有关此策略的更多信息，包括 JSON 策略文档，请参阅《Amazon 托管式策略参考指南》中的 AmazonRDSCustom实例ProfileRolePolicy。

Amazon 托管式策略：AmazonRDSPreviewServiceRolePolicy

您不应将 `AmazonRDSPreviewServiceRolePolicy` 附加到自己的 IAM 实体。此策略附加到一个服务相关角色，该角色支持 Amazon RDS 代表 RDS 数据库资源调用 Amazon 服务。有关更多信息，请参阅 Amazon RDS 预览版的服务相关角色。

权限详细信息

该策略包含以下权限：

- `ec2` – 可让主体描述可用区和网络资源。
- `secretsmanager` – 允许主体从 Amazon Secrets Manager 获取密钥的值。
- `cloudwatch`、`logs` - 可让 Amazon RDS 通过 CloudWatch 代理将数据库实例指标和日志上传到 CloudWatch。

有关此策略的更多信息，包括 JSON 策略文档，请参阅《Amazon 托管式策略参考指南》中的 AmazonRDSDataFullAccess。

Amazon 托管式策略：AmazonRDSBetaServiceRolePolicy

您不应将 `AmazonRDSBetaServiceRolePolicy` 附加到自己的 IAM 实体。此策略附加到一个服务相关角色，该角色支持 Amazon RDS 代表 RDS 数据库资源调用 Amazon 服务。有关更多信息，请参阅 Amazon RDS 测试版的服务相关角色权限。

权限详细信息

该策略包含以下权限：

- `ec2` - 可让 Amazon RDS 在提供时间点还原功能的数据库实例上执行备份操作。

- `secretsmanager` - 可让 Amazon RDS 管理由 Amazon RDS 创建的数据库实例特定密钥。
- `cloudwatch`、`logs` - 可让 Amazon RDS 通过 CloudWatch 代理将数据库实例指标和日志上传到 CloudWatch。

有关此策略的更多信息，包括 JSON 策略文档，请参阅《Amazon 托管式策略参考指南》中的 AmazonRDSBetaServiceRolePolicy。

Amazon 托管式策略的 Amazon RDS 更新

查看有关 Amazon RDS 的 Amazon 托管式策略的更新的详细信息（自从此服务开始跟踪这些更改）。有关此页面更改的自动提示，请订阅 Amazon RDS 文档历史记录页面上的 RSS 源。

更改	描述	日期
Amazon RDS Custom 的服务相关角色权限 – 对现有策略的更新	Amazon RDS 为 AWSServiceRoleForRDSCustom 服务相关角色的 AmazonRDSCustomServiceRolePolicy 添加了新权限。这些权限可让 RDS Custom 管理 EC2 密钥对，并支持 RDS Custom 与 Amazon SQS 集成。有关更多信息，请参阅 Amazon RDS Custom 的服务相关角色权限。	2025 年 3 月 25 日
Amazon 托管式策略：AmazonRDSCustom实例ProfileRolePolicy - 对现有策略的更新	Amazon RDS 向托管式策略 AmazonRDSCustom InstanceProfileRolePolicy 添加了新的权限，以支持在 RDS Custom 实例上使用 RDS Custom 托管式密钥。有关更多信息，请参阅 Amazon 托管式策略：AmazonRDSCustom实例ProfileRolePolicy。	2025 年 3 月 20 日
Amazon RDS Custom 的服务相关角色权限 - 对现有策略的更新	Amazon RDS 为 AWSServiceRoleForRDSCustom 服务相关角色的 AmazonRDSCustomServiceRolePolicy 添加了新权限。这些新权限支持 RDS Custom 列出和还原 Secrets Manager 密	2025 年 3 月 6 日

更改	描述	日期
	钥。有关更多信息，请参阅 Amazon RDS Custom 的服务相关角色权限。	
Amazon 托管式策略：AmazonRDSPreviewServiceRolePolicy - 对现有策略的更新	Amazon RDS 从 AWSServiceRoleForRDSPreview 服务相关角色的 AmazonRDSPreviewServiceRolePolicy 中删除了 sns:Publish 权限。有关更多信息，请参阅 Amazon 托管式策略：AmazonRDSPreviewServiceRolePolicy。	2024 年 8 月 7 日
Amazon 托管式策略：AmazonRDSBetaServiceRolePolicy - 对现有策略的更新	Amazon RDS 从 AWSServiceRoleForRDSBeta 服务相关角色的 AmazonRDSBetaServiceRolePolicy 中删除了 sns:Publish 权限。有关更多信息，请参阅 Amazon 托管式策略：AmazonRDSBetaServiceRolePolicy。	2024 年 8 月 7 日

更改	描述	日期
Amazon RDS Custom 的服务相关角色权限 - 对现有策略的更新	Amazon RDS 为 `AWSServiceRoleForRDSCustom` 服务相关角色的 `AmazonRDSCustomServiceRolePolicy` 添加了新权限。这些权限可让 RDS Custom 在其它 Amazon Web Services 区域中与 Amazon RDS 服务进行通信并复制 EC2 映像。有关更多信息，请参阅 Amazon RDS Custom 的服务相关角色权限。	2024 年 7 月 18 日
Amazon 托管式策略：AmazonRDSServiceRolePolicy - 对现有策略的更新	Amazon RDS 从 `AWSServiceRoleForRDS` 服务相关角色的 `AmazonRDSServiceRolePolicy` 中删除了 `sns:Publish` 权限。有关更多信息，请参阅 Amazon 托管式策略：AmazonRDSServiceRolePolicy。	2024 年 7 月 2 日
Amazon RDS Custom 的服务相关角色权限 - 对现有策略的更新	Amazon RDS 为 `AWSServiceRoleForRDSCustom` 服务相关角色的 `AmazonRDSCustomServiceRolePolicy` 添加了新权限。这项新权限允许 RDS Custom 将服务角色作为实例配置文件关联到 RDS Custom 实例。有关更多信息，请参阅 Amazon RDS Custom 的服务相关角色权限。	2024 年 4 月 19 日

更改	描述	日期
适用于 Amazon RDS 的 Amazon 托管式策略 - 对现有策略的更新	Amazon RDS 向 AWSServiceRoleForRDSCustom 服务相关角色的 AmazonRDSCustomServiceRolePolicy 添加了一项新权限，允许 RDS Custom for SQL Server 修改底层数据库主机实例类型。RDS 还增加了用于获取数据库主机的实例类型信息的 ec2:DescribeInstanceTypes 权限。有关更多信息，请参阅 适用于 Amazon RDS 的 Amazon 托管式策略。	2024 年 4 月 8 日
适用于 Amazon RDS 的 Amazon 托管式策略：新策略	Amazon RDS 添加了一个名为 AmazonRDSCustom InstanceProfileRolePolicy 的新托管策略，允许 RDS Custom 通过 EC2 实例配置文件执行自动化操作和数据库管理任务。有关更多信息，请参阅 适用于 Amazon RDS 的 Amazon 托管式策略。	2024 年 2 月 27 日
Amazon RDS 的服务相关角色权限 – 对现有策略的更新	Amazon RDS 为 AWSServiceRoleForRDS 服务相关角色的 AmazonRDSServiceRolePolicy 添加了新的语句 ID。 有关更多信息，请参阅 Amazon RDS 的服务相关角色权限。	2024 年 1 月 19 日

更改	描述	日期
适用于 Amazon RDS 的 Amazon 托管式策略 – 对现有策略的更新	`AmazonRDSPerformanceInsightsReadOnly` 和 `AmazonRDSPerformanceInsightsFullAccess` 托管式策略现在将 `Sid`（语句 ID）作为标识符包括在策略语句中。 有关更多信息，请参阅Amazon 托管式策略：AmazonRDSPerformanceInsightsReadOnly和Amazon 托管式策略：AmazonRDSPerformanceInsightsFullAccess	2023 年 10 月 23 日
Amazon RDS 的服务相关角色权限 – 对现有策略的更新	Amazon RDS 为 `AWSServiceRoleForRDSCustom` 服务相关角色的 `AmazonRDSCustomServiceRolePolicy` 添加了新权限。这些新权限允许 RDS Custom for Oracle 创建、修改和删除 EventBridge 托管式规则。 有关更多信息，请参阅 Amazon RDS Custom 的服务相关角色权限。	2023 年 9 月 20 日

更改	描述	日期
适用于 Amazon RDS 的 Amazon 托管式策略 – 对现有策略的更新	Amazon RDS 将新权限添加到 `AmazonRDSFullAccess` 托管式策略。这些权限允许您生成、查看和删除一段时间内的性能分析报告。 有关为性能详情配置访问策略的更多信息，请参阅为 Performance Insights 配置访问策略。	2023 年 8 月 17 日
适用于 Amazon RDS 的 Amazon 托管式策略 – 新策略和对现有策略的更新	Amazon RDS 向 `AmazonRDSPerformanceInsightsReadOnly` 托管式策略和名为 `AmazonRDSPerformanceInsightsFullAccess` 的新托管式策略添加了新权限。这些权限允许您分析一段时间内的性能详情、查看分析结果和建议以及删除报告。 有关为性能详情配置访问策略的更多信息，请参阅为 Performance Insights 配置访问策略。	2023 年 8 月 16 日

更改	描述	日期
Amazon RDS 的服务相关角色权限 – 对现有策略的更新	Amazon RDS 为 AWSServiceRoleForRDSCustom 服务相关角色的 AmazonRDSCustomServiceRolePolicy 添加了新权限。这些新权限允许 RDS Custom for Oracle 使用数据库快照。 有关更多信息，请参阅 Amazon RDS Custom 的服务相关角色权限。	2023 年 6 月 23 日
Amazon RDS 的服务相关角色权限 – 对现有策略的更新	Amazon RDS 为 AWSServiceRoleForRDSCustom 服务相关角色的 AmazonRDSCustomServiceRolePolicy 添加了新权限。这些新权限允许 RDS Custom for Oracle 使用数据库快照。 有关更多信息，请参阅 Amazon RDS Custom 的服务相关角色权限。	2023 年 6 月 23 日
Amazon RDS 的服务相关角色权限 – 对现有策略的更新	Amazon RDS 为 AWSServiceRoleForRDSCustom 服务相关角色的 AmazonRDSCustomServiceRolePolicy 添加了新权限。这些新权限允许 RDS Custom 创建网络接口。 有关更多信息，请参阅 Amazon RDS Custom 的服务相关角色权限。	2023 年 5 月 30 日

更改	描述	日期
Amazon RDS 的服务相关角色权限 – 对现有策略的更新	Amazon RDS 为 AWSServiceRoleForRDSCustom 服务相关角色的 AmazonRDSCustomServiceRolePolicy 添加了新权限。这些新权限允许 RDS Custom 调用 Amazon EBS 来检查存储限额。 有关更多信息，请参阅 Amazon RDS Custom 的服务相关角色权限。	2023 年 4 月 18 日
Amazon RDS 的服务相关角色权限 – 对现有策略的更新	Amazon RDS Custom 为 AWSServiceRoleForRDSCustom 服务相关角色的 AmazonRDSCustomServiceRolePolicy 添加了新权限，以便与 Amazon SQS 集成。RDS Custom 需要与 Amazon SQS 集成，才能在客户账户中创建和管理 SQS 队列。SQS 队列名称遵循格式 do-not-delete-rds-custom-[identifier] 并标记为 Amazon RDS Custom。还添加了 ec2:CreateSnapshot 的权限，以允许 RDS Custom 为连接到实例的卷创建备份。 有关更多信息，请参阅 Amazon RDS Custom 的服务相关角色权限。	2023 年 4 月 6 日

更改	描述	日期
适用于 Amazon RDS 的 Amazon 托管式策略 – 对现有策略的更新	Amazon RDS 向 `AmazonRDSFullAccess` 和 `AmazonRDSReadOnlyAccess` 添加了新的 Amazon CloudWatch 命名空间 `ListMetrics` 。 Amazon RDS 需要此命名空间来列出特定的资源使用情况指标。 有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的管理对您的 CloudWatch 资源的访问权限的概述。	2023 年 4 月 4 日
适用于 Amazon RDS 的 Amazon 托管式策略 – 对现有策略的更新	Amazon RDS 向 `AmazonRDSFullAccess` 和 `AmazonRDSReadOnlyAccess` 托管式策略添加了新权限，以允许在 RDS 控制台中显示 Amazon DevOps Guru 调查发现。 需要此权限才能显示 DevOps Guru 调查发现。 有关更多信息，请参阅 Amazon RDS 对 Amazon 托管策略的更新。	2023 年 3 月 30 日

更改	描述	日期
Amazon RDS 的服务相关角色权限 – 对现有策略的更新	Amazon RDS 为 `AWSServiceRoleForRDS` 服务相关角色的 `AmazonRDSServiceRolePolicy` 添加了新权限，以便与 Amazon Secrets Manager 集成。RDS 需要与 Secrets Manager 集成才能在 Secrets Manager 中管理主用户密码。该密钥使用保留的命名惯例并限制客户更新。 有关更多信息，请参阅 使用 Amazon RDS 和 Amazon Secrets Manager 管理密码。	2022 年 12 月 22 日
Amazon RDS 的服务相关角色权限 – 对现有策略的更新	Amazon RDS 为 `AWSServiceRoleForRDSCustom` 服务相关角色的 `AmazonRDSCustomServiceRolePolicy` 添加了新权限。RDS Custom 支持数据库集群。策略中的这些新权限允许 RDS Custom 代表您的数据库集群调用 Amazon Web Services 服务。 有关更多信息，请参阅 Amazon RDS Custom 的服务相关角色权限。	2022 年 11 月 9 日

更改	描述	日期
Amazon RDS 的服务相关角色权限 – 对现有策略的更新	Amazon RDS 为 AWSServiceRoleForRDS 服务相关角色添加了新权限，以便与 Amazon Secrets Manager 集成。 需要与 Secrets Manager 集成，SQL Server Reporting Services（SSRS）电子邮件才能在 RDS 上运行。SSRS 电子邮件代表客户创建一个密钥。该密钥使用保留的命名惯例并限制客户更新。 有关更多信息，请参阅 使用 SSRS 电子邮件发送报告。	2022 年 8 月 26 日
Amazon RDS 的服务相关角色权限 – 对现有策略的更新	Amazon RDS 向 PutMetricData 的 AmazonRDSPreviewServiceRolePolicy 添加了新的 Amazon CloudWatch 命名空间。 Amazon RDS 需要此命名空间来发布资源用量指标。 有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的使用条件键限制对 CloudWatch 命名空间的访问。	2022 年 6 月 7 日

更改	描述	日期
Amazon RDS 的服务相关角色权限 – 对现有策略的更新	Amazon RDS 向 `PutMetricData` 的 `AmazonRDSBetaServiceRolePolicy` 添加了新的 Amazon CloudWatch 命名空间。 Amazon RDS 需要此命名空间来发布资源用量指标。 有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的使用条件键限制对 CloudWatch 命名空间的访问。	2022 年 6 月 7 日
Amazon RDS 的服务相关角色权限 – 对现有策略的更新	Amazon RDS 向 `PutMetricData` 的 `AWSServiceRoleForRDS` 添加了新的 Amazon CloudWatch 命名空间。 Amazon RDS 需要此命名空间来发布资源用量指标。 有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的使用条件键限制对 CloudWatch 命名空间的访问。	2022 年 4 月 22 日

更改	描述	日期
Amazon RDS 的服务相关角色权限 – 对现有策略的更新	Amazon RDS 向 `AWSServiceRoleForRDS` 服务相关角色添加了新权限，以管理客户拥有的 IP 池和本地网关路由表 (LGW-RTB) 的权限。 RDS on Outposts 需要这些权限，才能跨 Outposts 的本地网络执行多可用区复制。 有关更多信息，请参阅 使用 Amazon RDS on Amazon Outposts 的多可用区部署。	2022 年 4 月 19 日
基于身份的策略 – 对现有策略的更新	Amazon RDS 向 `AmazonRDSFullAccess` 托管式策略添加了一项新权限，以描述对 LGW-RTB 的权限。 需要此权限来描述 RDS on Outposts 跨 Outposts 的本地网络执行多可用区复制所需的权限。 有关更多信息，请参阅 使用 Amazon RDS on Amazon Outposts 的多可用区部署。	2022 年 4 月 19 日

更改	描述	日期
适用于 Amazon RDS 的 Amazon 托管式策略：新策略	Amazon RDS 添加了名为 `AmazonRDSPerformanceInsightsReadOnly` 的新托管式策略，以允许 Amazon RDS 代表数据库实例调用 Amazon 服务。 有关为性能详情配置访问策略的更多信息，请参阅为 Performance Insights 配置访问策略。	2022 年 3 月 10 日
Amazon RDS 的服务相关角色权限 – 对现有策略的更新	Amazon RDS 向 `PutMetricData` 的 `AWSServiceRoleForRDS` 添加了新的 Amazon CloudWatch 命名空间。 要发布 CloudWatch 指标，Amazon DocumentDB（MongoDB 兼容）和 Amazon Neptune 需要这些命名空间。 有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的使用条件键限制对 CloudWatch 命名空间的访问。	2022 年 3 月 4 日
Amazon RDS Custom 的服务相关角色权限：新策略	Amazon RDS 添加了名为 `AWSServiceRoleForRDSCustom` 的新服务相关角色，以允许 RDS Custom 代表数据库实例调用 Amazon Web Services 服务。	2021 年 10 月 26 日

更改	描述	日期
Amazon RDS 开始跟踪更改	Amazon RDS 为其 Amazon 托管式策略开启了跟踪更改。	2021 年 10 月 26 日

防范跨服务混淆代理问题

混淆代理问题是一个安全问题，即没有执行操作权限的实体可能会迫使更具权限的实体执行该操作。在 Amazon 中，跨服务模拟可能会导致混淆代理问题。

一个服务（呼叫服务）调用另一项服务（所谓的服务）时，可能会发生跨服务模拟。可以操纵调用服务来使用其权限，以不应该具有的访问权限对另一个客户的资源进行操作。为防止这种情况，Amazon 提供可帮助您保护所有服务数据的工具，而这些服务中的服务主体有权限访问账户中的资源。有关更多信息，请参阅《IAM 用户指南》中的混淆代理人问题。

要限制 Amazon RDS 为其他服务提供对资源的权限，我们建议在资源策略中使用 `aws:SourceArn` 和 `aws:SourceAccount` 全局条件上下文键。

在某些情况下，`aws:SourceArn` 值不包含账户 ID，例如将 Amazon Resource Name (ARN) 用于 Amazon S3 存储桶时。在这些情况下，请确保使用两个全局条件上下文键来限制权限。在某些情况下，您可以使用两个全局条件上下文键并让 `aws:SourceArn` 值包含账户 ID。在这些情况下，当 `aws:SourceAccount` 值和 `aws:SourceArn` 值中的账户使用相同策略语句时，确保二者使用相同的账户 ID。如果您只希望将一个资源与跨服务访问相关联，请使用 `aws:SourceArn`。如果您想允许指定 Amazon 账户中的任何资源与跨服务使用相关联，请使用 `aws:SourceAccount`。

确保 `aws:SourceArn` 的值是 Amazon RDS 资源类型的 ARN。有关更多信息，请参阅 Amazon RDS 中的 Amazon 资源名称（ARN）。

防范混淆代理问题最有效的方法是使用 `aws:SourceArn` 全局条件上下文键和资源的完整 ARN。在某些情况下，您可能不知道资源的完整 ARN，或者您可能正在指定多个资源。在这些情况下，请将带通配符 (*) 的 `aws:SourceArn` 全局上下文条件键用于 ARN 的未知部分。例如，`arn:aws:rds:*:123456789012:*`。

以下示例演示如何使用 Amazon RDS 中的 `aws:SourceArn` 和 `aws:SourceAccount` 全局条件上下文键来防范混淆代理问题。

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "ConfusedDeputyPreventionExamplePolicy",
    "Effect": "Allow",
    "Principal": {
      "Service": "rds.amazonaws.com"
    },
    "Action": "sts:AssumeRole",
```

```
    "Condition": {
      "ArnLike": {
        "aws:SourceArn": "arn:aws:rds:us-east-1:123456789012:db:mydbinstance"
      },
      "StringEquals": {
        "aws:SourceAccount": "123456789012"
      }
    }
  }
}
```

有关使用 `aws:SourceArn` 和 `aws:SourceAccount` 全局条件上下文键的更多策略示例，请参阅以下部分：

- 授予向 Amazon SNS 主题发布通知的权限。
- 为本机备份和还原手动创建 IAM 角色
- 为 SQL Server 数据库实例设置 Windows 身份验证
- 将 RDS for SQL Server 与 S3 集成的先决条件
- 手动为 SQL Server Audit 创建 IAM 角色
- 为与 Amazon S3 集成的 RDS for Oracle 配置 IAM 权限
- 设置 Amazon S3 存储桶的访问权限（PostgreSQL 导入）
- 设置 Amazon S3 存储桶的访问权限（PostgreSQL 导出）

适用于 MariaDB、MySQL 和 PostgreSQL 的 IAM 数据库身份验证

您可以使用 Amazon Identity and Access Management（IAM）数据库身份验证对数据库实例进行身份验证。IAM 数据库身份验证适用于 MariaDB、MySQL 和 PostgreSQL。利用此身份验证方法，您在连接到数据库实例时将无需使用密码。而是使用身份验证令牌。

身份验证令牌 是 Amazon RDS 根据请求生成的唯一字符串。身份验证令牌是使用 Amazon 签名版本 4 生成的。每个令牌的使用期限为 15 分钟。您无需将用户凭证存储在数据库中，因为身份验证是使用 IAM 进行外部管理的。此外，您仍可使用标准数据库身份验证。令牌仅用于身份验证，建立后不会影响会话。

IAM 数据库身份验证具有以下优势：

- 数据库的出站和进站网络流量是使用安全套接字层 (SSL) 或传输层安全性 (TLS) 加密的。有关将 SSL/TLS 与 Amazon RDS 一起使用的更多信息，请参阅使用 SSL/TLS 加密与数据库实例或集群的连接。
- 您可以使用 IAM 集中管理对数据库资源的访问，而不是单独管理对每个数据库实例的访问。
- 对于在 Amazon EC2 上运行的应用程序，您可以使用 EC2 实例特定的配置文件凭证访问数据库以提高安全性，而不是使用密码。

一般来说，如果应用程序每秒创建的连接少于 200 个，而且您不想直接在应用程序代码中管理用户名和密码，请考虑使用 IAM 数据库身份验证。

Amazon Web Services（Amazon）JDBC 驱动程序支持 IAM 数据库身份验证。有关更多信息，请参阅 Amazon Web Services（Amazon）JDBC 驱动程序 GitHub 存储库中的 Amazon IAM 身份验证插件。

Amazon Web Services（Amazon）Python 驱动程序支持 IAM 数据库身份验证。有关更多信息，请参阅 Amazon Web Services（Amazon）Python 驱动程序 GitHub 存储库中的 Amazon IAM 身份验证插件。

浏览以下主题，学习设置 IAM 用于数据库身份验证的过程：

- 启用和禁用 IAM 数据库身份验证
- 创建和使用适用于 IAM 数据库访问的 IAM 策略
- 使用 IAM 身份验证创建数据库账户
- 使用 IAM 身份验证连接到数据库实例

区域和版本可用性

功能可用性和支持因每个数据库引擎的特定版本而异。有关适用于 Amazon RDS 和 IAM 数据库身份验证的引擎、版本和区域可用性的更多信息，请参阅支持 Amazon RDS 中 IAM 数据库身份验证功能的区域和数据库引擎。

CLI 和开发工具包支持

IAM 数据库身份验证可用于 Amazon CLI 以及以下特定于语言的 Amazon 软件开发工具包：

- 适用于 .NET 的 Amazon SDK
- 适用于 C++ 的 Amazon SDK
- 适用于 Go 的 Amazon SDK
- 适用于 Java 的 Amazon SDK
- 适用于 JavaScript 的 Amazon SDK
- 适用于 PHP 的 Amazon SDK
- 适用于 Python (Boto3) 的 Amazon SDK
- 适用于 Ruby 的 Amazon SDK

IAM 数据库身份验证的限制

使用 IAM 数据库身份验证时，以下限制适用：

- IAM 数据库身份验证会限制连接数，即每秒 200 个连接。

  使用同一身份验证令牌的连接不受限制。建议您尽可能重复使用身份验证令牌。
- 目前，IAM 数据库身份验证并不支持所有的全局条件上下文键。

  有关全局条件上下文键的更多信息，请参阅《IAM 用户指南》中的 Amazon 全局条件上下文键。
- 对于 PostgreSQL，如果将 IAM 角色（`rds_iam`）添加到某个用户（包括 RDS 主用户），IAM 身份验证将优先于密码身份验证，因此该用户必须以 IAM 用户身份登录。
- 对于 PostgreSQL，Amazon RDS 不支持同时启用 IAM 和 Kerberos 身份验证方法。
- 对于 PostgreSQL，您不能使用 IAM 身份验证来建立复制连接。
- 不能使用自定义 Route 53 DNS 记录代替数据库实例端点来生成身份验证令牌。
- CloudWatch 和 CloudTrail 不记录 IAM 身份验证。这些服务不会跟踪授权 IAM 角色启用数据库连接的 `generate-db-auth-token` API 调用。

- IAM 数据库身份验证需要数据库实例上的计算资源。您的数据库上必须有 300 到 1000 MiB 的额外内存，才能实现可靠的连接。要查看工作负载所需的内存，请在启用 IAM 数据库身份验证之前和之后，比较增强型监控进程列表中 RDS 进程的 RES 列。请参阅在 RDS 控制台中查看操作系统指标。

  如果您使用的是可突增类实例，请通过将其它参数使用的内存（如缓冲区和缓存）减少相同的量，从而避免内存耗尽。
- 对于 RDS for MySQL，您不能对使用 IAM 身份验证配置的数据库用户使用基于密码的身份验证。
- 对于任何引擎，RDS on Outposts 不支持 IAM 数据库身份验证。

IAM 数据库身份验证建议

在使用 IAM 数据库身份验证时，建议使用以下方法：

- 当您的应用程序每秒需要少于 200 个新的 IAM 数据库身份验证连接时，请使用 IAM 数据库身份验证。

  使用 Amazon RDS 的数据库引擎不会对每秒的身份验证尝试次数施加任何限制。不过，在使用 IAM 数据库身份验证时，您的应用程序必须生成身份验证令牌。之后，您的应用程序将使用该令牌连接到数据库实例。如果超出每秒的最大新连接数限制，则 IAM 数据库身份验证的额外开销可能会导致连接受到限制。

  考虑在应用程序中使用连接池来减少持续的连接创建。这可以减少 IAM 数据库身份验证的开销，并允许应用程序重用现有连接。或者，考虑对这些使用案例使用 RDS 代理。RDS 代理有额外费用。请参阅 RDS 代理定价。
- IAM 数据库身份验证令牌的大小取决于许多因素，包括 IAM 标签的数量、IAM 服务策略、ARN 长度以及其他 IAM 和数据库属性。此令牌的最小大小通常约为 1KB，但可以更大。由于使用 IAM 身份验证将此令牌用作数据库的连接字符串中的密码，因此，您应确保您的数据库驱动程序（例如 ODBC）和/或任何工具不会因其大小而限制或以其他方式截断该令牌。截断的令牌将导致数据库和 IAM 执行的身份验证失败。
- 如果您在创建 IAM 数据库身份验证令牌时使用临时凭证，则在使用 IAM 数据库身份验证令牌发出连接请求时，临时凭证必须仍然有效。

不支持的 Amazon 全局条件上下文键

IAM 数据库身份验证不支持 Amazon 全局条件上下文键的以下子集。

- `aws:Referer`

- `aws:SourceIp`
- `aws:SourceVpc`
- `aws:SourceVpce`
- `aws:UserAgent`
- `aws:VpcSourceIp`

有关更多信息，请参阅《IAM 用户指南》中的 Amazon 全局条件上下文键。

启用和禁用 IAM 数据库身份验证

默认情况下，已对数据库实例禁用 IAM 数据库身份验证。您可以使用 Amazon Web Services Management Console、Amazon CLI 或 API 启用或禁用 IAM 数据库身份验证。

您可以在执行以下操作之一时启用 IAM 数据库身份验证：

- 要创建启用 IAM 数据库身份验证的新数据库实例，请参阅 创建 Amazon RDS 数据库实例。
- 要修改数据库实例以启用 IAM 数据库身份验证，请参阅 修改 Amazon RDS 数据库实例。
- 要从启用了 IAM 数据库身份验证的快照还原数据库实例，请参阅还原到数据库实例。
- 要将数据库实例还原到已启用 IAM 数据库身份验证的某个时间点，请参阅 将 Amazon RDS 的数据库实例还原到指定时间。

PostgreSQL 数据库实例的 IAM 身份验证要求 SSL 值为 1。如果 SSL 值为 0，则无法为 PostgreSQL 数据库实例启用 IAM 身份验证。如果为 PostgreSQL 数据库实例启用了 IAM 身份验证，则无法将 SSL 值更改为 0。

控制台

每个创建或修改工作流程都有一个数据库身份验证部分，您可以在其中启用或禁用 IAM 数据库身份验证。在该部分中，选择密码和 IAM 数据库身份验证以启用 IAM 数据库身份验证。

为现有数据库实例启用或禁用 IAM 数据库身份验证

1. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。
3. 选择要修改的数据库实例。

> **Note**
>
> 确保数据库实例与 IAM 身份验证兼容。在区域和版本可用性中检查兼容性要求。

4. 选择修改。
5. 在 Database authentication (数据库身份验证) 部分中，选择 Password and IAM database authentication (密码和 IAM 数据库身份验证)，启用 IAM 数据库身份验证。选择密码身份验证或密码和 Kerberos 身份验证以禁用 IAM 身份验证。
6. 也可以选择启用向 CloudWatch Logs 发布 IAM 数据库身份验证日志。在日志导出下，选择 iam-db-auth-error log 选项。将您的日志发布到 CloudWatch Logs 会消耗存储空间，并且您需要为该存储空间支付费用。请务必删除所有不再需要的 CloudWatch Logs。
7. 选择 Continue (继续)。
8. 要立即应用更改，请在修改计划部分中选择立即。
9. 选择修改数据库实例 。

Amazon CLI

要使用 Amazon CLI 创建采用 IAM 身份验证的新数据库实例，请使用 `create-db-instance` 命令。指定 `--enable-iam-database-authentication` 选项，如以下示例中所示。

```
aws rds create-db-instance \
    --db-instance-identifier mydbinstance \
    --db-instance-class db.m3.medium \
    --engine MySQL \
    --allocated-storage 20 \
    --master-username masterawsuser \
    --manage-master-user-password \
    --enable-iam-database-authentication
```

要更新现有的数据库实例以使用或不使用 IAM 身份验证，请使用 Amazon CLI 命令 `modify-db-instance`。根据需要指定 `--enable-iam-database-authentication` 或 `--no-enable-iam-database-authentication` 选项。

> **Note**
>
> 确保数据库实例与 IAM 身份验证兼容。在区域和版本可用性中检查兼容性要求。

默认情况下，Amazon RDS 在下一个维护时段执行修改。如果您要覆盖该选项并尽快启用 IAM 数据库身份验证，请使用 `--apply-immediately` 参数。

以下示例说明了如何立即为现有数据库实例启用 IAM 身份验证。

```
aws rds modify-db-instance \
    --db-instance-identifier mydbinstance \
    --apply-immediately \
    --enable-iam-database-authentication
```

如果您将还原数据库实例，请使用下列 Amazon CLI 命令之一：

- [restore-db-instance-to-point-in-time]
- [restore-db-instance-from-db-snapshot]

IAM 数据库身份验证设置默认为源快照的设置。要更改此设置，请根据需要设置 `--enable-iam-database-authentication` 或 `--no-enable-iam-database-authentication` 选项。

RDS API

要使用 API 创建采用 IAM 身份验证的新数据库实例，请使用 API 操作 `CreateDBInstance`。将 `EnableIAMDatabaseAuthentication` 参数设置为 `true`。

要更新现有的数据库实例以采用 IAM 身份验证，请使用 API 操作 `ModifyDBInstance`。将 `EnableIAMDatabaseAuthentication` 参数设置为 `true` 可启用 IAM 身份验证，将其设置为 `false` 可禁用 IAM 身份验证。

> **Note**
> 确保数据库实例与 IAM 身份验证兼容。在区域和版本可用性中检查兼容性要求。

如果您将还原数据库实例，请使用以下 API 操作之一：

- `RestoreDBInstanceFromDBSnapshot`
- `RestoreDBInstanceToPointInTime`

IAM 数据库身份验证设置默认为源快照的设置。要更改此设置，请将 `EnableIAMDatabaseAuthentication` 参数设置为 `true` 以启用 IAM 身份验证，或将其设置为 `false` 以禁用 IAM 身份验证。

创建和使用适用于 IAM 数据库访问的 IAM 策略

要允许用户或角色连接到数据库实例，您必须创建 IAM policy。之后，将策略附加到权限集或角色。

> **Note**
>
> 要了解有关 IAM 策略的更多信息，请参阅 Amazon RDS 的 Identity and Access Management。

以下示例策略允许用户使用 IAM 数据库身份验证连接到数据库实例。

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": [
             "rds-db:connect"
         ],
         "Resource": [
             "arn:aws:rds-db:us-east-2:1234567890:dbuser:db-ABCDEFGHIJKL01234/db_user"
         ]
      }
   ]
}
```

> **Important**
>
> 具有管理员权限的用户即使在 IAM policy 中没有显式权限，也可以访问数据库实例。如果您希望限制管理员访问数据库实例，您可以创建具有合适的较低权限的 IAM 角色，并将其分配给管理员。

> **Note**
>
> 请不要将 `rds-db:` 前缀与以 `rds:` 开头的其他 RDS API 操作前缀混淆。仅在 IAM 数据库身份验证使用 `rds-db:` 前缀和 `rds-db:connect` 操作。它们在任何其他上下文中无效。

示例策略包含带以下元素的单个语句：

- `Effect` – 指定 `Allow` 以授予数据库实例的访问权限。如果您没有显式允许访问，则默认情况下将拒绝访问。
- `Action` – 指定 `rds-db:connect` 以允许连接到数据库实例。
- `Resource` – 指定 Amazon Resource Name (ARN) 以描述一个数据库实例中的一个数据库账户。ARN 格式如下所示。

  ```
  arn:aws:rds-db:region:account-id:dbuser:DbiResourceId/db-user-name
  ```

  在此格式中，替换以下内容：

  - *region* 是数据库实例所在的 Amazon 区域。在示例策略中，Amazon 区域为 `us-east-2`。
  - *account-id* 是数据库实例的 Amazon 账号。在示例策略中，账号为 `1234567890`。该用户必须位于与数据库实例的账户相同的账户中。

    要执行跨账户存取，请在数据库实例的账户中使用上面显示的策略创建 IAM 角色，并允许其他账户代入该角色。
  - *DbiResourceId* 是数据库实例的标识符。此标识符对 Amazon 区域是唯一的，并且绝不会更改。在示例策略中，标识符为 `db-ABCDEFGHIJKL01234`。

    要在适用于 Amazon RDS 的 Amazon Web Services Management Console 中查找数据库实例资源 ID，请选择数据库实例来查看其详细信息。然后，选择配置选项卡。Resource ID (资源 ID) 将显示在 Configuration (配置) 部分中。

    或者，您可以使用 Amazon CLI 命令列出当前 Amazon 区域中所有数据库实例的标识符和资源 ID，如下所示。

    ```
    aws rds describe-db-instances --query "DBInstances[*].
    [DBInstanceIdentifier,DbiResourceId]"
    ```

如果您使用的是 Amazon Aurora，请指定 DbClusterResourceId 而不是 DbiResourceId。有关更多信息，请参阅 Amazon Aurora 用户指南中的[创建和使用适用于 IAM 数据库访问的 IAM 策略](#)。

> **Note**
>
> 如果要通过 RDS 代理连接到数据库，请指定代理资源 ID，如 prx-ABCDEFGHIJKL01234。有关将 IAM 数据库身份验证与 RDS 代理结合使用的信息，请参阅 [使用 IAM 身份验证连接到代理](#)。

- *db-user-name* 是与 IAM 身份验证关联的数据库账户的名称。在示例策略中，数据库账户为 db_user。

您可以构造其他 ARN 以支持多种访问模式。以下策略允许访问一个数据库实例中的两个不同的数据库账户。

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": [
             "rds-db:connect"
         ],
         "Resource": [
             "arn:aws:rds-db:us-east-2:123456789012:dbuser:db-ABCDEFGHIJKL01234/
jane_doe",
             "arn:aws:rds-db:us-east-2:123456789012:dbuser:db-ABCDEFGHIJKL01234/
mary_roe"
         ]
      }
   ]
}
```

以下策略使用“*”字符匹配特定 Amazon 账户和 Amazon 区域的所有数据库实例和数据库账户。

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "rds-db:connect"
            ],
            "Resource": [
                "arn:aws:rds-db:us-east-2:1234567890:dbuser:*/*"
            ]
        }
    ]
}
```

以下策略匹配特定 Amazon 账户和 Amazon 区域的所有数据库实例。不过，该策略仅允许访问具有 jane_doe 数据库账户的数据库实例。

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": [
             "rds-db:connect"
         ],
         "Resource": [
             "arn:aws:rds-db:us-east-2:123456789012:dbuser:*/jane_doe"
         ]
      }
   ]
}
```

用户或角色只能访问数据库用户有权访问的那些数据库。例如，假设数据库实例具有一个名为 dev 的数据库，以及另一个名为 test 的数据库。如果数据库用户 jane_doe 只能访问 dev，则与 jane_doe 用户一起访问数据库实例的任何用户或角色也只能访问 dev。此访问限制还适用于其他数据库对象 (如表、视图等)。

管理员必须创建 IAM policy，以便为实体授予权限，从而对其所需的指定资源执行特定的 API 操作。然后，管理员必须将这些策略附加到需要这些权限的权限集或角色。有关策略示例，请参阅"Amazon RDS 的基于身份的策略示例"。

将 IAM policy 附加到权限集或角色

在创建允许数据库身份验证的 IAM policy 之后，您需要将该策略附加到权限集或角色。有关此主题的教程，请参阅 IAM 用户指南 中的创建和附加您的第一个客户托管策略。

在演练此教程时，您可以使用此部分中显示的策略示例之一作为起点并根据您的需求进行定制。在教程结束时，您将有一个权限集，该权限集附加了一个可利用 `rds-db:connect` 操作的策略。

> **Note**
>
> 您可以将多个权限集或角色映射到同一数据库用户账户。例如，假设您的 IAM 策略指定了以下资源 ARN。
>
> ```
> arn:aws:rds-db:us-east-2:123456789012:dbuser:db-12ABC34DEFG5HIJ6KLMNOP78QR/
> jane_doe
> ```
>
> 如果将该策略附加到 Jane、Bob 和 Diego，则其中的每个用户都可以使用 `jane_doe` 数据库账户连接到指定的数据库实例。

使用 IAM 身份验证创建数据库账户

在使用 IAM 数据库身份验证时，您不需要为创建的用户账户分配数据库密码。如果删除映射到某个数据库账户的用户，则还应使用 `DROP USER` 语句删除该数据库账户。

> **Note**
>
> 用于 IAM 身份验证的用户名必须与数据库中用户名的大小写相匹配。

主题

- 对 MariaDB 和 MySQL 使用 IAM 身份验证
- 对 PostgreSQL 使用 IAM 身份验证

对 MariaDB 和 MySQL 使用 IAM 身份验证

对于 MariaDB 和 MySQL，身份验证由 `AWSAuthenticationPlugin` 处理，这是 Amazon 提供的一个插件，可以与 IAM 无缝协作以验证您的用户的身份。以主用户或其他可以创建用户和授予权限的用户身份连接到数据库实例。连接后，发出 `CREATE USER` 语句，如以下示例中所示。

```
CREATE USER 'jane_doe' IDENTIFIED WITH AWSAuthenticationPlugin AS 'RDS';
```

`IDENTIFIED WITH` 子句允许 MariaDB 和 MySQL 使用 `AWSAuthenticationPlugin` 对数据库账户 (jane_doe) 进行身份验证。`AS 'RDS'` 子句是指身份验证方法。确保指定的数据库用户名与 IAM 数据库访问策略中的资源相同。有关更多信息，请参阅"创建和使用适用于 IAM 数据库访问的 IAM 策略"。

> **Note**
>
> 如果看到以下消息，则意味着 Amazon 提供的插件对当前的数据库实例不可用。
>
> `ERROR 1524 (HY000): Plugin 'AWSAuthenticationPlugin' is not loaded`
>
> 要纠正该错误，请确认您使用支持的配置，并且在数据库实例上启用了 IAM 数据库身份验证。有关更多信息，请参阅"区域和版本可用性"和"启用和禁用 IAM 数据库身份验证"。

在使用 `AWSAuthenticationPlugin` 创建一个账户后，可以像管理其他数据库账户一样管理此账户。例如，您可以使用 `GRANT` 语句和 `REVOKE` 语句修改账户权限，或使用 `ALTER USER` 语句修改各种账户属性。

使用 IAM 时，数据库网络流量使用 SSL/TLS 进行加密。要允许 SSL 连接，请使用以下命令修改用户账户。

```
ALTER USER 'jane_doe'@'%' REQUIRE SSL;
```

对 PostgreSQL 使用 IAM 身份验证

要在 PostgreSQL 中使用 IAM 身份验证，请以主用户或其他可以创建用户和授予权限的用户身份连接到数据库实例。连接后，创建数据库用户，然后向其授予 `rds_iam` 角色，如以下示例中所示。

```
CREATE USER db_userx;
```

```
GRANT rds_iam TO db_userx;
```

确保指定的数据库用户名与 IAM 数据库访问策略中的资源相同。有关更多信息，请参阅 创建和使用适用于 IAM 数据库访问的 IAM 策略。您必须授予 `rds_iam` 角色才能使用 IAM 身份验证。也可以使用该角色的嵌套成员资格或间接授权。

使用 IAM 身份验证连接到数据库实例

对于 IAM 数据库身份验证，您可以在连接到数据库实例时使用身份验证令牌。身份验证令牌 是您使用的一个字符串而不是密码。在生成身份验证令牌后，将在 15 分钟后失效。如果您尝试使用过期的令牌进行连接，则连接请求将被拒绝。

必须使用 Amazon 签名版本 4 为每个身份验证令牌附带一个有效签名。（有关更多信息，请参阅《Amazon Web Services 一般参考》中的签名版本 4 签名流程。）Amazon CLI 和 Amazon 软件开发工具包（例如 适用于 Java 的 Amazon SDK 或者 适用于 Python (Boto3) 的 Amazon SDK）可以自动对您创建的每个令牌进行签名。

从其他Amazon服务（如 Amazon Lambda）连接到 Amazon RDS 时，您可以使用身份验证令牌。通过使用令牌，可以避免在代码中放置密码。或者，您可以使用 Amazon 开发工具包以编程方式创建身份验证令牌并为其签名。

在获得签名的 IAM 身份验证令牌后，您可以连接到 Amazon RDS 数据库实例。在下文中，您可以了解如何使用命令行工具或 Amazon 开发工具包（例如 适用于 Java 的 Amazon SDK 或 适用于 Python (Boto3) 的 Amazon SDK）执行该操作。

有关更多信息，请参阅以下博客文章：

- 使用 IAM 身份验证通过 SQL Workbench/J 连接到 Aurora MySQL 或 Amazon RDS for MySQL
- 使用 IAM 身份验证通过 pgAdmin Amazon Aurora PostgreSQL 或 Amazon RDS for PostgreSQL 进行连接

先决条件

以下是使用 IAM 身份验证连接到数据库实例的先决条件：

- 启用和禁用 IAM 数据库身份验证
- 创建和使用适用于 IAM 数据库访问的 IAM 策略
- 使用 IAM 身份验证创建数据库账户

主题

使用 IAM 身份验证和 Amazon 驱动程序连接到数据库实例

借助 Amazon 驱动程序套件，可显著缩短切换和故障转移时间，并支持使用 Amazon Secrets Manager、Amazon Identity and Access Management（IAM）和联合身份进行身份验证。Amazon 驱动程序依靠监控数据库实例状态以及了解实例拓扑，来确定新的写入器。这种方法将切换和故障转移时间缩短到几秒钟，而开源驱动程序的切换和故障转移时间则为几十秒。

有关 Amazon 驱动程序的更多信息，请参阅 RDS for MariaDB、RDS for MySQL 或 RDS for PostgreSQL 数据库实例的相应语言驱动程序。

> **Note**
>
> RDS for MariaDB 支持的唯一功能是使用 Amazon Secrets Manager、Amazon Identity and Access Management（IAM）和联合身份进行身份验证。

通过命令行 Amazon CLI 和 mysql 客户端，使用 IAM 身份验证连接到数据库实例

可以使用 Amazon CLI 和 `mysql` 命令行工具从命令行连接到 Amazon RDS 数据库实例，如下所述。

先决条件

以下是使用 IAM 身份验证连接到数据库实例的先决条件：

- 启用和禁用 IAM 数据库身份验证
- 创建和使用适用于 IAM 数据库访问的 IAM 策略
- 使用 IAM 身份验证创建数据库账户

> **Note**
>
> 有关使用具有 IAM 身份验证的 SQL WorkBench/J 连接到数据库的信息，请参阅博客文章[使用 IAM 身份验证通过 SQL WorkBench/J 连接到 Aurora MySQL 或 Amazon RDS for MySQL](#)。

主题

- [生成 IAM 身份验证令牌](#)
- [连接到数据库实例](#)

生成 IAM 身份验证令牌

以下示例说明了如何使用 Amazon CLI 获取签名的身份验证令牌。

```
aws rds generate-db-auth-token \
   --hostname rdsmysql.123456789012.us-west-2.rds.amazonaws.com \
   --port 3306 \
   --region us-west-2 \
   --username jane_doe
```

在该示例中，参数如下所示：

- `--hostname` – 要访问的数据库实例的主机名
- `--port` – 用于连接到数据库实例的端口号
- `--region` – 在其中运行数据库实例的 Amazon 区域
- `--username` – 要访问的数据库账户

令牌的前几个字符与以下内容类似。

```
rdsmysql.123456789012.us-west-2.rds.amazonaws.com:3306/?
Action=connect&DBUser=jane_doe&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Expires=900...
```

> **Note**
>
> 您不能使用自定义 Route 53 DNS 记录代替数据库实例端点来生成身份验证令牌。

连接到数据库实例

通用连接格式如下所示。

```
mysql --host=hostName --port=portNumber --ssl-ca=full_path_to_ssl_certificate --enable-cleartext-plugin --user=userName --password=authToken
```

参数如下所示：

- `--host` – 要访问的数据库实例的主机名
- `--port` – 用于连接到数据库实例的端口号
- `--ssl-ca` – 包含公有密钥的 SSL 证书文件的完整路径

  有关 MariaDB 的 SSL/TLS 支持的更多信息，请参阅 Amazon RDS 上 MariaDB 数据库实例的 SSL/TLS 支持。

  有关 MySQL 的 SSL/TLS 支持的更多信息，请参阅 Amazon RDS 上 MySQL 数据库实例的 SSL/TLS 支持。

  要下载 SSL 证书，请参阅 使用 SSL/TLS 加密与数据库实例或集群的连接。
- `--enable-cleartext-plugin` – 一个指定 `AWSAuthenticationPlugin` 必须用于此连接的值

  如果您使用 MariaDB 客户端，则无需 `--enable-cleartext-plugin` 选项。
- `--user` – 要访问的数据库账户
- `--password` – 已签名的 IAM 身份验证令牌

身份验证令牌包含几百个字符。很难使用命令行对其进行处理。该问题的解决方式是，将令牌保存到一个环境变量中，然后在连接时使用此变量。以下示例说明了一种执行此解决方法的方式。在该示例中，*/sample_dir/* 是包含公有密钥的 SSL 证书文件的完整路径。

```
RDSHOST="mysqldb.123456789012.us-east-1.rds.amazonaws.com"
TOKEN="$(aws rds generate-db-auth-token --hostname $RDSHOST --port 3306 --region us-west-2 --username jane_doe )"

mysql --host=$RDSHOST --port=3306 --ssl-ca=/sample_dir/global-bundle.pem --enable-cleartext-plugin --user=jane_doe --password=$TOKEN
```

在使用 AWSAuthenticationPlugin 进行连接时，将使用 SSL 保护连接。要进行验证，请在 mysql> 命令提示符处键入以下内容。

```
show status like 'Ssl%';
```

输出中的以下行显示了更多详细信息。

```
+---------------+-------------+
| Variable_name | Value                                                                                                                                          |
+---------------+-------------+
| ...           | ...
| Ssl_cipher    | AES256-SHA                                                                                                                                     |
| ...           | ...
| Ssl_version   | TLSv1.1                                                                                                                                        |
| ...           | ...
+-----------------------------+
```

如果您想通过代理连接到数据库实例，请参阅使用 IAM 身份验证连接到代理。

通过命令行使用 IAM 身份验证连接到数据库实例：Amazon CLI 和 psql 客户端

可以使用 Amazon CLI 和 psql 命令行工具从命令行连接到 Amazon RDS for PostgreSQL 数据库实例，如下所述。

先决条件

以下是使用 IAM 身份验证连接到数据库实例的先决条件：

- 启用和禁用 IAM 数据库身份验证
- 创建和使用适用于 IAM 数据库访问的 IAM 策略
- 使用 IAM 身份验证创建数据库账户

> **Note**
>
> 有关使用具有 IAM 身份验证的 pgAdmin 连接到数据库的信息，请参阅博客文章使用 IAM 身份验证通过 pgAdmin Amazon Aurora PostgreSQL 或 Amazon RDS for PostgreSQL 进行连接。

主题

生成 IAM 身份验证令牌

身份验证令牌包含几百个字符，因此，很难使用命令行对其进行处理。该问题的解决方式是，将令牌保存到一个环境变量中，然后在连接时使用此变量。以下示例说明了如何使用 Amazon CLI 通过 `generate-db-auth-token` 命令获取签名的身份验证令牌，并将其存储在 `PGPASSWORD` 环境变量中。

```
export RDSHOST="rdspostgres.123456789012.us-west-2.rds.amazonaws.com"
export PGPASSWORD="$(aws rds generate-db-auth-token --hostname $RDSHOST --port 5432 --
region us-west-2 --username jane_doe )"
```

在该示例中，`generate-db-auth-token` 命令的参数如下所示：

- `--hostname` – 要访问的数据库实例（集群终端节点）的主机名。
- `--port` – 用于连接到数据库实例的端口号
- `--region` – 在其中运行数据库实例的 Amazon 区域
- `--username` – 要访问的数据库账户

生成的令牌的前几个字符与以下内容类似。

```
rdspostgres.123456789012.us-west-2.rds.amazonaws.com:5432/?
Action=connect&DBUser=jane_doe&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Expires=900...
```

> **Note**
> 您不能使用自定义 Route 53 DNS 记录代替数据库实例端点来生成身份验证令牌。

连接到 Amazon RDS PostgreSQL 实例

使用 psql 进行连接的一般格式如下所示。

```
psql "host=hostName port=portNumber sslmode=verify-full
 sslrootcert=full_path_to_ssl_certificate dbname=DBName user=userName
 password=authToken"
```

参数如下所示：

- `host` – 要访问的数据库实例（集群终端节点）的主机名。
- `port` – 用于连接到数据库实例的端口号
- `sslmode` – 要使用的 SSL 模式

  当您使用 `sslmode=verify-full` 时，SSL 连接将针对 SSL 证书中的终端节点验证数据库实例终端节点。
- `sslrootcert` – 包含公有密钥的 SSL 证书文件的完整路径

  有关更多信息，请参阅 将 SSL 与 PostgreSQL 数据库实例结合使用。

  要下载 SSL 证书，请参阅 使用 SSL/TLS 加密与数据库实例或集群的连接。
- `dbname` – 要访问的数据库
- `user` – 要访问的数据库账户
- `password` – 已签名的 IAM 身份验证令牌

> **Note**
> 您不能使用自定义 Route 53 DNS 记录代替数据库实例端点来生成身份验证令牌。

以下示例演示了使用 psql 进行连接。在该示例中，psql 将环境变量 `RDSHOST` 用于主机，将环境变量 `PGPASSWORD` 用于生产的令牌。*/sample_dir/* 则是包含公有密钥的 SSL 证书文件的完整路径。

```
export RDSHOST="rdspostgres.123456789012.us-west-2.rds.amazonaws.com"
export PGPASSWORD="$(aws rds generate-db-auth-token --hostname $RDSHOST --port 5432 --
region us-west-2 --username jane_doe )"

psql "host=$RDSHOST port=5432 sslmode=verify-full sslrootcert=/sample_dir/global-
bundle.pem dbname=DBName user=jane_doe password=$PGPASSWORD"
```

如果您想通过代理连接到数据库实例，请参阅使用 IAM 身份验证连接到代理。

使用 IAM 身份验证和 适用于 .NET 的 Amazon SDK 连接到数据库实例

您可以使用 适用于 .NET 的 Amazon SDK 连接到 RRDS for MariaDB、MySQL 或 PostgreSQL 数据库实例 ，如下所述。

先决条件

以下是使用 IAM 身份验证连接到数据库实例的先决条件：

- 启用和禁用 IAM 数据库身份验证
- 创建和使用适用于 IAM 数据库访问的 IAM 策略
- 使用 IAM 身份验证创建数据库账户

示例

以下代码示例演示如何生成身份验证令牌，然后使用该令牌连接到数据库实例。

要运行该代码示例，您需要使用 适用于 .NET 的 Amazon SDK 网站上提供的 Amazon。`AWSSDK.CORE` 和 `AWSSDK.RDS` 程序包是必需的。要连接到数据库实例，请使用用于数据库引擎的 .NET 数据库连接器，例如 MySqlConnector for MariaDB 或 MySQL，或 Npgsql for PostgreSQL。

此代码连接到 MariaDB 或 MySQL 数据库实例。根据需要修改以下变量的值：

- `server` – 要访问的数据库实例的终端节点。
- `user` – 要访问的数据库账户
- `database` – 要访问的数据库
- `port` – 用于连接到数据库实例的端口号
- `SslMode` – 要使用的 SSL 模式

当您使用 SslMode=Required 时，SSL 连接将针对 SSL 证书中的终端节点验证数据库实例终端节点。

- SslCa – Amazon RDS 的 SSL 证书的完整路径

要下载证书，请参阅 使用 SSL/TLS 加密与数据库实例或集群的连接。

> **Note**
> 您不能使用自定义 Route 53 DNS 记录代替数据库实例端点来生成身份验证令牌。

```
using System;
using System.Data;
using MySql.Data;
using MySql.Data.MySqlClient;
using Amazon;

namespace ubuntu
{
  class Program
  {
    static void Main(string[] args)
    {
      var pwd =
 Amazon.RDS.Util.RDSAuthTokenGenerator.GenerateAuthToken(RegionEndpoint.USEast1,
 "mysqldb.123456789012.us-east-1.rds.amazonaws.com", 3306, "jane_doe");
      // for debug only Console.Write("{0}\n", pwd);  //this verifies the token is
 generated

      MySqlConnection conn = new MySqlConnection($"server=mysqldb.123456789012.us-
east-1.rds.amazonaws.com;user=jane_doe;database=mydB;port=3306;password={pwd};SslMode=Required;
      conn.Open();

      // Define a query
      MySqlCommand sampleCommand = new MySqlCommand("SHOW DATABASES;", conn);

      // Execute a query
      MySqlDataReader mysqlDataRdr = sampleCommand.ExecuteReader();

      // Read all rows and output the first column in each row
```

```
      while (mysqlDataRdr.Read())
        Console.WriteLine(mysqlDataRdr[0]);

      mysqlDataRdr.Close();
      // Close connection
      conn.Close();
    }
  }
}
```

此代码连接到 PostgreSQL 数据库实例。

根据需要修改以下变量的值：

- `Server` – 要访问的数据库实例的终端节点。
- `User ID` – 要访问的数据库账户
- `Database` – 要访问的数据库
- `Port` – 用于连接到数据库实例的端口号
- `SSL Mode` – 要使用的 SSL 模式

  当您使用 `SSL Mode=Required` 时，SSL 连接将针对 SSL 证书中的终端节点验证数据库实例终端节点。
- `Root Certificate` – Amazon RDS 的 SSL 证书的完整路径

  要下载证书，请参阅 使用 SSL/TLS 加密与数据库实例或集群的连接。

> **Note**
>
> 您不能使用自定义 Route 53 DNS 记录代替数据库实例端点来生成身份验证令牌。

```
using System;
using Npgsql;
using Amazon.RDS.Util;

namespace ConsoleApp1
{
    class Program
    {
```

```
        static void Main(string[] args)
        {
            var pwd =
 RDSAuthTokenGenerator.GenerateAuthToken("postgresmydb.123456789012.us-
east-1.rds.amazonaws.com", 5432, "jane_doe");
// for debug only Console.Write("{0}\n", pwd);  //this verifies the token is generated

            NpgsqlConnection conn = new
 NpgsqlConnection($"Server=postgresmydb.123456789012.us-east-1.rds.amazonaws.com;User
 Id=jane_doe;Password={pwd};Database=mydb;SSL Mode=Require;Root
 Certificate=full_path_to_ssl_certificate");
            conn.Open();

            // Define a query
                   NpgsqlCommand cmd = new NpgsqlCommand("select count(*) FROM
 pg_user", conn);

            // Execute a query
            NpgsqlDataReader dr = cmd.ExecuteReader();

            // Read all rows and output the first column in each row
            while (dr.Read())
                Console.Write("{0}\n", dr[0]);

            // Close connection
            conn.Close();
        }
    }
}
```

如果您想通过代理连接到数据库实例，请参阅使用 IAM 身份验证连接到代理。

使用 IAM 身份验证和 适用于 Go 的 Amazon SDK 连接到数据库实例

您可以使用 适用于 Go 的 Amazon SDK 连接到 RRDS for MariaDB、MySQL 或 PostgreSQL 数据库实例 ，如下所述。

先决条件

以下是使用 IAM 身份验证连接到数据库实例的先决条件：

- 启用和禁用 IAM 数据库身份验证
- 创建和使用适用于 IAM 数据库访问的 IAM 策略

- 使用 IAM 身份验证创建数据库账户

示例

要运行上述代码示例，您需要使用 适用于 Go 的 Amazon SDK 网站上提供的 Amazon。

根据需要修改以下变量的值：

- `dbName` – 要访问的数据库
- `dbUser` – 要访问的数据库账户
- `dbHost` – 要访问的数据库实例的终端节点。

> **Note**
>
> 您不能使用自定义 Route 53 DNS 记录代替数据库实例端点来生成身份验证令牌。

- `dbPort` – 用于连接到数据库实例的端口号
- `region` – 在其中运行数据库实例的 Amazon 区域

此外，请确保示例代码中的导入库存在于您的系统中。

> **Important**
>
> 本节中的示例使用以下代码提供从本地环境访问数据库的凭证：
>
> `creds := credentials.NewEnvCredentials()`
>
> 如果要从Amazon服务（如 Amazon EC2 或 Amazon ECS）访问数据库，则可以用以下代码替换：
>
> ```
> sess := session.Must(session.NewSession())
> creds := sess.Config.Credentials
> ```
>
> 如果您进行此更改，请确保添加以下导入：
>
> `"github.com/aws/aws-sdk-go/aws/session"`

主题

- 使用 IAM 身份验证和 适用于 Go 的 Amazon SDK V2 进行连接
- 使用 IAM 身份验证和 适用于 Go 的 Amazon SDK V1 进行连接

使用 IAM 身份验证和 适用于 Go 的 Amazon SDK V2 进行连接

您可以使用 IAM 身份验证和 适用于 Go 的 Amazon SDK V2 连接到数据库实例

以下代码示例演示如何生成身份验证令牌，然后使用该令牌连接到数据库实例。

此代码连接到 MariaDB 或 MySQL 数据库实例。

```
package main

import (
	"context"
	"database/sql"
	"fmt"

	"github.com/aws/aws-sdk-go-v2/config"
	"github.com/aws/aws-sdk-go-v2/feature/rds/auth"
	_ "github.com/go-sql-driver/mysql"
)

func main() {

	var dbName string = "DatabaseName"
	var dbUser string = "DatabaseUser"
	var dbHost string = "mysqldb.123456789012.us-east-1.rds.amazonaws.com"
	var dbPort int = 3306
	var dbEndpoint string = fmt.Sprintf("%s:%d", dbHost, dbPort)
	var region string = "us-east-1"

    cfg, err := config.LoadDefaultConfig(context.TODO())
    if err != nil {
     panic("configuration error: " + err.Error())
    }

    authenticationToken, err := auth.BuildAuthToken(
     context.TODO(), dbEndpoint, region, dbUser, cfg.Credentials)
    if err != nil {
     panic("failed to create authentication token: " + err.Error())
    }

    dsn := fmt.Sprintf("%s:%s@tcp(%s)/%s?tls=true&allowCleartextPasswords=true",
        dbUser, authenticationToken, dbEndpoint, dbName,
    )
```

```
    db, err := sql.Open("mysql", dsn)
    if err != nil {
        panic(err)
    }

    err = db.Ping()
    if err != nil {
        panic(err)
    }
}
```

此代码连接到 PostgreSQL 数据库实例。

```
package main

import (
     "context"
     "database/sql"
     "fmt"

     "github.com/aws/aws-sdk-go-v2/config"
     "github.com/aws/aws-sdk-go-v2/feature/rds/auth"
     _ "github.com/lib/pq"
)

func main() {

     var dbName string = "DatabaseName"
     var dbUser string = "DatabaseUser"
     var dbHost string = "postgresmydb.123456789012.us-east-1.rds.amazonaws.com"
     var dbPort int = 5432
     var dbEndpoint string = fmt.Sprintf("%s:%d", dbHost, dbPort)
     var region string = "us-east-1"

    cfg, err := config.LoadDefaultConfig(context.TODO())
    if err != nil {
     panic("configuration error: " + err.Error())
    }

    authenticationToken, err := auth.BuildAuthToken(
     context.TODO(), dbEndpoint, region, dbUser, cfg.Credentials)
    if err != nil {
     panic("failed to create authentication token: " + err.Error())
```

```
    }

    dsn := fmt.Sprintf("host=%s port=%d user=%s password=%s dbname=%s",
        dbHost, dbPort, dbUser, authenticationToken, dbName,
    )

    db, err := sql.Open("postgres", dsn)
    if err != nil {
        panic(err)
    }

    err = db.Ping()
    if err != nil {
        panic(err)
    }
}
```

如果您想通过代理连接到数据库实例，请参阅使用 IAM 身份验证连接到代理。

使用 IAM 身份验证和 适用于 Go 的 Amazon SDK V1 进行连接

使用 IAM 身份验证和 适用于 Go 的 Amazon SDK V1 连接到数据库实例

以下代码示例演示如何生成身份验证令牌，然后使用该令牌连接到数据库实例。

此代码连接到 MariaDB 或 MySQL 数据库实例。

```
package main

import (
    "database/sql"
    "fmt"
    "log"

    "github.com/aws/aws-sdk-go/aws/credentials"
    "github.com/aws/aws-sdk-go/service/rds/rdsutils"
    _ "github.com/go-sql-driver/mysql"
)

func main() {
    dbName := "app"
    dbUser := "jane_doe"
    dbHost := "mysqldb.123456789012.us-east-1.rds.amazonaws.com"
    dbPort := 3306
```

```
    dbEndpoint := fmt.Sprintf("%s:%d", dbHost, dbPort)
    region := "us-east-1"

    creds := credentials.NewEnvCredentials()
    authToken, err := rdsutils.BuildAuthToken(dbEndpoint, region, dbUser, creds)
    if err != nil {
        panic(err)
    }

    dsn := fmt.Sprintf("%s:%s@tcp(%s)/%s?tls=true&allowCleartextPasswords=true",
        dbUser, authToken, dbEndpoint, dbName,
    )

    db, err := sql.Open("mysql", dsn)
    if err != nil {
        panic(err)
    }

    err = db.Ping()
    if err != nil {
        panic(err)
    }
}
```

此代码连接到 PostgreSQL 数据库实例。

```
package main

import (
 "database/sql"
 "fmt"

 "github.com/aws/aws-sdk-go/aws/credentials"
 "github.com/aws/aws-sdk-go/service/rds/rdsutils"
 _ "github.com/lib/pq"
)

func main() {
    dbName := "app"
    dbUser := "jane_doe"
    dbHost := "postgresmydb.123456789012.us-east-1.rds.amazonaws.com"
    dbPort := 5432
    dbEndpoint := fmt.Sprintf("%s:%d", dbHost, dbPort)
```

```
    region := "us-east-1"

    creds := credentials.NewEnvCredentials()
    authToken, err := rdsutils.BuildAuthToken(dbEndpoint, region, dbUser, creds)
    if err != nil {
        panic(err)
    }

    dsn := fmt.Sprintf("host=%s port=%d user=%s password=%s dbname=%s",
        dbHost, dbPort, dbUser, authToken, dbName,
    )

    db, err := sql.Open("postgres", dsn)
    if err != nil {
        panic(err)
    }

    err = db.Ping()
    if err != nil {
        panic(err)
    }
}
```

如果您想通过代理连接到数据库实例，请参阅使用 IAM 身份验证连接到代理。

使用 IAM 身份验证和 适用于 Java 的 Amazon SDK 连接到数据库实例

您可以使用 适用于 Java 的 Amazon SDK 连接到 RRDS for MariaDB、MySQL 或 PostgreSQL 数据库实例 ，如下所述。

先决条件

以下是使用 IAM 身份验证连接到数据库实例的先决条件：

- 启用和禁用 IAM 数据库身份验证
- 创建和使用适用于 IAM 数据库访问的 IAM 策略
- 使用 IAM 身份验证创建数据库账户
- 设置适用于 Java 的Amazon开发工具包

有关如何使用 SDK for Java 2.x 的示例，请参阅 Amazon RDS examples using SDK for Java 2.x。也可以使用 Amazon 高级 JDBC 包装器，请参阅 Amazon Advanced JDBC Wrapper documentation。

主题

- 生成 IAM 身份验证令牌
- 手动构造 IAM 身份验证令牌
- 连接到数据库实例

生成 IAM 身份验证令牌

如果您使用适用于 Java 的 Amazon SDK编写程序，则可使用 RdsIamAuthTokenGenerator 类获取已签名的身份验证令牌。使用此类需要提供 Amazon 凭证。为此，创建 DefaultAWSCredentialsProviderChain 类的实例。DefaultAWSCredentialsProviderChain 使用它在默认凭证提供程序链中找到的第一个 Amazon 访问密钥和私有密钥。有关 Amazon 访问密钥的更多信息，请参阅管理用户的访问密钥。

> **Note**
>
> 您不能使用自定义 Route 53 DNS 记录代替数据库实例端点来生成身份验证令牌。

在创建一个 RdsIamAuthTokenGenerator 实例后，您可调用 getAuthToken 方法以获取已签名的令牌。提供 Amazon 区域、主机名、端口号和用户名。以下代码示例说明了如何执行该操作。

```
package com.amazonaws.codesamples;

import com.amazonaws.auth.DefaultAWSCredentialsProviderChain;
import com.amazonaws.services.rds.auth.GetIamAuthTokenRequest;
import com.amazonaws.services.rds.auth.RdsIamAuthTokenGenerator;

public class GenerateRDSAuthToken {

    public static void main(String[] args) {

     String region = "us-west-2";
     String hostname = "rdsmysql.123456789012.us-west-2.rds.amazonaws.com";
     String port = "3306";
     String username = "jane_doe";

     System.out.println(generateAuthToken(region, hostname, port, username));
    }
```

```
    static String generateAuthToken(String region, String hostName, String port, String
 username) {

     RdsIamAuthTokenGenerator generator = RdsIamAuthTokenGenerator.builder()
      .credentials(new DefaultAWSCredentialsProviderChain())
      .region(region)
      .build();

     String authToken = generator.getAuthToken(
      GetIamAuthTokenRequest.builder()
      .hostname(hostName)
      .port(Integer.parseInt(port))
      .userName(username)
      .build());

     return authToken;
    }

}
```

手动构造 IAM 身份验证令牌

在 Java 中，生成身份验证令牌的最简单方式是使用 `RdsIamAuthTokenGenerator`。此类将为您创建一个身份验证令牌，然后使用 Amazon 签名版本 4 为该令牌签名。有关更多信息，请参阅《Amazon Web Services 一般参考》中的签名版本 4 签名流程。

不过，您也可以手动构造身份验证令牌并为之签名，如以下代码示例中所示。

```
package com.amazonaws.codesamples;

import com.amazonaws.SdkClientException;
import com.amazonaws.auth.DefaultAWSCredentialsProviderChain;
import com.amazonaws.auth.SigningAlgorithm;
import com.amazonaws.util.BinaryUtils;
import org.apache.commons.lang3.StringUtils;

import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;
import java.nio.charset.Charset;
import java.security.MessageDigest;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.SortedMap;
```

```
import java.util.TreeMap;

import static com.amazonaws.auth.internal.SignerConstants.AWS4_TERMINATOR;
import static com.amazonaws.util.StringUtils.UTF8;

public class CreateRDSAuthTokenManually {
    public static String httpMethod = "GET";
    public static String action = "connect";
    public static String canonicalURIParameter = "/";
    public static SortedMap<String, String> canonicalQueryParameters = new TreeMap();
    public static String payload = StringUtils.EMPTY;
    public static String signedHeader = "host";
    public static String algorithm = "AWS4-HMAC-SHA256";
    public static String serviceName = "rds-db";
    public static String requestWithoutSignature;

    public static void main(String[] args) throws Exception {

        String region = "us-west-2";
        String instanceName = "rdsmysql.123456789012.us-west-2.rds.amazonaws.com";
        String port = "3306";
        String username = "jane_doe";

        Date now = new Date();
        String date = new SimpleDateFormat("yyyyMMdd").format(now);
        String dateTimeStamp = new
 SimpleDateFormat("yyyyMMdd'T'HHmmss'Z'").format(now);
        DefaultAWSCredentialsProviderChain creds = new
 DefaultAWSCredentialsProviderChain();
     String awsAccessKey = creds.getCredentials().getAWSAccessKeyId();
     String awsSecretKey = creds.getCredentials().getAWSSecretKey();
        String expiryMinutes = "900";

        System.out.println("Step 1:  Create a canonical request:");
        String canonicalString = createCanonicalString(username, awsAccessKey, date,
 dateTimeStamp, region, expiryMinutes, instanceName, port);
        System.out.println(canonicalString);
        System.out.println();

        System.out.println("Step 2:  Create a string to sign:");
        String stringToSign = createStringToSign(dateTimeStamp, canonicalString,
 awsAccessKey, date, region);
        System.out.println(stringToSign);
        System.out.println();
```

```
        System.out.println("Step 3:  Calculate the signature:");
        String signature = BinaryUtils.toHex(calculateSignature(stringToSign,
newSigningKey(awsSecretKey, date, region, serviceName)));
        System.out.println(signature);
        System.out.println();

        System.out.println("Step 4:  Add the signing info to the request");

        System.out.println(appendSignature(signature));
        System.out.println();

    }

    //Step 1: Create a canonical request date should be in format YYYYMMDD and dateTime
should be in format YYYYMMDDTHHMMSSZ
    public static String createCanonicalString(String user, String accessKey, String
date, String dateTime, String region, String expiryPeriod, String hostName, String
port) throws Exception {
        canonicalQueryParameters.put("Action", action);
        canonicalQueryParameters.put("DBUser", user);
        canonicalQueryParameters.put("X-Amz-Algorithm", "AWS4-HMAC-SHA256");
        canonicalQueryParameters.put("X-Amz-Credential", accessKey + "%2F" + date +
"%2F" + region + "%2F" + serviceName + "%2Faws4_request");
        canonicalQueryParameters.put("X-Amz-Date", dateTime);
        canonicalQueryParameters.put("X-Amz-Expires", expiryPeriod);
        canonicalQueryParameters.put("X-Amz-SignedHeaders", signedHeader);
        String canonicalQueryString = "";
        while(!canonicalQueryParameters.isEmpty()) {
            String currentQueryParameter = canonicalQueryParameters.firstKey();
            String currentQueryParameterValue =
canonicalQueryParameters.remove(currentQueryParameter);
            canonicalQueryString = canonicalQueryString + currentQueryParameter + "=" +
currentQueryParameterValue;
            if (!currentQueryParameter.equals("X-Amz-SignedHeaders")) {
                canonicalQueryString += "&";
            }
        }
        String canonicalHeaders = "host:" + hostName + ":" + port + '\n';
        requestWithoutSignature = hostName + ":" + port + "/?" + canonicalQueryString;

        String hashedPayload = BinaryUtils.toHex(hash(payload));
        return httpMethod + '\n' + canonicalURIParameter + '\n' + canonicalQueryString
+ '\n' + canonicalHeaders + '\n' + signedHeader + '\n' + hashedPayload;
```

```
    }

    //Step 2: Create a string to sign using sig v4
    public static String createStringToSign(String dateTime, String canonicalRequest,
 String accessKey, String date, String region) throws Exception {
        String credentialScope = date + "/" + region + "/" + serviceName + "/
aws4_request";
        return algorithm + '\n' + dateTime + '\n' + credentialScope + '\n' +
 BinaryUtils.toHex(hash(canonicalRequest));

    }

    //Step 3: Calculate signature
    /**
     * Step 3 of the &AWS; Signature version 4 calculation. It involves deriving
     * the signing key and computing the signature. Refer to
     * http://docs.aws.amazon
     * .com/general/latest/gr/sigv4-calculate-signature.html
     */
    public static byte[] calculateSignature(String stringToSign,
                                            byte[] signingKey) {
        return sign(stringToSign.getBytes(Charset.forName("UTF-8")), signingKey,
                SigningAlgorithm.HmacSHA256);
    }

    public static byte[] sign(byte[] data, byte[] key,
                          SigningAlgorithm algorithm) throws SdkClientException {
        try {
            Mac mac = algorithm.getMac();
            mac.init(new SecretKeySpec(key, algorithm.toString()));
            return mac.doFinal(data);
        } catch (Exception e) {
            throw new SdkClientException(
                    "Unable to calculate a request signature: "
                            + e.getMessage(), e);
        }
    }

    public static byte[] newSigningKey(String secretKey,
                                   String dateStamp, String regionName, String
 serviceName) {
        byte[] kSecret = ("AWS4" + secretKey).getBytes(Charset.forName("UTF-8"));
        byte[] kDate = sign(dateStamp, kSecret, SigningAlgorithm.HmacSHA256);
```

```
        byte[] kRegion = sign(regionName, kDate, SigningAlgorithm.HmacSHA256);
        byte[] kService = sign(serviceName, kRegion,
                SigningAlgorithm.HmacSHA256);
        return sign(AWS4_TERMINATOR, kService, SigningAlgorithm.HmacSHA256);
    }

    public static byte[] sign(String stringData, byte[] key,
                       SigningAlgorithm algorithm) throws SdkClientException {
        try {
            byte[] data = stringData.getBytes(UTF8);
            return sign(data, key, algorithm);
        } catch (Exception e) {
            throw new SdkClientException(
                    "Unable to calculate a request signature: "
                            + e.getMessage(), e);
        }
    }

    //Step 4: append the signature
    public static String appendSignature(String signature) {
        return requestWithoutSignature + "&X-Amz-Signature=" + signature;
    }

    public static byte[] hash(String s) throws Exception {
        try {
            MessageDigest md = MessageDigest.getInstance("SHA-256");
            md.update(s.getBytes(UTF8));
            return md.digest();
        } catch (Exception e) {
            throw new SdkClientException(
                    "Unable to compute hash while signing request: "
                            + e.getMessage(), e);
        }
    }
}
```

连接到数据库实例

以下代码示例演示如何生成一个身份验证令牌，然后使用该令牌连接到运行 MariaDB 或 MySQL 的实例。

要运行该代码示例，您需要使用 适用于 Java 的 Amazon SDK 网站上提供的 Amazon。此外，您需要：

- MySQL Connector/J。此代码示例已使用 `mysql-connector-java-5.1.33-bin.jar` 进行测试。
- 特定于Amazon区域的 Amazon RDS 的中间证书。（有关更多信息，请参阅 使用 SSL/TLS 加密与数据库实例或集群的连接。）在运行时，类加载程序会在与此 Java 代码示例相同的目录中查找证书以便可以找到它。
- 根据需要修改以下变量的值：
  - `RDS_INSTANCE_HOSTNAME` – 要访问的数据库实例的主机名。
  - `RDS_INSTANCE_PORT` – 用于连接到 PostgreSQL 数据库实例的端口号。
  - `REGION_NAME` – 在其中运行数据库实例的Amazon区域。
  - `DB_USER` – 要访问的数据库账户
  - `SSL_CERTIFICATE` – 特定于Amazon区域的 Amazon RDS 的中间证书。

  要为Amazon区域下载证书，请参阅使用 SSL/TLS 加密与数据库实例或集群的连接。将 SSL 证书放置在此 Java 程序文件所在的同一目录中，以便类加载程序可在运行时找到此证书。

此代码示例将从默认凭证提供程序链获取 Amazon 凭证。

Note

作为安全最佳实践，请为 `DEFAULT_KEY_STORE_PASSWORD` 指定除此处所示提示以外的密码。

```
package com.amazonaws.samples;

import com.amazonaws.services.rds.auth.RdsIamAuthTokenGenerator;
import com.amazonaws.services.rds.auth.GetIamAuthTokenRequest;
import com.amazonaws.auth.BasicAWSCredentials;
import com.amazonaws.auth.DefaultAWSCredentialsProviderChain;
import com.amazonaws.auth.AWSStaticCredentialsProvider;

import java.io.File;
import java.io.FileOutputStream;
import java.io.InputStream;
import java.security.KeyStore;
import java.security.cert.CertificateFactory;
import java.security.cert.X509Certificate;
```

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
import java.util.Properties;

import java.net.URL;

public class IAMDatabaseAuthenticationTester {
    //&AWS; Credentials of the IAM user with policy enabling IAM Database Authenticated
 access to the db by the db user.
    private static final DefaultAWSCredentialsProviderChain creds = new
 DefaultAWSCredentialsProviderChain();
    private static final String AWS_ACCESS_KEY =
 creds.getCredentials().getAWSAccessKeyId();
    private static final String AWS_SECRET_KEY =
 creds.getCredentials().getAWSSecretKey();

    //Configuration parameters for the generation of the IAM Database Authentication
 token
    private static final String RDS_INSTANCE_HOSTNAME = "rdsmysql.123456789012.us-
west-2.rds.amazonaws.com";
    private static final int RDS_INSTANCE_PORT = 3306;
    private static final String REGION_NAME = "us-west-2";
    private static final String DB_USER = "jane_doe";
    private static final String JDBC_URL = "jdbc:mysql://" + RDS_INSTANCE_HOSTNAME +
 ":" + RDS_INSTANCE_PORT;

    private static final String SSL_CERTIFICATE = "rds-ca-2019-us-west-2.pem";

    private static final String KEY_STORE_TYPE = "JKS";
    private static final String KEY_STORE_PROVIDER = "SUN";
    private static final String KEY_STORE_FILE_PREFIX = "sys-connect-via-ssl-test-
cacerts";
    private static final String KEY_STORE_FILE_SUFFIX = ".jks";
    private static final String DEFAULT_KEY_STORE_PASSWORD = "changeit";

    public static void main(String[] args) throws Exception {
        //get the connection
        Connection connection = getDBConnectionUsingIam();

        //verify the connection is successful
        Statement stmt= connection.createStatement();
        ResultSet rs=stmt.executeQuery("SELECT 'Success!' FROM DUAL;");
```

```
        while (rs.next()) {
             String id = rs.getString(1);
            System.out.println(id); //Should print "Success!"
        }

        //close the connection
        stmt.close();
        connection.close();

        clearSslProperties();

    }

    /**
     * This method returns a connection to the db instance authenticated using IAM
 Database Authentication
     * @return
     * @throws Exception
     */
    private static Connection getDBConnectionUsingIam() throws Exception {
        setSslProperties();
        return DriverManager.getConnection(JDBC_URL, setMySqlConnectionProperties());
    }

    /**
     * This method sets the mysql connection properties which includes the IAM Database
 Authentication token
     * as the password. It also specifies that SSL verification is required.
     * @return
     */
    private static Properties setMySqlConnectionProperties() {
        Properties mysqlConnectionProperties = new Properties();
        mysqlConnectionProperties.setProperty("verifyServerCertificate","true");
        mysqlConnectionProperties.setProperty("useSSL", "true");
        mysqlConnectionProperties.setProperty("user",DB_USER);
        mysqlConnectionProperties.setProperty("password",generateAuthToken());
        return mysqlConnectionProperties;
    }

    /**
     * This method generates the IAM Auth Token.
     * An example IAM Auth Token would look like follows:
     * btusi123.cmz7kenwo2ye.rds.cn-north-1.amazonaws.com.cn:3306/?
Action=connect&DBUser=iamtestuser&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-
```

```
Date=20171003T010726Z&X-Amz-SignedHeaders=host&X-Amz-Expires=899&X-Amz-
Credential=AKIAPFXHGVDI5RNFO4AQ%2F20171003%2Fcn-north-1%2Frds-db%2Faws4_request&X-Amz-
Signature=f9f45ef96c1f770cdad11a53e33ffa4c3730bc03fdee820cfdf1322eed15483b
     * @return
     */
    private static String generateAuthToken() {
        BasicAWSCredentials awsCredentials = new BasicAWSCredentials(AWS_ACCESS_KEY,
 AWS_SECRET_KEY);

        RdsIamAuthTokenGenerator generator = RdsIamAuthTokenGenerator.builder()
                .credentials(new
 AWSStaticCredentialsProvider(awsCredentials)).region(REGION_NAME).build();
        return generator.getAuthToken(GetIamAuthTokenRequest.builder()

 .hostname(RDS_INSTANCE_HOSTNAME).port(RDS_INSTANCE_PORT).userName(DB_USER).build());
    }

    /**
     * This method sets the SSL properties which specify the key store file, its type
 and password:
     * @throws Exception
     */
    private static void setSslProperties() throws Exception {
        System.setProperty("javax.net.ssl.trustStore", createKeyStoreFile());
        System.setProperty("javax.net.ssl.trustStoreType", KEY_STORE_TYPE);
        System.setProperty("javax.net.ssl.trustStorePassword",
 DEFAULT_KEY_STORE_PASSWORD);
    }

    /**
     * This method returns the path of the Key Store File needed for the SSL
 verification during the IAM Database Authentication to
     * the db instance.
     * @return
     * @throws Exception
     */
    private static String createKeyStoreFile() throws Exception {
        return createKeyStoreFile(createCertificate()).getPath();
    }

    /**
     *  This method generates the SSL certificate
     * @return
     * @throws Exception
```

```
     */
    private static X509Certificate createCertificate() throws Exception {
        CertificateFactory certFactory = CertificateFactory.getInstance("X.509");
        URL url = new File(SSL_CERTIFICATE).toURI().toURL();
        if (url == null) {
            throw new Exception();
        }
        try (InputStream certInputStream = url.openStream()) {
            return (X509Certificate) certFactory.generateCertificate(certInputStream);
        }
    }

    /**
     * This method creates the Key Store File
     * @param rootX509Certificate - the SSL certificate to be stored in the KeyStore
     * @return
     * @throws Exception
     */
    private static File createKeyStoreFile(X509Certificate rootX509Certificate) throws
 Exception {
        File keyStoreFile = File.createTempFile(KEY_STORE_FILE_PREFIX,
 KEY_STORE_FILE_SUFFIX);
        try (FileOutputStream fos = new FileOutputStream(keyStoreFile.getPath())) {
            KeyStore ks = KeyStore.getInstance(KEY_STORE_TYPE, KEY_STORE_PROVIDER);
            ks.load(null);
            ks.setCertificateEntry("rootCaCertificate", rootX509Certificate);
            ks.store(fos, DEFAULT_KEY_STORE_PASSWORD.toCharArray());
        }
        return keyStoreFile;
    }

    /**
     * This method clears the SSL properties.
     * @throws Exception
     */
    private static void clearSslProperties() throws Exception {
           System.clearProperty("javax.net.ssl.trustStore");
           System.clearProperty("javax.net.ssl.trustStoreType");
           System.clearProperty("javax.net.ssl.trustStorePassword");
    }

}
```

如果您想通过代理连接到数据库实例，请参阅使用 IAM 身份验证连接到代理。

使用 IAM 身份验证和 适用于 Python (Boto3) 的 Amazon SDK 连接到数据库实例

您可以使用 适用于 Python (Boto3) 的 Amazon SDK 连接到 RRDS for MariaDB、MySQL 或 PostgreSQL 数据库实例 ，如下所述。

先决条件

以下是使用 IAM 身份验证连接到数据库实例的先决条件：

- 启用和禁用 IAM 数据库身份验证
- 创建和使用适用于 IAM 数据库访问的 IAM 策略
- 使用 IAM 身份验证创建数据库账户

此外，请确保示例代码中的导入库存在于您的系统中。

示例

该代码示例将配置文件用于共享凭证。有关指定凭证的信息，请参阅 适用于 Python (Boto3) 的 Amazon SDK 文档中的凭证。

以下代码示例演示如何生成身份验证令牌，然后使用该令牌连接到数据库实例。

要运行该代码示例，您需要使用 适用于 Python (Boto3) 的 Amazon SDK 网站上提供的 Amazon。

根据需要修改以下变量的值：

- ENDPOINT – 要访问的数据库实例的终端节点。
- PORT – 用于连接到数据库实例的端口号
- USER – 要访问的数据库账户
- REGION – 在其中运行数据库实例的 Amazon 区域
- DBNAME – 要访问的数据库
- SSLCERTIFICATE – Amazon RDS 的 SSL 证书的完整路径

  对于 ssl_ca，请指定 SSL 证书。要下载 SSL 证书，请参阅 使用 SSL/TLS 加密与数据库实例或集群的连接。

> **Note**
>
> 您不能使用自定义 Route 53 DNS 记录代替数据库实例端点来生成身份验证令牌。

此代码连接到 MariaDB 或 MySQL 数据库实例。

在运行此代码之前，请按照 Python 包索引中的说明安装 PyMySQL 驱动程序。

```
import pymysql
import sys
import boto3
import os

ENDPOINT="mysqldb.123456789012.us-east-1.rds.amazonaws.com"
PORT="3306"
USER="jane_doe"
REGION="us-east-1"
DBNAME="mydb"
os.environ['LIBMYSQL_ENABLE_CLEARTEXT_PLUGIN'] = '1'

#gets the credentials from .aws/credentials
session = boto3.Session(profile_name='default')
client = session.client('rds')

token = client.generate_db_auth_token(DBHostname=ENDPOINT, Port=PORT, DBUsername=USER,
 Region=REGION)

try:
    conn =
 pymysql.connect(auth_plugin_map={'mysql_clear_password':None},host=ENDPOINT,
 user=USER, password=token, port=PORT, database=DBNAME, ssl_ca='SSLCERTIFICATE',
 ssl_verify_identity=True)
    cur = conn.cursor()
    cur.execute("""SELECT now()""")
    query_results = cur.fetchall()
    print(query_results)
except Exception as e:
    print("Database connection failed due to {}".format(e))
```

此代码连接到 PostgreSQL 数据库实例。

在运行此代码之前，请按照 Psycopg 文档中的说明安装 psycopg2。

```
import psycopg2
import sys
import boto3
import os

ENDPOINT="postgresmydb.123456789012.us-east-1.rds.amazonaws.com"
PORT="5432"
USER="jane_doe"
REGION="us-east-1"
DBNAME="mydb"

#gets the credentials from .aws/credentials
session = boto3.Session(profile_name='RDSCreds')
client = session.client('rds')

token = client.generate_db_auth_token(DBHostname=ENDPOINT, Port=PORT, DBUsername=USER,
 Region=REGION)

try:
    conn = psycopg2.connect(host=ENDPOINT, port=PORT, database=DBNAME, user=USER,
 password=token, sslrootcert="SSLCERTIFICATE")
    cur = conn.cursor()
    cur.execute("""SELECT now()""")
    query_results = cur.fetchall()
    print(query_results)
except Exception as e:
    print("Database connection failed due to {}".format(e))
```

如果您想通过代理连接到数据库实例，请参阅使用 IAM 身份验证连接到代理。

IAM 数据库身份验证故障排除

下面，您可以找到一些常见 IAM 数据库身份验证问题的故障排除思路，以及有关 IAM 数据库身份验证的 CloudWatch 日志和指标的信息。

将 IAM 数据库身份验证错误日志导出到 CloudWatch Logs

IAM 数据库身份验证错误日志存储在数据库主机上，您可以将这些日志导出到您的 CloudWatch Logs 账户。使用本页中的日志和补救方法来解决 IAM 数据库身份验证问题。

可以通过控制台、Amazon CLI 和 RDS API 启用向 CloudWatch Logs 导出日志。有关控制台说明，请参阅 将数据库日志发布到 Amazon CloudWatch Logs。

要在从 Amazon CLI 中创建数据库实例时将 IAM 数据库身份验证错误日志导出到 CloudWatch Logs，请使用以下命令：

```
aws rds create-db-instance --db-instance-identifier mydbinstance \
--region us-east-1 \
--db-instance-class db.t3.large \
--allocated-storage 50 \
--engine postgres \
--engine-version 16 \
--port 5432 \
--master-username master \
--master-user-password password \
--publicly-accessible \
--enable-iam-database-authentication \
--enable-cloudwatch-logs-exports=iam-db-auth-error
```

要在从 Amazon CLI 中修改数据库实例时将 IAM 数据库身份验证错误日志导出到 CloudWatch Logs，请使用以下命令：

```
aws rds modify-db-instance --db-instance-identifier mydbinstance \
--region us-east-1 \
--cloudwatch-logs-export-configuration '{"EnableLogTypes":["iam-db-auth-error"]}'
```

要验证数据库实例是否正在将 IAM 数据库身份验证日志导出到 CloudWatch Logs，请检查 `describe-db-instances` 命令输出中的 `EnabledCloudwatchLogsExports` 参数是否设置为 `iam-db-auth-error`。

```
aws rds describe-db-instances --region us-east-1 --db-instance-identifier mydbinstance
            ...

             "EnabledCloudwatchLogsExports": [
                "iam-db-auth-error"
            ],
            ...
```

IAM 数据库身份验证 CloudWatch 指标

Amazon RDS 向您的 Amazon CloudWatch 账户提供有关 IAM 数据库身份验证的近乎实时的指标。下表列出了使用 CloudWatch 时可用的 IAM 数据库身份验证指标：

指标	描述
`IamDbAuthConnectionRequests`	使用 IAM 数据库身份验证发出的连接请求总数。
`IamDbAuthConnectionSuccess`	成功的 IAM 数据库身份验证请求总数。
`IamDbAuthConnectionFailure`	失败的 IAM 数据库身份验证请求总数。
`IamDbAuthConnectionFailureInvalidToken`	由于令牌无效而失败的 IAM 数据库身份验证请求总数。
`IamDbAuthConnectionFailureInsufficientPermissions`	由于策略或权限不正确而失败的 IAM 数据库身份验证请求总数。
`IamDbAuthConnectionFailureThrottling`	由于 IAM 数据库身份验证节流而失败的 IAM 数据库身份验证请求总数。
`IamDbAuthConnectionFailureServerError`	由于 IAM 数据库身份验证功能中的内部服务器错误而失败的 IAM 数据库身份验证请求总数。

常见问题和解决方案

使用 IAM 数据库身份验证时可能会遇到以下问题。使用表中的修复步骤来解决问题：

错误	指标	原因	解决方案

`[ERROR] Failed to authenticate the connection request for user` *`db_user`* `because the provided token`	`IamDbAuthConnectionFailure`
	
	`IamDbAuthConnectio`
		连接请求中的 IAM 数据库身份验证令牌要么不是有效的 SigV4a 令牌，要么格式不正确。	在应用程序中检查令牌生成策略。在某些情况下，请确保使用有效的格式传递令牌。截断令牌（或字符串格式不正确）将使令牌失效。

错误	指标	原因	解决方案
is malformed or otherwise invalid. (Status Code: 400, Error Code: InvalidToken)	nFailureInvalidToken		
[ERROR] Failed to authenticate the connection request for user *db_user* because the token age is longer than 15 minutes. (Status Code: 400, Error Code:ExpiredToken)	IamDbAuthConnectionFailure IamDbAuthConnectionFailureInvalidToken	IAM 数据库身份验证令牌已到期。令牌仅在 15 分钟内有效。	检查应用程序中的令牌缓存和/或令牌重用逻辑。您不应重用超过 15 分钟的令牌。

错误	指标	原因	解决方案
[ERROR] Failed to authorize the connection request for user *db_user* because the IAM policy assumed by the caller 'arn:aws:sts::123456789012:assumed-role/ <RoleName>/ <RoleSession>' is not authorized to perform `rds-db:connect` on the DB instance. (Status Code: 403, Error Code:NotAuthorized)	IamDbAuthConnectionFailure IamDbAuthConnectionFailureInsufficientPermissions	该错误可能是由于以下原因引起的： • 应用程序采用的 IAM 策略未授权执行 `rds-db:connect` 操作。 • 您为 *db_user* 采用的角色/策略不正确，无法连接到数据库。 • 您为 *db_user* 采用的策略是正确的，但是您没有连接到正确的数据库。	验证您在应用程序中采用的 IAM 角色和/或策略。请务必采用与连接到数据库相同的策略来生成令牌。
[ERROR] Failed to authorize the connection request for user *db_user* due to IAM DB authentication throttling. (Status Code: 429, Error Code: ThrottlingException)	IamDbAuthConnectionFailure IamDbAuthConnectionFailureThrottling	您在短时间内向数据库发出的连接请求过多。IAM 数据库身份验证节流限制为每秒 200 个连接。	降低使用 IAM 身份验证建立新连接的速率。考虑使用 RDS 代理来实现连接池，以便在应用程序中重用已建立的连接。

错误	指标	原因	解决方案
`[ERROR] Failed to authorize the connection request for user` *`db_user`* `due to an internal IAM DB authentication error. (Status Code: 500, Error Code: InternalError)`	`IamDbAuthConnectionFailure` `IamDbAuthConnectionFailureThrottling`	使用 IAM 数据库身份验证授权数据库连接时出现内部错误。	请联系 https://aws.amazon.com/premiumsupport/ 以便调查此问题。

对 Amazon RDS 身份和访问权限问题进行故障排除

使用以下信息可帮助您诊断和修复在使用 Amazon RDS 和 IAM 时可能遇到的常见问题。

主题

- 我无权在 Amazon RDS 中执行操作
- 未授权我执行 iam:PassRole
- 我希望允许我 Amazon 账户以外的人访问我的 Amazon RDS 资源

我无权在 Amazon RDS 中执行操作

如果 Amazon Web Services Management Console 告诉您，您无权执行某个操作，则必须联系您的管理员寻求帮助。管理员是向您提供登录凭证的人。

当 `mateojackson` 用户尝试使用控制台查看有关 *`widget`* 的详细信息但不具有 `rds:`*`GetWidget`* 权限时，会发生以下示例错误。

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 rds:GetWidget on resource: my-example-widget
```

在这种情况下，Mateo 请求他的管理员更新其策略，以允许他使用 `rds:`*`GetWidget`* 操作访问 *`my-example-widget`* 资源。

未授权我执行 iam:PassRole

如果您收到错误消息，提示您无权执行 `iam:PassRole` 操作，则必须联系您的管理员寻求帮助。管理员是向您提供登录凭证的人。请求该人员更新您的策略，以便允许您将角色传递给 Amazon RDS。

有些 Amazon 服务允许您将现有角色传递到该服务，而不是创建新服务角色或服务相关角色。为此，您必须具有将角色传递到服务的权限。

当名为 `marymajor` 的用户尝试使用控制台在 Amazon RDS 中执行操作时，会发生以下示例错误。但是，服务必须具有服务角色所授予的权限才可执行操作。Mary 不具有将角色传递到服务的权限。

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

在这种情况下，Mary 请求她的管理员来更新其策略，以允许她执行 `iam:PassRole` 操作。

我希望允许我 Amazon 账户以外的人访问我的 Amazon RDS 资源

您可以创建一个角色，以便其他账户中的用户或您组织外的人员可以使用该角色来访问您的资源。您可以指定谁值得信赖，可以担任角色。对于支持基于资源的策略或访问控制列表（ACL）的服务，您可以使用这些策略向人员授予对您的资源的访问权。

要了解更多信息，请参阅以下内容：

- 要了解 Amazon RDS 是否支持这些功能，请参阅Amazon RDS 如何与 IAM 协同工作。
- 要了解如何为您拥有的 Amazon 账户中的资源提供访问权限，请参阅 IAM 用户指南中的为您拥有的另一个 Amazon 账户中的 IAM 用户提供访问权限。
- 要了解如何为第三方 Amazon 账户提供您的资源的访问权限，请参阅IAM 用户指南中的 为第三方拥有的 Amazon 账户提供访问权限。
- 要了解如何通过联合身份验证提供访问权限，请参阅 IAM 用户指南中的为经过外部身份验证的用户（联合身份验证）提供访问权限。
- 要了解使用角色和基于资源的策略进行跨账户访问之间的差别，请参阅《IAM 用户指南》中的 IAM 角色与基于资源的策略有何不同。

Amazon RDS 中的日志记录和监控

对于保持 Amazon RDS 和您的 Amazon 解决方案的可靠性、可用性和性能，监控发挥着非常重要的作用。您应该从 Amazon 解决方案的各个部分收集监控数据，以便更轻松地调试多点故障（如果发生）。Amazon 提供了多种工具来监控您的 Amazon RDS 资源并对潜在事件做出响应：

Amazon CloudWatch 警报

使用 Amazon CloudWatch 警报，您可以在指定时间段内监控某个指标。如果指标超过给定阈值，则会向 Amazon SNS 主题或 Amazon Auto Scaling 策略发送通知。CloudWatch 警报将不会调用操作，因为这些操作处于特定状态。而是必须在状态已改变并在指定的若干个时间段内保持不变后才调用。

Amazon CloudTrail 日志

CloudTrail 提供了用户、角色或 Amazon 服务在 Amazon RDS 中所执行操作的记录。CloudTrail 将对 Amazon RDS 的所有 API 调用作为事件捕获，包括来自控制台的调用和代码对 Amazon RDS API 操作的调用。通过使用 CloudTrail 收集的信息，您可以确定向 Amazon RDS 发出了什么请求、发出请求的 IP 地址、何人发出的请求、请求的发出时间以及其他详细信息。有关更多信息，请参阅 监控 Amazon CloudTrail 中的 Amazon RDS API 调用。

增强监控

Amazon RDS 针对运行数据库实例的操作系统（OS），实时提供操作系统的指标。您可以使用控制台查看数据库实例的指标，或者在您选择的监控系统中使用 Amazon CloudWatch Logs 的增强监控 JSON 输出。有关更多信息，请参阅 使用增强监控来监控操作系统指标。

Amazon RDS Performance Insights

Performance Insights 在现有 Amazon RDS 监控功能的基础上进行了扩展，以便说明您的数据库的性能并帮助您分析影响性能的任何问题。利用 Performance Insights 控制面板，您可以可视化数据库负载并按等待状态、SQL 语句、主机或用户来筛选负载。有关更多信息，请参阅 在 Amazon RDS 上使用性能详情监控数据库负载。

数据库日志

您可以使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 查看、下载和监控数据库日志。有关更多信息，请参阅 监控 Amazon RDS 日志文件。

Amazon RDS 建议

Amazon RDS 可自动提供数据库资源建议。这些建议通过分析数据库实例配置、使用情况和性能数据来提供最佳实践准则。有关更多信息，请参阅 来自 Amazon RDS 的建议。

Amazon RDS 事件通知

Amazon RDS 使用 Amazon Simple Notification Service（Amazon SNS）在发生 Amazon RDS 事件时提供通知。这些通知可以采用Amazon区域 Amazon SNS 支持的任何通知形式，例如电子邮件、文本消息或对 HTTP 终端节点的调用。有关更多信息，请参阅 使用 Amazon RDS 事件通知。

Amazon Trusted Advisor

Trusted Advisor 凝聚了从为数十万 Amazon 客户提供服务中总结的最佳实践。Trusted Advisor 可检查您的 Amazon 环境，然后在有可能节省开支、提高系统可用性和性能或弥补安全漏洞时为您提供建议。所有 Amazon 客户均有权访问五个 Trusted Advisor 检查。使用“商业”和“企业”支持计划的客户可以查看所有 Trusted Advisor 检查。

Trusted Advisor 提供与 Amazon RDS 相关的下列检查：

- Amazon RDS 闲置数据库实例
- Amazon RDS 安全组访问风险
- Amazon RDS 备份
- Amazon RDS 多可用区

有关这些检查的更多信息，请参阅 Trusted Advisor 最佳实践（检查）。

有关监控 Amazon RDS 的更多信息，请参阅监控 Amazon RDS 实例中的指标。

Amazon RDS 的合规性验证

作为多个Amazon合规性计划的一部分，第三方审计员将评估 Amazon RDS 的安全性和合规性。其中包括 SOC、PCI、FedRAMP、HIPAA 及其他。

有关特定合规性计划范围内的 Amazon 服务的列表，请参阅合规性计划范围内的 Amazon 服务。有关一般信息，请参阅 Amazon 合规性计划。

您可以使用 Amazon Artifact 下载第三方审计报告。有关更多信息，请参阅在 Amazon Artifact 中下载报告。

您在使用 Amazon RDS 时的合规性责任由您的数据的敏感性、您组织的合规性目标以及适用的法律法规决定。Amazon提供以下资源以帮助满足合规性要求：

- 安全性与合规性快速入门指南 – 这些部署指南讨论了架构注意事项，并提供了在Amazon上部署基于安全性和合规性的基准环境的步骤。
- Amazon Web Services 上的 HIPAA 安全性和合规性架构设计 – 此白皮书介绍了公司如何使用 Amazon 创建符合 HIPAA 标准的应用程序。
- Amazon 合规性资源 – 可能适用于您的行业和位置的业务手册和指南集合。
- Amazon Config – 此Amazon服务评估您的资源配置对内部实践、行业指南和法规的遵循情况。
- Amazon Security Hub – 此 Amazon Web Services 服务 向您提供 Amazon 中安全状态的全面视图。Security Hub 通过安全控件评估您的 Amazon 资源并检查其是否符合安全行业标准和最佳实践。有关受支持服务及控件的列表，请参阅 Security Hub 控件参考。

Amazon RDS 中的弹性

Amazon全球基础设施围绕Amazon区域和可用区构建。Amazon区域提供多个在物理上独立且隔离的可用区，这些可用区通过延迟低、吞吐量高且冗余性高的网络连接在一起。利用可用区，您可以设计和操作在可用区之间无中断地自动实现故障转移的应用程序和数据库。与传统的单个或多个数据中心基础架构相比，可用区具有更高的可用性、容错性和可扩展性。

有关 Amazon 区域和可用区的更多信息，请参阅 Amazon 全球基础设施。

除了Amazon全球基础设施之外，Amazon RDS 还提供了相应功能来帮助支持您的数据弹性和备份需求。

备份与还原

Amazon RDS 创建并保存数据库实例的自动备份。Amazon RDS 创建数据库实例的存储卷快照，并备份整个数据库实例而不仅仅是单个数据库。

在数据库实例的备份时段中，Amazon RDS 创建数据库实例的自动备份。Amazon RDS 根据您指定的备份保留期保存数据库实例的自动备份。如果需要，您可以将数据库恢复到备份保留期中的任意时间点。您还可以通过手动创建数据库快照来手动备份数据库实例。

如果源数据库实例失败，则可以通过从该数据库快照还原为灾难恢复解决方案来创建数据库实例。

有关更多信息，请参阅“备份、还原和导出数据”。

复制

Amazon RDS 使用 MariaDB、MySQL、Oracle 和 PostgreSQL 数据库引擎的内置复制功能，从源数据库实例创建一个特殊类型的数据库实例，称为只读副本。对源数据库实例的更新将异步复制到只读副本。您可以将应用程序发出的读取查询路由到只读副本，以减轻源数据库实例上的负载。利用只读副本，可以实现弹性扩展并超越单个数据库实例的容量限制，以处理高读取量的数据库工作负载。如果源数据库实例出现故障，则将只读副本提升为独立的实例作为灾难恢复解决方案。对于一些数据库引擎，Amazon RDS 还支持其他复制选项。

有关更多信息，请参阅“使用数据库实例只读副本”。

故障转移

Amazon RDS 使用多可用区部署为数据库实例提供高可用性和故障转移支持。Amazon RDS 使用几种不同的技术来提供故障转移支持。用于 Oracle、PostgreSQL、MySQL 和 MariaDB 数据库实例的

多可用区部署使用 Amazon 的故障转移技术。SQL Server 数据库实例使用 SQL Server 数据库镜像 (DBM)。

有关更多信息，请参阅“配置和管理 Amazon RDS 的多可用区部署”。

Amazon RDS 中的基础设施安全性

作为一项托管式服务，Amazon Relational Database Service 受 Amazon 全球网络安全保护。有关 Amazon 安全服务以及 Amazon 如何保护基础设施的信息，请参阅 Amazon 云安全。要按照基础架构安全最佳实践设计您的 Amazon 环境，请参阅《安全性支柱 Amazon Well‐Architected Framework》中的 基础架构保护。

您可以使用 Amazon 发布的 API 调用通过网络访问 Amazon RDS。客户端必须支持以下内容：

- 传输层安全性协议 (TLS) 我们要求使用 TLS 1.2，建议使用 TLS 1.3。
- 具有完全向前保密 (PFS) 的密码套件，例如 DHE（临时 Diffie-Hellman）或 ECDHE（临时椭圆曲线 Diffie-Hellman）。大多数现代系统（如 Java 7 及更高版本）都支持这些模式。

此外，必须使用访问密钥 ID 和与 IAM 主体关联的秘密访问密钥来对请求进行签名。或者，您可以使用 Amazon Security Token Service（Amazon STS）生成临时安全凭证来对请求进行签名。

此外，Amazon RDS 提供了相应功能来帮助支持基础设施安全性。

安全组

安全组控制着流量在数据库实例内外拥有的访问权限。原定设置情况下，将对您的数据库实例关闭网络访问。您可以在安全组中指定规则，允许从 IP 地址范围、端口或安全组进行访问。配置传入规则后，会向与该安全组关联的所有数据库实例应用相同的规则。

有关更多信息，请参阅 使用安全组控制访问权限。

公开可用性

在基于 Amazon VPC 服务的虚拟私有云（VPC）内启动数据库实例时，可以打开或关闭该数据库实例的公共可访问性。要指定您创建的数据库实例是否具有解析为公共 IP 地址的 DNS 名称，请使用公共可用性 参数。通过使用此参数，您可以指定是否可以公开访问数据库实例。您可以通过修改 Public accessibility 参数修改数据库实例以打开或关闭公开可用性。

有关更多信息，请参阅“对互联网隐藏 VPC 中的数据库实例”。

> **Note**
>
> 如果您的数据库实例位于 VPC 中但不可公开访问，则您还可以使用Amazon Site-to-Site VPN 连接或 Amazon Direct Connect 连接从专用网络访问该实例。有关更多信息，请参阅 互联网络流量隐私。

Amazon RDS API 和接口 VPC 终端节点 (Amazon PrivateLink)

您可以通过创建接口 VPC 终端节点 在 VPC 和 Amazon RDS API 终端节点之间建立私有连接。接口终端节点由 提供支持Amazon PrivateLink

Amazon PrivateLink 使您可以私下访问 Amazon RDS API 操作，而无需互联网网关、NAT 设备、VPN 连接或 Amazon Direct Connect 连接。VPC 中的数据库实例不需要公有 IP 地址即可与 Amazon RDS API 端点进行通信，进而启动、修改或终止数据库实例。您的数据库实例也不需要公有 IP 地址即可使用任何可用的 RDS API 操作。您的 VPC 和 Amazon RDS 之间的流量不会脱离 Amazon 网络。

每个接口终端节点均由子网中的一个或多个弹性网络接口表示。有关弹性网络接口的更多信息，请参阅《Amazon EC2 用户指南》中的弹性网络接口。

有关 VPC 终端节点的更多信息，请参阅 Amazon VPC 用户指南中的接口 VPC 终端节点 (Amazon PrivateLink)。有关 RDS API 操作的信息，请参阅 Amazon RDS API 参考。

您不需要接口 VPC 端点即可连接到数据库实例。有关更多信息，请参阅 在 VPC 中访问数据库实例的场景。

VPC 终端节点注意事项

在为 Amazon RDS API 终端节点设置接口 VPC 终端节点之前，请务必查看Amazon VPC 用户指南 中的接口终端节点属性和限制。

可以从使用 Amazon PrivateLink 的 VPC 中获取所有与管理 Amazon RDS 资源相关的 RDS API 操作。

RDS API 终端节点支持 VPC 终端节点策略。默认情况下，允许通过终端节点对 RDS API 操作进行完全访问。有关更多信息，请参阅 Amazon VPC 用户指南中的使用 VPC 终端节点控制对服务的访问。

可用性

Amazon RDS API 当前在以下Amazon区域中支持 VPC 终端节点：

- US East (Ohio)
- 美国东部（弗吉尼亚州北部）
- 美国西部（加利福尼亚北部）
- 美国西部（俄勒冈州）

- 非洲（开普敦）
- 亚太地区（香港）
- Asia Pacific (Mumbai)
- 亚太地区（大阪）
- 亚太地区（首尔）
- 亚太地区（新加坡）
- 亚太地区（悉尼）
- 亚太地区（东京）
- 加拿大（中部）
- 加拿大西部（卡尔加里）
- 中国（北京）
- 中国（宁夏）
- 欧洲地区（法兰克福）
- 欧洲（苏黎世）
- 欧洲地区（爱尔兰）
- 欧洲地区（伦敦）
- 欧洲地区（巴黎）
- Europe (Stockholm)
- 欧洲地区（米兰）
- 以色列（特拉维夫）
- 中东（巴林）
- 南美洲（圣保罗）

为 Amazon RDS API 创建接口 VPC 终端节点

您可以使用 Amazon VPC 控制台或 Amazon Command Line Interface (Amazon CLI) 为 EBS 直接 Amazon RDS API 服务创建 VPC 终端节点。有关更多信息，请参阅 Amazon VPC 用户指南中的创建接口终端节点。

使用服务名称 `com.amazonaws.`*`region`*`.rds` 为 Amazon RDS API 创建 VPC 终端节点。

如果您为终端节点启用私有 DNS，则可以将其默认 DNS 名称用于Amazon区域（例如 `rds.us-east-1.amazonaws.com`），从而通过 VPC 终端节点向 Amazon RDS 发出 API 请求（中国的Amazon区域除外）。对于中国（北京）和中国（宁夏）Amazon区域，您可以通过 VPC 终端节点分别使用 `rds-api.cn-north-1.amazonaws.com.cn` 和 `rds-api.cn-northwest-1.amazonaws.com.cn` 发出 API 请求。

有关更多信息，请参阅 Amazon VPC 用户指南中的[通过接口终端节点访问服务]。

为 Amazon RDS API 创建 VPC 终端节点策略

您可以为 VPC 终端节点附加控制对 Amazon RDS API 的访问的终端节点策略。该策略指定以下信息：

- 可执行操作的主体。
- 可执行的操作。
- 可对其执行操作的资源。

有关更多信息，请参阅 Amazon VPC 用户指南中的[使用 VPC 终端节点控制对服务的访问]。

示例：Amazon RDS API 操作的 VPC 终端节点策略

下面是用于 Amazon RDS API 的终端节点策略示例。当附加到终端节点时，此策略会向所有委托人授予对列出的针对所有资源的 Amazon RDS API 操作的访问权限。

```
{
   "Statement":[
      {
         "Principal":"*",
         "Effect":"Allow",
         "Action":[
            "rds:CreateDBInstance",
            "rds:ModifyDBInstance",
            "rds:CreateDBSnapshot"
         ],
         "Resource":"*"
      }
   ]
}
```

示例：拒绝来自指定 Amazon 账户的所有访问的 VPC 终端节点策略

以下 VPC 端点策略会拒绝 Amazon 账户 123456789012 所有使用端点访问资源的权限。此策略允许来自其他账户的所有操作。

```
{
  "Statement": [
    {
      "Action": "*",
      "Effect": "Allow",
      "Resource": "*",
      "Principal": "*"
    },
    {
      "Action": "*",
      "Effect": "Deny",
      "Resource": "*",
      "Principal": { "AWS": [ "123456789012" ] }
     }
   ]
}
```

Amazon RDS 的安全最佳实践

使用 Amazon Identity and Access Management（IAM）账户可控制对 Amazon RDS API 操作（特别是创建、修改或删除 Amazon RDS 资源的操作）的访问权限。此类资源包括数据库实例、安全组和参数组。此外，使用 IAM 可控制执行常见管理任务的操作，例如备份和还原数据库实例。

- 为管理 Amazon RDS 资源的每个人（包括您自己）创建一个单独的用户。请勿使用 Amazon 根凭证管理 Amazon RDS 资源。
- 授予每位用户执行其职责所需的最低权限集。
- 使用 IAM 组有效地管理适用于多个用户的权限。
- 定期轮换您的 IAM 凭证。
- 将 Amazon Secrets Manager 配置为自动轮换 Amazon RDS 的密钥。有关更多信息，请参阅 Amazon Secrets Manager用户指南中的轮换 Amazon Secrets Manager 密钥。您也可以从 Amazon Secrets Manager 中以编程方式检索凭证。有关更多信息，请参阅 Amazon Secrets Manager 用户指南 中的检索密钥值。

有关 Amazon RDS 安全性的更多信息，请参阅 Amazon RDS 中的安全性。有关 IAM 的更多信息，请参阅。Amazon Identity and Access Management有关 IAM 最佳实践的信息，请参阅 IAM 最佳实践。

Amazon Security Hub 使用安全控件来评估资源配置和安全标准，以帮助您遵守各种合规框架。有关使用 Security Hub 评估 Lambda 资源的更多信息，请参阅《Amazon Security Hub 用户指南》中的 Amazon Relational Database Service 控件。

您可以监控 RDS 的使用情况，因为它与使用 Security Hub 的安全最佳实践有关。有关更多信息，请参阅什么是 Amazon Security Hub？。

使用 Amazon Web Services Management Console、Amazon CLI 或 RDS API 更改主用户的密码。如果使用另一个工具（如 SQL 客户端）来更改主用户密码，则可能会无意中取消用户的权限。

使用安全组控制访问权限

VPC 安全组控制传入和传出数据库实例的流量拥有的访问权限。默认情况下，将为您的数据库实例关闭网络访问。您可以在安全组中指定规则，允许从 IP 地址范围、端口或安全组进行访问。配置传入规则后，会向与该安全组关联的所有数据库实例应用相同的规则。您最多可以在一个安全组中指定 20 个规则。

VPC 安全组概述

每个 VPC 安全组规则都使特定的源能够访问 VPC 中与该 VPC 安全组关联的数据库实例。源可以是一个地址范围 (例如，203.0.113.0/24) 或另一个 VPC 安全组。指定作为源的 VPC 安全组后，就可以允许从使用此源 VPC 安全组的所有实例 (通常为应用程序服务器) 中传入流量。VPC 安全组可能具有管理入站和出站流量的规则。但是，出站流量规则通常不适用于数据库实例。仅在数据库实例充当客户端时，出站流量规则才适用。例如，出站流量规则适用于具有出站数据库链接的 Oracle 数据库实例。您必须使用 Amazon EC2 API 或 VPC 控制台中的 Security Group（安全组）选项创建 VPC 安全组。

当您为 VPC 安全组创建允许访问 VPC 中的实例的规则时，必须为规则允许访问的每个地址范围指定一个端口。例如，如果您要对 VPC 中的实例开启 Secure Shell（SSH）访问，则为指定的地址范围创建一条允许访问 TCP 端口 22 的规则。

您可以为 VPC 中不同的实例配置允许访问不同端口的多个 VPC 安全组。例如，您可以创建一个允许访问您的 VPC 中的 Web 服务器的 TCP 端口 80 的 VPC 安全组。之后，您可以创建另一个 VPC 安全组，来允许访问您的 VPC 中的 RDS for MySQL 数据库实例的 TCP 端口 3306。

有关 VPC 安全组的更多信息，请参阅 Amazon Virtual Private Cloud 用户指南 中的安全组。

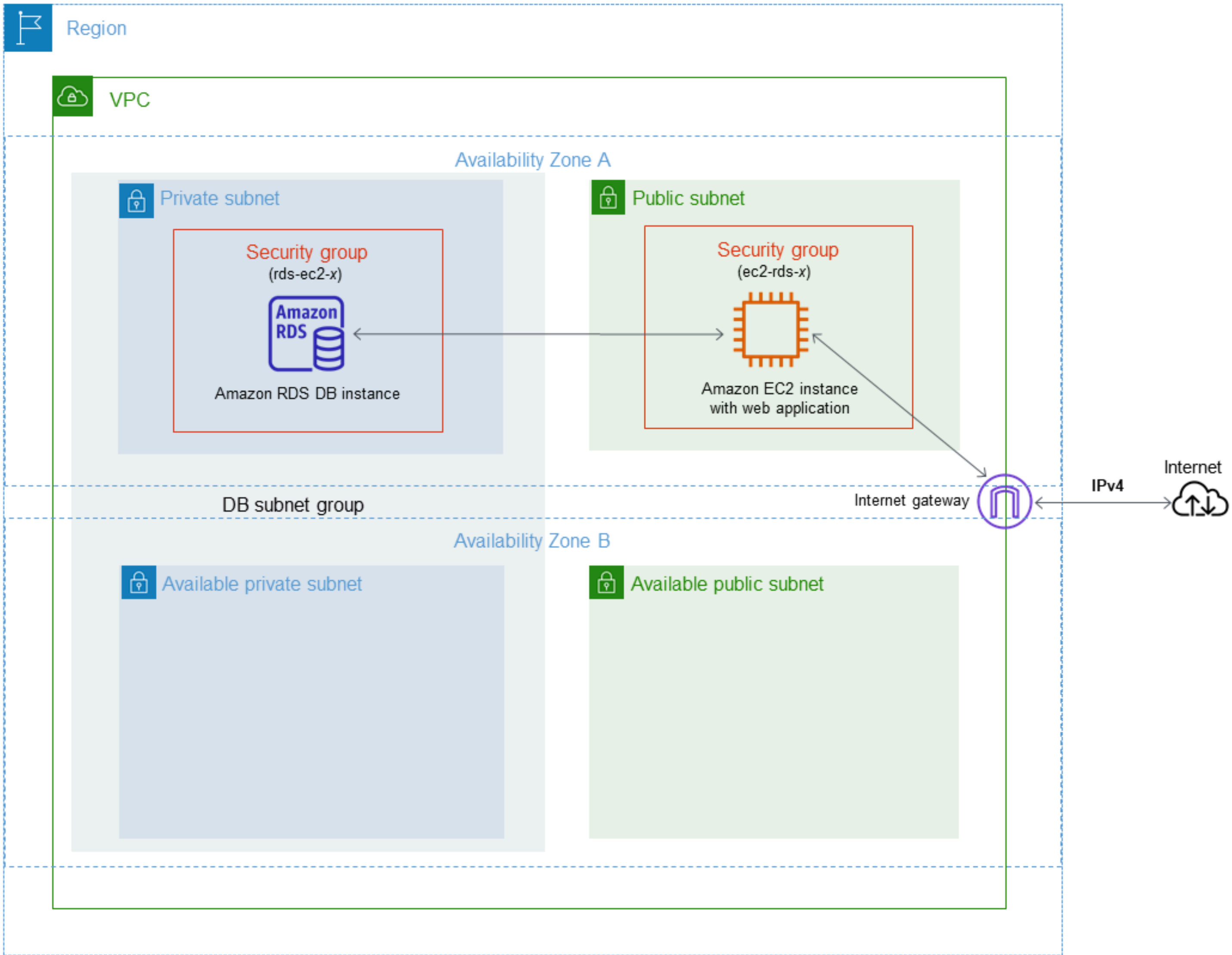
> **Note**
>
> 如果您的数据库实例位于 VPC 中但不可公开访问，则您还可以使用 Amazon Site-to-Site VPN 连接或 Amazon Direct Connect 连接从专用网络访问它。有关更多信息，请参阅 互联网络流量隐私。

安全组情况

VPC 中的数据库实例的一种常见用法是，与运行在相同 VPC 中 Amazon EC2 实例上的应用程序服务器分享数据，这些数据可通过 VPC 外的客户端应用程序进行访问。对于此情况，使用 Amazon Web Services Management Console上的 RDS 和 VPC 页面或 RDS 和 EC2 API 操作来创建必要的实例和安全组：

1. 创建一个 VPC 安全组 (例如，`sg-0123ec2example`)，然后定义使用客户端应用程序 IP 地址作为源的入站规则。通过此安全组，客户端应用程序可连接到使用此安全组的 VPC 中的 EC2 实例。
2. 创建一个适用于该应用程序的 EC2 实例，然后将该 EC2 实例添加到上一步中创建的 VPC 安全组 (`sg-0123ec2example`)。
3. 创建第二个 VPC 安全组 (例如，`sg-6789rdsexample`)，然后通过将步骤 1 中创建的 VPC 安全组 (`sg-0123ec2example`) 指定为源来创建一个新规则。
4. 创建一个新的数据库实例，然后将该数据库实例添加到在上一步骤中创建的 VPC 安全组（`sg-6789rdsexample`）。在创建数据库实例时，使用的端口号应与为您在步骤 3 中创建的 VPC 安全组（`sg-6789rdsexample`）规则指定的端口号相同。

下图说明了此情形。



有关针对此场景配置 VPC 的详细说明，请参阅教程：创建 VPC 以用于数据库实例（仅限 IPv4）。有关使用 VPC 的更多信息，请参阅 Amazon VPC 和 Amazon RDS。

创建 VPC 安全组

您可以使用 VPC 控制台为数据库实例创建 VPC 安全组。有关创建安全组的信息，请参阅 Amazon Virtual Private Cloud 用户指南 中的通过创建安全组提供对 VPC 中的数据库实例的访问和安全组。

将安全组与数据库实例关联

您可以使用 RDS 控制台的 Modify (修改) 选项、`ModifyDBInstance` Amazon RDS API 或 `modify-db-instance` Amazon CLI 命令将安全组与数据库实例关联。

以下 CLI 示例关联特定 VPC 安全组并从数据库实例中移除数据库安全组

```
aws rds modify-db-instance --db-instance-identifier dbName --vpc-security-group-ids sg-ID
```

有关修改数据库实例的信息，请参阅修改 Amazon RDS 数据库实例。有关从数据库快照还原数据库实例时的安全组注意事项，请参阅安全组注意事项。

> **Note**
>
> 如果将端口值配置为非原定设置值，RDS 控制台会显示数据库的不同安全组规则名称。

对于 RDS for Oracle 数据库实例，可以通过填充 Oracle Enterprise Manager Database Express（OEM）、适用于 Enterprise Manager Cloud Control 的 Oracle Management Agent（OEM 代理）和 Oracle 安全套接字层选项的安全组选项设置来关联其它安全组。在这种情况下，与数据库实例关联的安全组和选项设置都适用于该数据库实例。有关这些选项组的更多信息，请参阅 Oracle Enterprise Manager、适用于 Enterprise Manager Cloud Control 的 Oracle Management Agent 和 Oracle 安全套接字层。

主用户账户权限

在创建一个新的数据库实例时，您使用的默认主用户将获得针对该数据库实例的特定权限。您无法在创建数据库实例之后更改主用户名。

> **Important**
>
> 我们强烈建议不要直接在应用程序中使用主用户。请遵守使用数据库用户的最佳实践，按照您的应用程序所需的最少权限创建用户。

> **Note**
>
> 如果您不小心删除了主用户的权限，要想恢复权限，您可以修改数据库实例并设置新的主用户密码。有关修改数据库实例的更多信息，请参阅修改 Amazon RDS 数据库实例。

下表显示了主用户为每个数据库引擎获取的权限和数据库角色。

数据库引擎	系统权限	数据库角色
RDS for Db2	主用户被分配到 masterdba 组并被分配 master_user_role 。 SYSMON、具有 DATAACCESS 和 ACCCESSCTRL 的 DBADM、BINDADD、CONNECT、CREATETAB 、CREATE_SECURE_OBJECT 、EXPLAIN、IMPLICIT_SCHEMA 、LOAD、SQLADM、WLMADM	DBA,DBA_RESTRICTED ，DEVELOPER ,ROLE_NULLID_PACKAGES ，ROLE_PROCEDURES ,ROLE_TABLESPACES 有关更多信息，请参阅 Amazon RDS for Db2 默认角色。
RDS for MariaDB	SELECT,INSERT,UPDATE,DELETE, CREATE,DROP,RELOAD, PROCESS,REFERENCES ,INDEX, ALTER,SHOW DATABASES ,CREATE TEMPORARY TABLES ,LOCK TABLES, EXECUTE,REPLICATION CLIENT,CREATE VIEW ,SHOW VIEW,CREATE ROUTINE, ALTER ROUTINE,CREATE USER, EVENT,TRIGGER,REPLICATION SLAVE 从 RDS for MariaDB 版本 11.4 开始，主用户也将获得 SHOW CREATE ROUTINE 权限。	—
RDS for MySQL 8.0.36 及更高版本	SELECT,INSERT,UPDATE, DELETE,CREATE,DROP, RELOAD,PROCESS, REFERENCES ,INDEX,ALTER, SHOW DATABASES ,CREATE TEMPORARY TABLES,LOCK TABLES,EXECUTE, REPLICATION SLAVE ,REPLICATION CLIENT ,CREATE VIEW,SHOW VIEW,CREATE ROUTINE,ALTER ROUTINE,CREATE USER,EVENT,TRIGGER, CREATE ROLE,DROP ROLE, APPLICATION_PASSWORD_ADMIN ,ROLE_ADMIN ,SET_USER_ID ，XA_RECOVER_ADMIN	rds_superuser_role 有关 rds_superuser_role 的更多信息，请参阅 RDS for MySQL 的基于角色的权限模型。

数据库引擎	系统权限	数据库角色
低于 8.0.36 的 RDS for MySQL 版本	SELECT,INSERT,UPDATE, DELETE,CREATE,DROP, RELOAD,PROCESS, REFERENCES ,INDEX,ALTER, SHOW DATABASES ,CREATE TEMPORARY TABLES,LOCK TABLES,EXECUTE, REPLICATION CLIENT ,CREATE VIEW, SHOW VIEW,CREATE ROUTINE,ALTER ROUTINE,CREATE USER,EVENT, TRIGGER,REPLICATION SLAVE	—
RDS for PostgreSQL	CREATE ROLE,CREATE DB, PASSWORD VALID UNTIL INFINITY,CREATE EXTENSION ,ALTER EXTENSION ,DROP EXTENSION ,CREATE TABLESPACE ,ALTER <OBJECT> OWNER,CHECKPOINT , PG_CANCEL_BACKEND() , PG_TERMINATE_BACKEND() ,SELECT PG_STAT_REPLICATION ,EXECUTE PG_STAT_STATEMENTS_RESET() ,OWN POSTGRES_FDW_HANDLER() ,OWN POSTGRES_FDW_VALIDATOR() ,OWN POSTGRES_FDW , EXECUTE PG_BUFFERCACHE_PAGES() ,SELECT PG_BUFFERCACHE	RDS_SUPERUSER 有关 RDS_SUPERUSER 的更多信息，请参阅了解 PostgreSQL 角色和权限。

数据库引擎	系统权限	数据库角色
RDS for Oracle	ADMINISTER DATABASE TRIGGER ,ALTER DATABASE LINK,ALTER PUBLIC DATABASE LINK, AUDIT SYSTEM,CHANGE NOTIFICATION , DROP ANY DIRECTORY ,EXEMPT ACCESS POLICY,EXEMPT IDENTITY POLICY,EXEMPT REDACTION POLICY,FLASHBACK ANY TABLE, GRANT ANY OBJECT PRIVILEGE ,RESTRICTED SESSION ,SELECT ANY TABLE,UNLIMITED TABLESPACE	DBA Note DBA 角色不具有以下权限： ALTER DATABASE,ALTER SYSTEM, CREATE ANY DIRECTORY ,CREATE EXTERNAL JOB,CREATE PLUGGABLE DATABASE, GRANT ANY PRIVILEGE ,GRANT ANY ROLE,READ ANY FILE GROUP
Amazon RDS for Microsoft SQL Server	ADMINISTER BULK OPERATIONS ,ALTER ANY CONNECTION ,ALTER ANY CREDENTIAL , ALTER ANY EVENT SESSION,ALTER ANY LINKED SERVER,ALTER ANY LOGIN,ALTER ANY SERVER AUDIT,ALTER ANY SERVER ROLE, ALTER SERVER STATE,ALTER TRACE, CONNECT SQL,CREATE ANY DATABASE, VIEW ANY DATABASE,VIEW ANY DEFINITION ,VIEW SERVER STATE,ALTER ON ROLE SQLAgentOperatorRole	DB_OWNER（数据库级别角色）、PROCESSADMIN （服务器级别角色）、SETUPADMIN （服务器级别角色）、SQLAgentUserRole （数据库级别角色）

将服务相关角色用于 Amazon RDS

将 Amazon Identity and Access Management（IAM）服务相关角色用于 Amazon RDS。服务相关角色是一种与 Amazon RDS 直接关联的独特类型的 IAM 角色。服务相关角色由 Amazon RDS 预定义，并包含该服务代表您调用其他Amazon服务所需的一切权限。

服务相关角色可让您更轻松地使用 Amazon RDS，因为您不必手动添加必要的权限。Amazon RDS 定义其服务相关角色的权限，除非另外定义，否则只有 Amazon RDS 可以代入该角色。定义的权限包括信任策略和权限策略，以及不能附加到任何其他 IAM 实体的权限策略。

只有在首先删除角色的相关资源后，才能删除角色。这将保护您的 Amazon RDS 资源，因为您不会无意中删除对资源的访问权限。

有关支持服务相关角色的其他服务的信息，请参阅可与 IAM 搭配使用的 Amazon 服务，并查找 Service-Linked Role（服务相关角色）列中为 Yes（是）的服务。选择是和链接，查看该服务的服务相关角色文档。

Amazon RDS 的服务相关角色权限

Amazon RDS 使用名为 AWSServiceRoleForRDS 的服务相关角色允许 Amazon RDS 代表您的数据库实例调用 Amazon 服务。

AWSServiceRoleForRDS 服务相关角色信任以下服务以担任该角色：

- `rds.amazonaws.com`

此服务相关角色附加了一个名为 `AmazonRDSServiceRolePolicy` 的权限策略，授予其在您的账户中操作的权限。

有关此策略的更多信息，包括 JSON 策略文档，请参阅《Amazon 托管式策略参考指南》中的 AmazonRDSServiceRolePolicy。

> **Note**
>
> 您必须配置权限，允许 IAM 实体（如用户、组或角色）创建、编辑或删除服务相关角色。如果您遇到以下错误消息：
>
> Unable to create the resource. Verify that you have permission to create service linked role。Otherwise wait and try again later.
>
> 确保您已启用以下权限：

```
{
    "Action": "iam:CreateServiceLinkedRole",
    "Effect": "Allow",
    "Resource": "arn:aws:iam::*:role/aws-service-role/rds.amazonaws.com/
AWSServiceRoleForRDS",
    "Condition": {
        "StringLike": {
            "iam:AWSServiceName":"rds.amazonaws.com"
        }
    }
}
```

有关更多信息，请参阅《IAM 用户指南》中的服务相关角色权限。

为 Amazon RDS 创建服务相关角色

您无需手动创建服务相关角色。创建数据库实例时，Amazon RDS 将为您创建服务相关角色。

> **Important**
>
> 如果在 2017 年 12 月 1 日（从此时开始支持服务相关角色）之前已使用 Amazon RDS 服务，则 Amazon RDS 已在您的账户中创建 AWSServiceRoleForRDS 角色。要了解更多信息，请参阅我的 Amazon 账户中出现新角色。

如果您删除了此服务相关角色然后需要再次创建它，则可以使用相同的流程在您的账户中重新创建此角色。创建数据库实例时，Amazon RDS 将再次为您创建服务相关角色。

为 Amazon RDS 编辑服务相关角色

Amazon RDS 不允许您编辑 AWSServiceRoleForRDS 服务相关角色。创建服务相关角色后，您将无法更改角色的名称，因为可能有多种实体引用该角色。但是可以使用 IAM 编辑角色描述。有关更多信息，请参阅《IAM 用户指南》中的编辑服务相关角色。

删除 Amazon RDS 的服务相关角色

如果您不再需要使用某个需要服务相关角色的功能或服务，我们建议您删除该角色。这样您就没有未被主动监控或维护的未使用实体。但是，您必须先删除所有数据库实例才能删除服务相关角色。

清除服务相关角色

必须先确认服务相关角色没有活动会话并删除该角色使用的任何资源，然后才能使用 IAM 删除服务相关角色。

在 IAM 控制台中检查服务相关角色是否具有活动会话

1. 登录 Amazon Web Services Management Console，打开 IAM 控制台：https://console.aws.amazon.com/iam/。
2. 在 IAM 控制台的导航窗格中，选择角色。然后选择 AWSServiceRoleForRDS 角色的名称（不是复选框）。
3. 在所选角色的摘要页面上，选择上次访问时间选项卡。
4. 在上次访问时间选项卡上，查看服务相关角色的近期活动。

> **Note**
>
> 如果您不确定 Amazon RDS 是否正在使用 AWSServiceRoleForRDS 角色，可以尝试删除该角色。如果服务正在使用该角色，则删除操作会失败，并且您可以查看正在使用该角色的 Amazon 区域。如果该角色已被使用，则您必须等待会话结束，然后才能删除该角色。您无法撤销服务相关角色对会话的权限。

如果您要删除 AWSServiceRoleForRDS 角色，必须先删除您的所有数据库实例。

删除所有实例

使用以下过程之一删除每个实例。

删除一个实例 (控制台)

1. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择 Databases (数据库)。
3. 选择要删除的实例。
4. 对于 Actions，选择 Delete。
5. 如果系统提示您 Create final Snapshot? (是否创建最终快照?)，请选择 Yes (是) 或 No (否)。
6. 如果您在上一步中选择了 Yes (是)，请为 Final snapshot name (最终快照名称) 输入最终快照的名称。
7. 选择删除。

删除一个实例 (CLI)

请参阅 Amazon CLI Command Reference 中的 [delete-db-instance]。

删除一个实例 (API)

请参阅 Amazon RDS API Reference 中的 [DeleteDBInstance]。

使用 IAM 控制台、IAM CLI 或 IAM API 删除 AWSServiceRoleForRDS 服务相关角色。有关更多信息，请参阅《IAM 用户指南》中的删除服务相关角色。

Amazon RDS Custom 的服务相关角色权限

Amazon RDS Custom 使用名为 `AWSServiceRoleForRDSCustom` 的服务相关角色，来支持 RDS Custom 代表 RDS 数据库资源调用 Amazon 服务。

AWSServiceRoleForRDSCustom 服务相关角色信任以下服务以担任该角色：

- `custom.rds.amazonaws.com`

此服务相关角色附加了一个名为 `AmazonRDSCustomServiceRolePolicy` 的权限策略，授予其在您的账户中操作的权限。

为 RDS Custom 创建、编辑或删除服务相关角色的工作方式与 Amazon RDS 相同。有关更多信息，请参阅 Amazon 托管式策略：AmazonRDSCustomServiceRolePolicy。

> **Note**
>
> 您必须配置权限，允许 IAM 实体（如用户、组或角色）创建、编辑或删除服务相关角色。如果您遇到以下错误消息：
> Unable to create the resource. Verify that you have permission to create service linked role。Otherwise wait and try again later.
> 确保您已启用以下权限：

```
{
    "Action": "iam:CreateServiceLinkedRole",
    "Effect": "Allow",
    "Resource": "arn:aws:iam::*:role/aws-service-role/custom.rds.amazonaws.com/
AmazonRDSCustomServiceRolePolicy",
    "Condition": {
```

```
        "StringLike": {
            "iam:AWSServiceName":"custom.rds.amazonaws.com"
        }
    }
}
```

有关更多信息，请参阅《IAM 用户指南》中的服务相关角色权限。

Amazon RDS 测试版的服务相关角色权限

Amazon RDS 使用名为 `AWSServiceRoleForRDSBeta` 的服务相关角色，来支持 Amazon RDS 代表 RDS 数据库资源调用 Amazon 服务。

AWSServiceRoleForRDSBeta 服务相关角色信任以下服务来代入该角色：

- `rds.amazonaws.com`

此服务相关角色附加了一个名为 `AmazonRDSBetaServiceRolePolicy` 的权限策略，授予其在您的账户中操作的权限。有关更多信息，请参阅 Amazon 托管式策略：AmazonRDSBetaServiceRolePolicy。

> **Note**
>
> 您必须配置权限，允许 IAM 实体（如用户、组或角色）创建、编辑或删除服务相关角色。如果您遇到以下错误消息：
>
> Unable to create the resource. Verify that you have permission to create service linked role。Otherwise wait and try again later.
>
> 确保您已启用以下权限：

```
{
    "Action": "iam:CreateServiceLinkedRole",
    "Effect": "Allow",
    "Resource": "arn:aws:iam::*:role/aws-service-role/custom.rds.amazonaws.com/
AmazonRDSBetaServiceRolePolicy",
    "Condition": {
        "StringLike": {
            "iam:AWSServiceName":"custom.rds.amazonaws.com"
        }
```

```
    }
}
```

有关更多信息，请参阅《IAM 用户指南》中的服务相关角色权限。

Amazon RDS 预览版的服务相关角色

Amazon RDS 使用名为 `AWSServiceRoleForRDSPreview` 的服务相关角色，来支持 Amazon RDS 代表 RDS 数据库资源调用 Amazon 服务。

AWSServiceRoleForRDSPreview 服务相关角色信任以下服务来代入该角色：

- `rds.amazonaws.com`

此服务相关角色附加了一个名为 `AmazonRDSPreviewServiceRolePolicy` 的权限策略，授予其在您的账户中操作的权限。有关更多信息，请参阅 Amazon 托管式策略：AmazonRDSPreviewServiceRolePolicy。

> **Note**
>
> 您必须配置权限，允许 IAM 实体（如用户、组或角色）创建、编辑或删除服务相关角色。如果您遇到以下错误消息：
>
> Unable to create the resource. Verify that you have permission to create service linked role。Otherwise wait and try again later.
>
> 确保您已启用以下权限：

```
{
    "Action": "iam:CreateServiceLinkedRole",
    "Effect": "Allow",
    "Resource": "arn:aws:iam::*:role/aws-service-role/custom.rds.amazonaws.com/
AmazonRDSPreviewServiceRolePolicy",
    "Condition": {
        "StringLike": {
            "iam:AWSServiceName":"custom.rds.amazonaws.com"
        }
    }
}
```

有关更多信息，请参阅《IAM 用户指南》中的服务相关角色权限。

Amazon VPC 和 Amazon RDS

利用 Amazon Virtual Private Cloud（Amazon VPC），您可能能够在虚拟私有云（VPC）中启动 Amazon 资源，如 Amazon RDS 数据库实例。

使用 VPC 时，您的虚拟联网环境完全由您控制。您可以选择自己的 IP 地址范围、创建子网以及配置路由和访问控制列表。无需其他成本即可在 VPC 中运行数据库实例。

账户具有原定设置 VPC。除非您另行指定，否则所有新的数据库实例都将在原定设置 VPC 中创建。

主题

接下来，您可以查找与 Amazon RDS 数据库实例相关的 VPC 功能的讨论。有关 Amazon VPC 的更多信息，请参阅 Amazon VPC 入门指南和 Amazon VPC 用户指南。

在 VPC 中使用数据库实例

您的数据库实例位于虚拟私有云（VPC）内。Amazon VPC 是一个虚拟网络，在逻辑上与Amazon云中的其他虚拟网络相互隔离。使用 Amazon VPC，您可能能够将 Amazon 资源（例如，Amazon RDS 数据库实例或 Amazon EC2 实例），启动到 VPC 中。VPC 可以是您的账户附带的默认 VPC，也可以是您创建的 VPC。所有 VPC 均与您的 Amazon 账户关联。

默认 VPC 具有可用来隔离 VPC 内资源的三个子网。默认 VPC 还具有一个互联网网关，可用来从 VPC 外部访问 VPC 内部的资源。

有关涉及 VPC 内和 VPC 外的 Amazon RDS 数据库实例的场景的列表，请参阅在 VPC 中访问数据库实例的场景。

主题

- 共享子网
- Amazon RDS IP 寻址
- 对互联网隐藏 VPC 中的数据库实例
- 在 VPC 中创建数据库实例

在以下教程中，您可以学习创建可用于常见 Amazon RDS 场景的 VPC：

- 教程：创建 VPC 以用于数据库实例（仅限 IPv4）
- 教程：创建 VPC 以用于数据库实例（双堆栈模式）

在 VPC 中使用数据库实例

下面是一些有关在 VPC 中使用数据库实例的提示：

- 您的 VPC 必须至少有两个子网。这些子网必须位于要部署数据库实例的 Amazon Web Services 区域中两个不同的可用区。子网 是 VPC 的 IP 地址范围段，您可以指定子网，利用子网并根据安全和操作需要对数据库实例进行分组。

  对于多可用区部署，通过为 Amazon Web Services 区域中的两个或更多可用区定义子网，Amazon RDS 可在根据需要在另一个可用区中创建新的备用实例。即使是单可用区部署，您也确保要这样做，以防以后需要将它们转换为多可用区部署。

  > **Note**
  >
  > 本地扩展区的数据库子网组只能有一个子网。

- 如果要让 VPC 中的数据库实例实现公开访问，请确保开启 VPC 属性 DNS hostnames（DNS 主机名）和 DNS resolution（DNS 解析）。
- 您的 VPC 必须具有您创建的数据库子网组。您可通过指定创建的子网来创建数据库子网组。Amazon RDS 选择要与数据库实例关联的子网和该子网组中的 IP 地址。数据库实例使用包含该子网的可用区。
- 您的 VPC 必须具有允许访问数据库实例的 VPC 安全组。

  有关更多信息，请参阅 在 VPC 中访问数据库实例的场景。
- 每个子网必须包含足够大的 CIDR 数据块，以便在维护活动（包括故障转移和扩展计算）期间有可供 Amazon RDS 使用的备用 IP 地址。例如，诸如 10.0.0.1.0/24 和 10.0.1.0/24 的范围通常足够大。

- VPC 的 instance tenancy 属性可能为 default 或 dedicated。所有默认 VPC 的“instance tenancy”属性设置为“default”，则默认的 VPC 可支持任何数据库实例类。

  如果您选择将数据库实例放在“instance tenancy”属性设置为“dedicated”的专用 VPC 中，则数据库实例的数据库实例类必须是已批准的 Amazon EC2 专用实例类型之一。例如，r5.large EC2 专用实例对应于 db.r5.large 数据库实例类。有关 VPC 中实例租期的信息，请参阅 Amazon Elastic Compute Cloud 用户指南 中的专用实例。

> **Note**
>
> 当您将实例租赁属性设置为专用于数据库实例时，它不能保证数据库实例将在专属主机上运行。

- 将选项组分配给数据库实例时，它与该数据库实例的 VPC 相关联。这种关联意味着，如果您尝试将数据库实例还原到不同的 VPC 中，则无法使用分配给该数据库实例的选项组。
- 如果将数据库实例还原到不同的 VPC，请确保将默认选项组分配给数据库实例，分配与该 VPC 关联的选项组，或创建新选项组并将其分配给数据库实例。对于持久性或永久性选项（如 Oracle TDE），在将数据库实例还原到不同的 VPC 中时，您必须创建包含该持久性或永久性选项的新选项组。

使用数据库子网组

子网 是您指定的用来根据安全和操作需要对资源进行分组的 VPC 的 IP 地址范围段。数据库子网组 是您在 VPC 中创建并随后指定给数据库实例的子网（通常为私有子网）的集合。使用数据库子网组，您可以在使用 Amazon CLI 或 RDS API 创建数据库实例时指定特定的 VPC。如果您使用控制台，则可以选择要使用的 VPC 和子网组。

每个数据库子网组应包含给定 Amazon Web Services 区域中至少两个可用区的子网。在 VPC 中创建数据库实例时，为其选择数据库子网组。从数据库子网组中，Amazon RDS 会选择子网和该子网中的 IP 地址（与数据库集群中的相关联）。数据库使用包含该子网的可用区。Amazon RDS 始终从具有可用 IP 地址空间的子网中分配 IP 地址。

如果某一多可用区部署的主数据库实例出现故障，Amazon RDS 可以提升相应的备用实例，并随后在另外一个可用区中使用子网的 IP 地址来创建一个新的备用实例。

数据库子网组中的子网要么是公有子网，要么是私有子网。根据您为网络访问控制列表（网络 ACL）和路由表设置的配置，这些子网是公有子网或私有子网。要使数据库实例可公开访问，其数据库子网组

中的所有子网必须均为公有子网。如果与可公开访问的数据库实例关联的子网从公有更改为私有，则可能会影响数据库实例的可用性。

要创建支持双堆栈模式的数据库子网组，请确保添加到该数据库子网组的每个子网都有一个与之关联的 Internet 协议版本 6 (IPv6) CIDR 块。有关更多信息，请参阅 Amazon VPC 用户指南中的 Amazon RDS IP 寻址和迁移到 IPv6。

> **Note**
>
> 本地扩展区的数据库子网组只能有一个子网。

Amazon RDS 在 VPC 中创建数据库实例时，它使用数据库子网组中的 IP 地址将网络接口分配给数据库实例。不过，我们强烈建议您使用域名系统（DNS）名称来连接数据库实例。之所以建议这样做，是因为底层 IP 地址在失效转移期间会发生变化。

> **Note**
>
> 对于在 VPC 中运行的每个数据库实例，请确保数据库子网组的每个子网中预留至少一个地址，以供 Amazon RDS 用来执行恢复操作。

共享子网

您可以在共享 VPC 中创建数据库实例。

使用共享 VPC 时需要记住的一些注意事项：

- 您可以将数据库实例从共享 VPC 子网移动到非共享 VPC 子网，反之亦然。
- 共享 VPC 中的参与者必须在 VPC 中创建安全组，才能允许他们创建数据库实例。
- 共享 VPC 中的拥有者和参与者可以使用 SQL 查询访问数据库。但是，只有资源的创建者才能对该资源进行任何 API 调用。

Amazon RDS IP 寻址

IP 地址使 VPC 中的资源能够相互通信以及与 Internet 上的资源进行通信。Amazon RDS 同时支持 IPv4 和 IPv6 寻址协议。默认情况下，Amazon RDS 和 Amazon VPC 使用 IPv4 寻址协议。您无法关

闭这种行为。创建 VPC 时，请确保指定 IPv4 CIDR 块 (一系列私有 IPv4 地址)。您可以选择将 IPv6 CIDR 块分配给您的 VPC 和子网，并将来自该块的 IPv6 地址分配给您子网中的数据库实例。

对 IPv6 协议的支持扩展了支持的 IP 地址数量。通过使用 IPv6 协议，您可以确保有足够的可用地址来应对 Internet 的未来发展。新的和现有 RDS 资源可以在 VPC 内使用 IPv4 和 IPv6 地址。在应用程序不同部分使用的两个协议之间配置、保护和转换网络流量可能会产生运营开销。您可以对 Amazon RDS 资源的 IPv6 协议进行标准化，以简化网络配置。

主题

- IPv4 地址
- IPv6 地址
- 双堆栈模式

IPv4 地址

创建 VPC 时，您必须以 CIDR 块（如 `10.0.0.0/16`）的形式为 VPC 指定一个 IPv4 地址范围。数据库子网组 定义此 CIDR 块中可供数据库实例使用的 IP 地址范围。这些 IP 地址可以是私有地址，也可以是公有地址。

私有 IPv4 地址是指无法通过 Internet 访问的 IP 地址。您可以使用私有 IPv4 地址在同一 VPC 中的数据库实例和其他资源（如 Amazon EC2 实例）之间进行通信。每个数据库实例都有一个用于在 VPC 中通信的私有 IP 地址。

公有 IP 地址是指可通过 Internet 访问的 IPv4 地址。您可以使用公有地址在数据库实例和 Internet 上的资源（如 SQL 客户端）之间进行通信。您可以控制数据库实例是否接收公有 IP 地址。

有关说明如何创建可用于常见 Amazon RDS 场景的 VPC（只有私有 IPv4 地址）的教程，请参阅 教程：创建 VPC 以用于数据库实例（仅限 IPv4）。

IPv6 地址

您可以选择向 VPC 和子网关联 IPv6 CIDR 块，然后将此块中的 IPv6 地址分配给 VPC 中的资源。每个 IPv6 地址都是全局唯一的。

我们将自动从 Amazon 的 IPv6 地址池中为您的 VPC 分配 IPv6 CIDR 块。您不能自行选择范围。

连接到 IPv6 地址时，请确保满足以下条件：

- 客户端配置为允许通过 IPv6 进行客户端到数据库的通信。
- 数据库实例使用的 RDS 安全组已正确配置，允许通过 IPv6 进行客户端到数据库的通信。

- 客户端操作系统堆栈允许 IPv6 地址上有流量，操作系统驱动程序和库已配置为选择正确的默认数据库实例终端节点（IPv4 或 IPv6）。

有关 IPv6 的更多信息，请参阅 Amazon VPC 用户指南中的 IP 寻址。

双堆栈模式

当数据库实例可以通过 IPv4 和 IPv6 寻址协议进行通信时，它在双堆栈模式下运行。因此，资源可以通过 IPv4 和/或 IPv6 与数据库实例进行通信。RDS 禁用互联网网关访问私有双堆栈模式数据库实例的 IPv6 端点。RDS 这样做是为了确保您的 IPv6 端点是私有的，并且只能从您的 VPC 内部对其进行访问。

主题

有关说明如何创建您可用于常见 Amazon RDS 场景的 VPC（具有 IPv4 和 IPv6 地址）的教程，请参阅 教程：创建 VPC 以用于数据库实例（双堆栈模式）。

双堆栈模式和数据库子网组

要使用双堆栈模式，请确保与数据库实例关联的数据库子网组中的每个子网都具有与之关联的 IPv6 CIDR 块。您可以创建新的数据库子网组或修改现有数据库子网组来满足此要求。当数据库实例处于双堆栈模式后，客户端可以与它正常连接。确保准确配置客户端安全防火墙和 RDS 数据库实例安全组，以允许通过 IPv6 的流量。要进行连接，客户端使用数据库实例的端点。。客户端应用程序可以指定连接到数据库时首选哪种协议。在双堆栈模式下，数据库实例会检测客户端的首选网络协议（IPv4 或 IPv6），并使用该协议进行连接。

如果某数据库子网组因子网删除或 CIDR 断开关联而停止支持双堆栈模式，则与该数据库子网组关联的数据库实例存在网络状态不兼容的风险。此外，创建新的双堆栈模式数据库实例时，您不能使用数据库子网组。

要使用 Amazon Web Services Management Console 确定数据库子网组是否支持双堆栈模式，请查看数据库子网组的详细信息页面上的 Network type（网络类型）。要使用 Amazon CLI 确

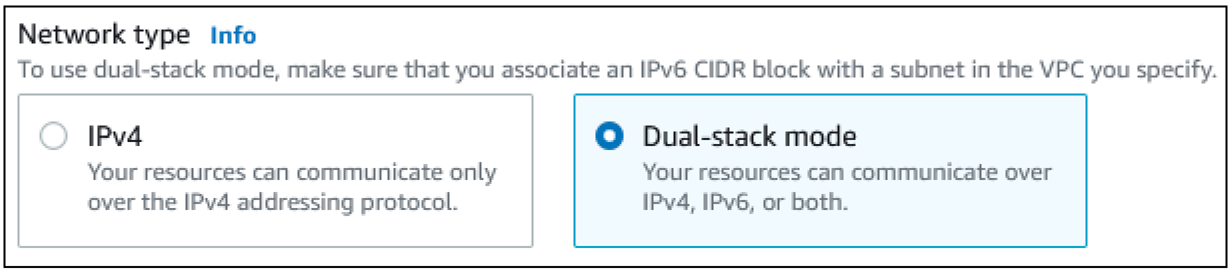
定数据库子网组是否支持双堆栈模式，请运行 [describe-db-subnet-groups] 命令并查看输出中的 `SupportedNetworkTypes`。

只读副本被视为独立的数据库实例，并且可以具有与主数据库实例不同的网络类型。如果您更改只读副本的主数据库实例的网络类型，则只读副本不会受到影响。当您还原数据库实例时，可以将其还原为支持的任何网络类型。

使用双堆栈模式数据库实例

创建或修改数据库实例时，您可以指定双堆栈模式，以允许您的资源通过 IPv4 和/或 IPv6 与数据库实例进行通信。

当您使用 Amazon Web Services Management Console 创建或修改数据库实例时，可以在 Network type（网络类型）部分中指定双堆栈模式。下图显示了控制台中的 Network type（网络类型）部分。



当您使用 Amazon CLI 创建或修改数据库实例时，请将 `--network-type` 选项设置为 `DUAL` 以使用双堆栈模式。当您使用 RDS API 创建或修改数据库实例时，请将 `NetworkType` 参数设置为 `DUAL` 以使用双堆栈模式。当您修改数据库实例的网络类型时，可能会出现停机。如果指定的数据库引擎版本或数据库子网组不支持双堆栈模式，则返回 `NetworkTypeNotSupported` 错误。

有关创建数据库实例的更多信息，请参阅 [创建 Amazon RDS 数据库实例]。有关修改数据库实例的更多信息，请参阅 [修改 Amazon RDS 数据库实例]。

要使用控制台确定数据库实例是否处于双堆栈模式，请查看数据库实例的 Connectivity & security（连接和安全）选项卡上的 Network type（网络类型）。

修改仅限 IPv4 的数据库实例以使用双堆栈模式

您可以修改仅限 IPv4 的数据库实例以使用双堆栈模式。为此，请更改数据库实例的网络类型。修改可能会导致停机。

建议您在维护时段内更改 Amazon RDS 数据库实例的网络类型。目前，不支持将新实例的网络类型设置为双堆栈模式。您可以使用 `modify-db-instance` 命令手动设置网络类型。

修改数据库实例以使用双堆栈模式之前，请确保其数据库子网组支持双堆栈模式。如果与数据库实例关联的数据库子网组不支持双堆栈模式，请在修改数据库实例时指定支持该模式的其他数据库子网组。修改数据库实例的数据库子网组可能会导致停机。

如果在将数据库实例更改为使用双堆栈模式之前，修改数据库实例的数据库子网组，请确保该数据库子网组在更改前后对数据库实例有效。

对于 RDS for PostgreSQL、RDS for MySQL、RDS for Oracle 和 RDS for MariaDB 单可用区实例，我们建议您只使用设置为 `DUAL` 的 `--network-type` 参数运行 [modify-db-instance](https://docs.aws.amazon.com/cli/latest/reference/rds/modify-db-instance.html) 命令，以将网络更改为双堆栈模式。在同一 API 调用中将其他参数与 `--network-type` 参数一起添加可能会导致停机。要修改多个参数，请确保网络类型修改已成功完成，然后使用其他参数发送另一个 `modify-db-instance` 请求。

如果您仅使用 `--network-type` 参数或在 modify-db-instance 命令中组合使用参数，则修改 RDS for PostgreSQL、RDS for MySQL、RDS for Oracle 和 RDS for MariaDB 多可用区数据库实例的网络类型会导致短暂停机并触发失效转移。

如果仅使用 `--network-type` 参数或在 `modify-db-instance` 命令中组合使用参数，则修改 RDS for SQL Server 单可用区或多可用区数据库实例上的网络类型会导致停机。网络类型修改会导致 SQL Server 多可用区实例发生失效转移。

如果更改后无法连接到数据库实例，请确保已准确配置客户端和数据库安全防火墙和路由表，以允许流量流向选定网络（IPv4 或 IPv6）上的数据库。您可能还需要修改操作系统参数、库或驱动程序才能使用 IPv6 地址进行连接。

当您将数据库实例修改为使用双堆栈模式时，不能存在从单可用区部署到多可用区部署或从多可用区部署到单可用区部署的待处理更改。

修改仅限 IPv4 的数据库实例以使用双堆栈模式

1. 修改数据库子网组以支持双堆栈模式，或者创建支持双堆栈模式的数据库子网组：

   a. 将 IPv6 CIDR 块与 VPC 关联。

      有关说明，请参阅《Amazon VPC 用户指南》中的将 IPv6 CIDR 块添加到 VPC。

   b. 将 IPv6 CIDR 块附加到数据库子网组中的所有子网。

      有关说明，请参阅《Amazon VPC 用户指南》中的将 IPv6 CIDR 块添加到子网。

   c. 确认数据库子网组支持双堆栈模式。

如果您使用 Amazon Web Services Management Console，请选择数据库子网组，并确保 Supported network types（支持的网络类型）值为 Dual, IPv4（双，IPv4）。

如果您使用 Amazon CLI，请运行 describe-db-subnet-groups 命令，并确保数据库实例的 `SupportedNetworkType` 值为 `Dual, IPv4`。

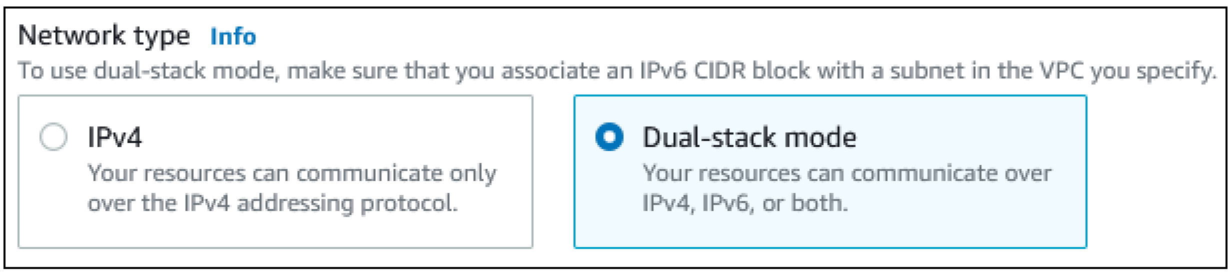
2. 修改与数据库实例关联的安全组以允许与数据库建立 IPv6 连接，或者新建允许 IPv6 连接的安全组。

   有关说明，请参阅 Amazon VPC 用户指南中的安全组规则。

3. 修改数据库实例以支持双堆栈模式。为此，请将 Network type（网络类型）设置为 Dual-stack mode（双堆栈模式）。

   如果您使用控制台，请确保以下设置正确：

   - Network type（网络类型）– Dual-stack mode（双堆栈模式）

   - DB subnet group（数据库子网组）– 在上一步中配置的数据库子网组
   - Security group（安全组）– 在上一步中配置的安全组

   如果您使用 Amazon CLI，请确保以下设置正确：

   - `--network-type` – `dual`
   - `--db-subnet-group-name` – 在上一步中配置的数据库子网组
   - `--vpc-security-group-ids` – 在上一步中配置的 VPC 安全组

   例如：

```
aws rds modify-db-instance --db-instance-identifier my-instance --network-type
 "DUAL"
```

4. 确认数据库实例支持双堆栈模式。

如果您使用控制台，请对于数据库实例选择 Connectivity & security（连接和安全）选项卡。在该选项卡上，确保 Network type（网络类型）值为 Dual-stack mode（双堆栈模式）。

如果您使用 Amazon CLI，请运行 describe-db-instances 命令，并确保数据库实例的 `NetworkType` 值为 `dual`。

在数据库实例端点上运行 `dig` 命令，以标识与其关联的 IPv6 地址。

```
dig db-instance-endpoint AAAA
```

使用数据库实例端点（而不是 IPv6 地址）连接到数据库实例。

区域和版本可用性

功能可用性和支持因每个数据库引擎的特定版本以及 Amazon Web Services 区域而异。有关双堆栈模式下版本和区域可用性的更多信息，请参阅支持 Amazon RDS 中双堆栈模式的区域和数据库引擎。

双堆栈网络数据库实例的限制

以下限制适用于双堆栈网络数据库实例：

- 数据库实例不能仅使用 IPv6 协议。它们可以仅使用 IPv4，也可以同时使用 IPv4 和 IPv6 协议（双堆栈模式）。
- Amazon RDS 不支持本机 IPv6 子网。
- 使用双堆栈模式的数据库实例必须是私有的。它们不可公开访问。
- 对于 RDS for SQL Server，使用 Always On AG 可用性组侦听器终端节点的双堆栈模式数据库实例仅显示 IPv4 地址。
- 不能将 RDS 代理与双堆栈模式数据库实例一起使用。
- 不能将双堆栈模式与 RDS on Amazon Outposts 数据库实例一起使用。
- 不能将双堆栈模式与本地区域中的数据库实例一起使用。

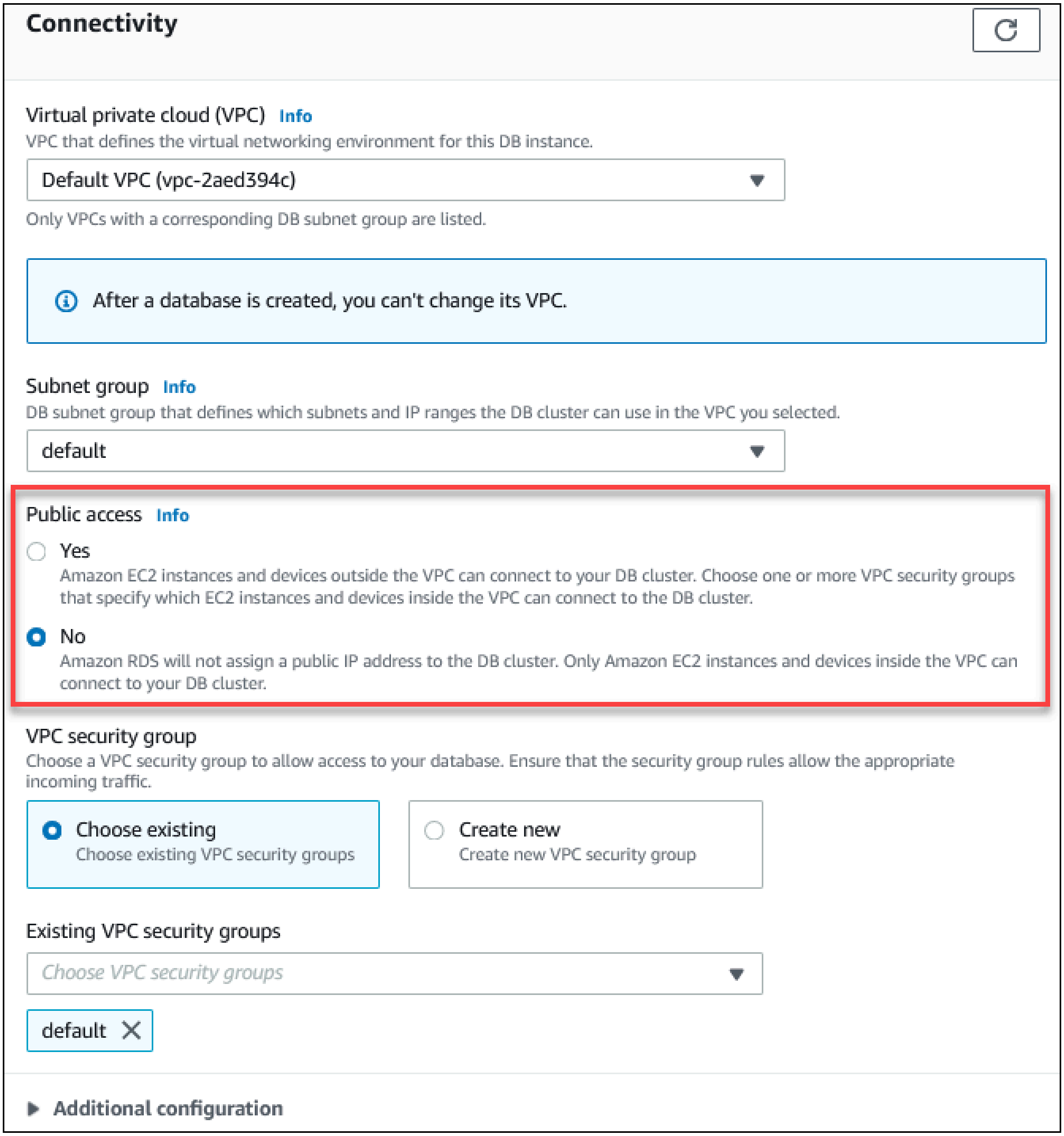
对互联网隐藏 VPC 中的数据库实例

一个常见的 Amazon RDS 场景是具有一个 VPC，其中有一个 Amazon EC2 实例（具有一个面向公众的 Web 应用程序）和一个数据库实例（具有一个不能公开访问的数据库）。例如，您可创建包含公有子网和私有子网的 VPC。可将充当 Web 服务器的 EC2 实例部署在公有子网中。数据库实例部署在私

有子网中。在此部署方案中，只有 Web 服务器才能访问数据库实例。有关此方案的说明，请参阅 VPC 中的数据库实例由同一 VPC 中的 Amazon EC2 实例访问。

当您在 VPC 中启动数据库实例时，该数据库实例具有用于 VPC 内流量的私有 IP 地址。此私有 IP 地址不可公开访问。您可以使用 Public access（公有访问权限）选项指定数据库实例除了私有 IP 地址之外是否还具有公有 IP 地址。如果将数据库实例指定为可公开访问，则其 DNS 端点解析为 VPC 内部的私有 IP 地址。它从 VPC 外部解析为公有 IP 地址。对数据库实例的访问最终由它使用的安全组控制。如果分配给数据库实例的安全组不包含允许公有访问的入站规则，则不允许该公有访问。此外，要使数据库实例可公开访问，其数据库子网组中的子网必须具有互联网网关。有关更多信息，请参阅无法连接到 Amazon RDS 数据库实例

您可以通过修改 Public access（公开访问）选项，修改数据库实例来开启或关闭公开可访问性。下图显示了其他连接配置部分中的公开访问选项。要设置此选项，请打开连接部分中的其他连接配置部分。



有关修改数据库实例以设置公开访问选项的信息，请参阅修改 Amazon RDS 数据库实例。

在 VPC 中创建数据库实例

以下过程帮助您在 VPC 中创建数据库实例。要使用默认 VPC，可以从步骤 2 开始，并使用已经为您创建的 VPC 和数据库子网组。如果您想创建其他 VPC，则可创建新的 VPC。

> **Note**
>
> 如果要让 VPC 中的数据库实例实现公开访问，则必须通过启用 VPC 属性 DNS hostnames（DNS 主机名）和 DNS resolution（DNS 解析）更新 VPC 的 DNS 信息。有关更新 VPC 实例的 DNS 信息的信息，请参阅更新对 VPC 的 DNS 支持。

执行以下步骤可在 VPC 中创建数据库实例：

- 步骤 1：创建 VPC
- 步骤 2：创建数据库子网组
- 步骤 3：创建 VPC 安全组
- 步骤 4：在 VPC 中创建数据库实例

步骤 1：创建 VPC

创建一个 VPC，该 VPC 具有的子网位于至少两个可用区内。您在创建数据库子网组时将使用这些子网。如果您拥有默认 VPC，则系统会在该 Amazon Web Services 区域中的每个可用区中自动为您创建子网。

有关更多信息，请参阅创建包含公有子网和私有子网的 VPC，或参阅《Amazon VPC 用户指南》中的创建 VPC。

步骤 2：创建数据库子网组

数据库子网组是您为 VPC 创建、然后为数据库实例指定的子网（通常为私有子网）的集合。数据库子网组可让您在使用 Amazon CLI 或 RDS API 创建数据库实例时指定特定的 VPC。如果您使用控制台，您就只能选择要使用的 VPC 和子网。每个数据库子网组必须至少包含给定 Amazon Web Services 区域中至少两个可用区的一个子网。作为最佳实践，每个数据库子网组应至少包含 Amazon Web Services 区域中每个可用区的一个子网。

对于多可用区部署，为 Amazon Web Services 区域中的所有可用区定义子网，使得 Amazon RDS 可以在必要时在另一个可用区中创建新的备用副本。即使是单可用区部署，您也可以遵循此最佳实践，因为将来可能会将其转换为多可用区部署。

为使数据库实例可公开访问，数据库子网组中的子网必须具有互联网网关。有关子网的互联网网关的更多信息，请参阅《Amazon VPC 用户指南》中的使用互联网网关将子网连接到互联网。

> **Note**
>
> 本地扩展区的数据库子网组只能有一个子网。

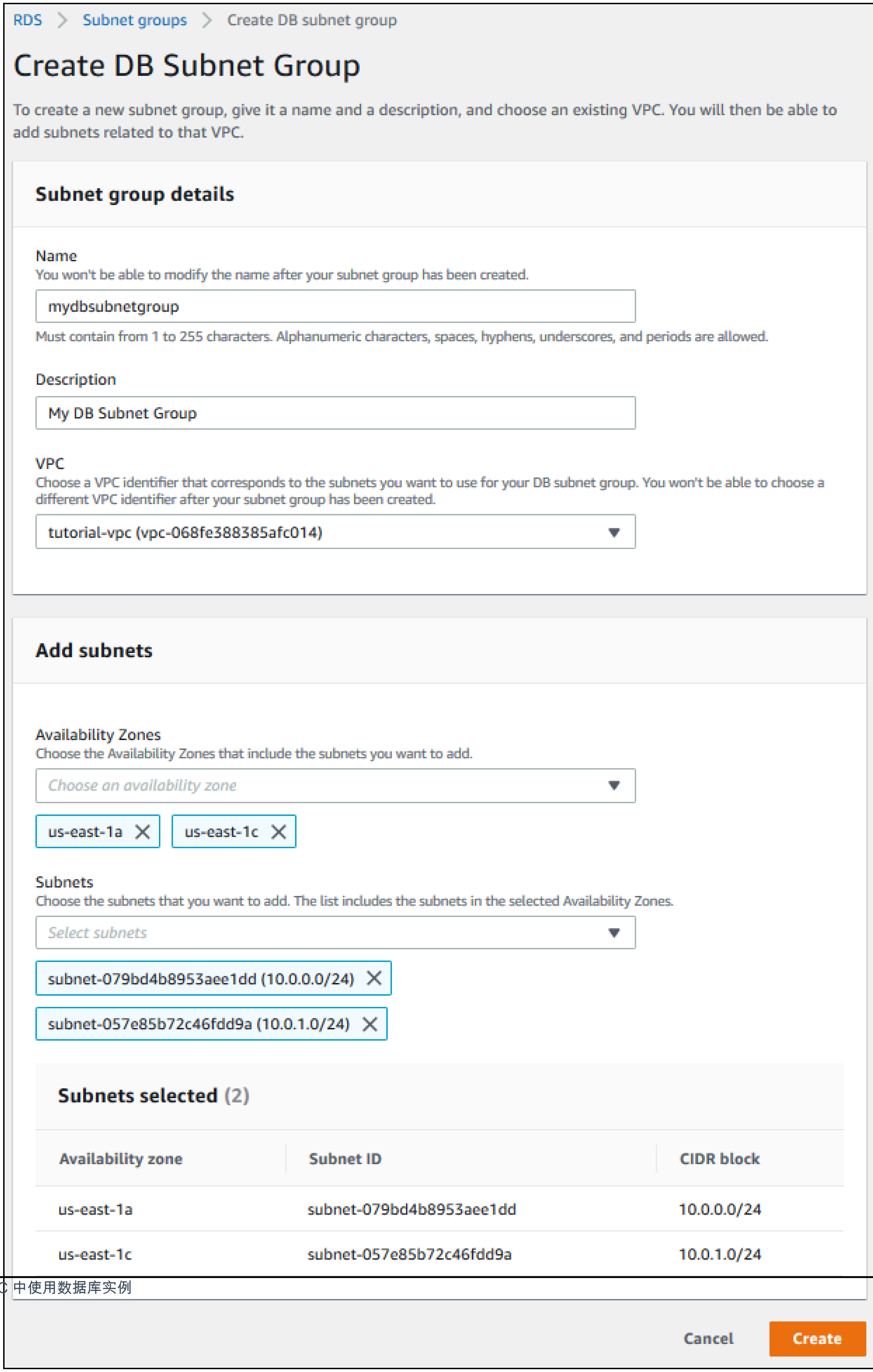
在 VPC 中创建数据库实例时，您可以选择数据库子网组。Amazon RDS 会选择要与数据库实例关联的子网和该子网中的 IP 地址。如果不存在数据库子网组，Amazon RDS 会在您创建数据库实例时创建默认子网组。Amazon RDS 使用该 IP 地址创建弹性网络接口并将其关联到您的数据库实例。数据库实例使用包含该子网的可用区。

对于多可用区部署，通过为 Amazon Web Services 区域 中的两个或更多可用区定义子网，Amazon RDS 可在需求增加时在另一个可用区中创建新的备用实例。即使是单可用区部署，您也需要这样做，以防以后需要将它们转换为多可用区部署。

在此步骤中，您创建一个数据库子网组，然后添加为 VPC 创建的子网。

创建数据库子网组

1. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在导航窗格中，选择子网组。
3. 选择创建数据库子网组。
4. 对于名称，键入您的数据库子网组的名称。
5. 对于描述，键入您的数据库子网组的描述。
6. 对于 VPC，选择默认 VPC 或您创建的 VPC。
7. 在添加子网部分中，从可用区中选择包含子网的可用区，然后从子网中选择子网。


RDS > Subnet groups > Create DB subnet group
Create DB Subnet Group
To create a new subnet group, give it a name and a description, and choose an existing VPC. You will then be able to add subnets related to that VPC.
Subnet group details
Name
You won't be able to modify the name after your subnet group has been created.
mydbsubnetgroup
Must contain from 1 to 255 characters. Alphanumeric characters, spaces, hyphens, underscores, and periods are allowed.
Description
My DB Subnet Group
VPC
Choose a VPC identifier that corresponds to the subnets you want to use for your DB subnet group. You won't be able to choose a different VPC identifier after your subnet group has been created.
tutorial-vpc (vpc-068fe388385afc014)
Add subnets
Availability Zones
Choose the Availability Zones that include the subnets you want to add.
Choose an availability zone
us-east-1a
us-east-1c
Subnets
Choose the subnets that you want to add. The list includes the subnets in the selected Availability Zones.
Select subnets
subnet-079bd4b8953aee1dd (10.0.0.0/24)
subnet-057e85b72c46fdd9a (10.0.1.0/24)
Subnets selected (2)
Availability zone
Subnet ID
CIDR block
us-east-1a
subnet-079bd4b8953aee1dd
10.0.0.0/24
us-east-1c
subnet-057e85b72c46fdd9a
10.0.1.0/24
Cancel
Create

> **Note**
>
> 如果已启用本地扩展区，则可以在创建数据库子网组页面上选择可用区组。在这种情况下，请选择可用区组、可用区和子网。

8. 选择创建。

   您的新数据库子网组显示在 RDS 控制台的数据库子网组列表中。可选择该数据库子网组，在窗口底部的详细信息窗格中查看详细信息，其中包括与该组关联的所有子网。

步骤 3：创建 VPC 安全组

创建数据库实例前，您可以创建要与数据库实例相关联的 VPC 安全组。如果您不创建 VPC 安全组，则可以在创建数据库实例时使用默认安全组。有关如何为数据库实例创建安全组的说明，请参阅为私有数据库实例创建 VPC 安全组，或参阅《Amazon VPC 用户指南》中的使用安全组控制到资源的流量。

步骤 4：在 VPC 中创建数据库实例

在此步骤中，创建一个数据库实例，并使用在之前的步骤中创建的 VPC 名称、数据库子网组和 VPC 安全组。

> **Note**
>
> 如果要让 VPC 中的数据库实例实现公开访问，必须启用 VPC 属性 DNS hostnames（DNS 主机名）和 DNS resolution（DNS 解析）。有关更多信息，请参阅《Amazon VPC 用户指南》中的 VPC 的 DNS 属性。

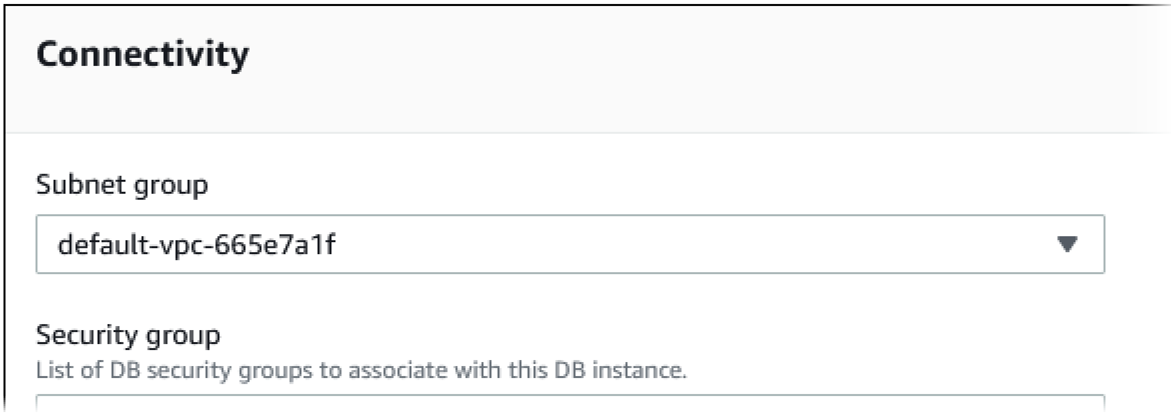
有关如何创建数据库实例的详细信息，请参阅创建 Amazon RDS 数据库实例。

在 Connectivity（连接）部分出现提示时，输入 VPC 名称、数据库子网组和 VPC 安全组。

更新数据库实例的 VPC

您可以使用 Amazon Web Services Management Console将您的数据库实例移动到其他 VPC。

有关修改 数据库实例的信息，请参阅修改 Amazon RDS 数据库实例。在修改页面的 Connectivity（连接）部分（如下所示）中，为 DB subnet group（数据库子网组）输入新的数据库子网组。新子网组必须是新 VPC 中的子网组。



如果满足以下条件，便无法更改数据库实例的 VPC：

- 数据库实例位于多个可用区中。您可以将数据库实例转换为单个可用区，将其移动到新的 VPC，然后将其转换回多可用区数据库实例。有关更多信息，请参阅 配置和管理 Amazon RDS 的多可用区部署。
- 数据库实例拥有一个或多个只读副本。您可以删除只读副本，将数据库实例移动到新 VPC，然后再次添加只读副本。有关更多信息，请参阅 使用数据库实例只读副本。
- 数据库实例为只读副本。您可以提升只读副本，然后将独立数据库实例移动到新 VPC。有关更多信息，请参阅 将只读副本提升为独立的数据库实例。
- 目标 VPC 中的子网组在数据库实例的可用区中没有子网。您可以将数据库实例可用区中的子网添加到数据库子网组，然后将数据库实例移动到新 VPC。有关更多信息，请参阅 使用数据库子网组。

在 VPC 中访问数据库实例的场景

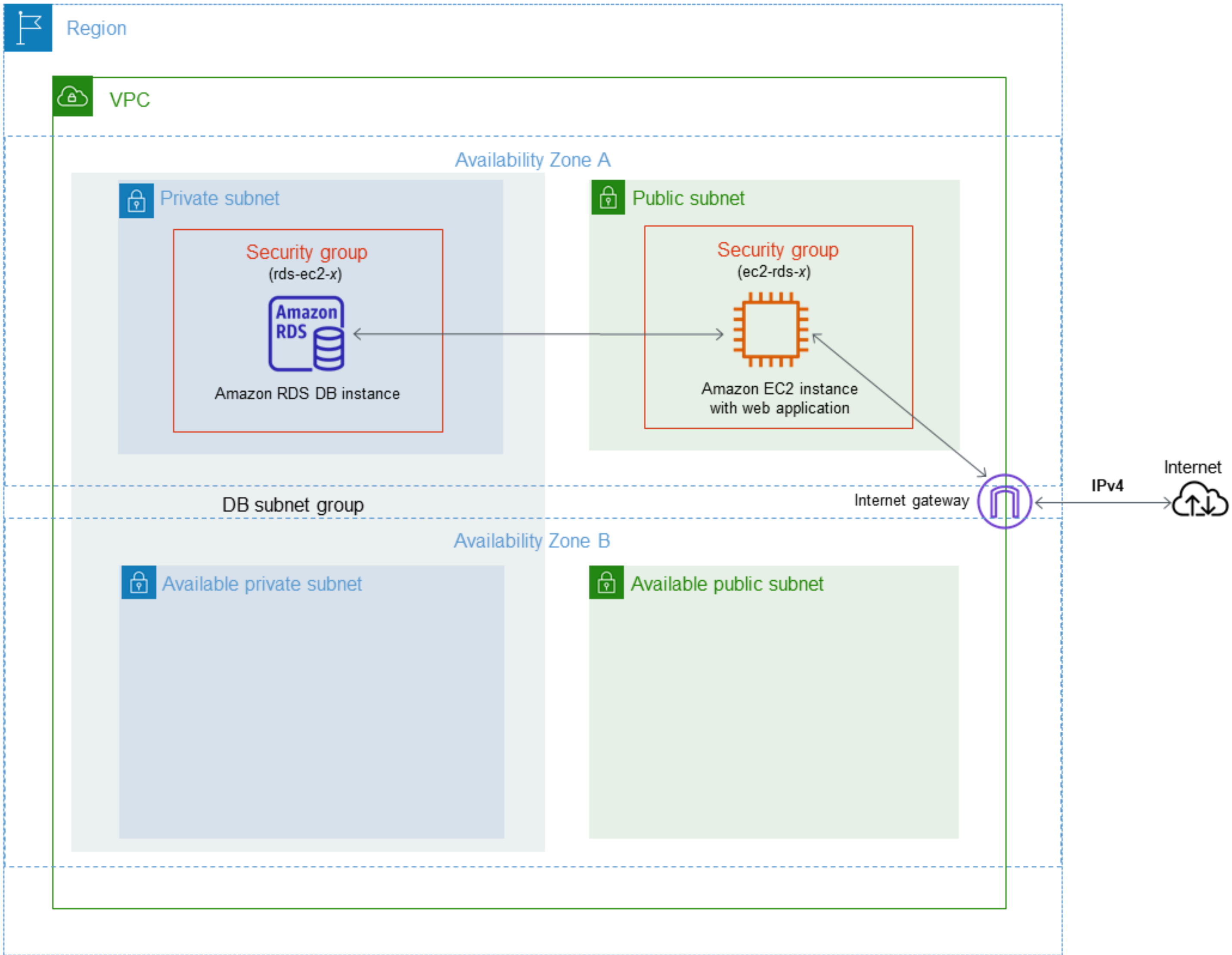
Amazon RDS 支持以下在 VPC 中访问数据库实例的场景：

- 同一 VPC 中的 Amazon EC2 实例
- 不同 VPC 中的 EC2 实例
- 通过 Internet 访问的客户端应用程序
- 私有网络

VPC 中的数据库实例由同一 VPC 中的 Amazon EC2 实例访问

VPC 中的数据库实例的常见用途是与在相同 VPC 的 Amazon EC2 实例中运行的应用程序服务器共享数据。

下图说明了此情形。



管理相同 VPC 中 EC2 实例与数据库实例之间的访问的最简单方法是执行以下操作：

- 创建数据库实例将位于其中的 VPC 安全组。此安全组可用于限制对数据库实例的访问权限。例如，您可以为该安全组创建自定义规则。该规则可能允许使用您创建数据库实例时分配给它的端口以及您用来访问数据库实例的 IP 地址（用于开发或其他目的）进行 TCP 访问。
- 创建您的 EC2 实例（Web 服务器和客户端）将位于其中的 VPC 安全组。如果需要，此安全组可允许使用 VPC 的路由表从 Internet 访问 EC2 实例。例如，您可设置此安全组的规则以允许通过端口 22 对 EC2 实例进行 TCP 访问。
- 为数据库实例的安全组创建自定义规则，这些规则允许从您为 EC2 实例创建的安全组进行连接。这些规则将允许安全组的任何成员访问数据库实例。

在单独的可用区中还有一个额外的公有和私有子网。RDS 数据库子网组需要位于至少两个可用区中的一个子网。额外的子网使将来很容易切换到多可用区数据库实例部署。

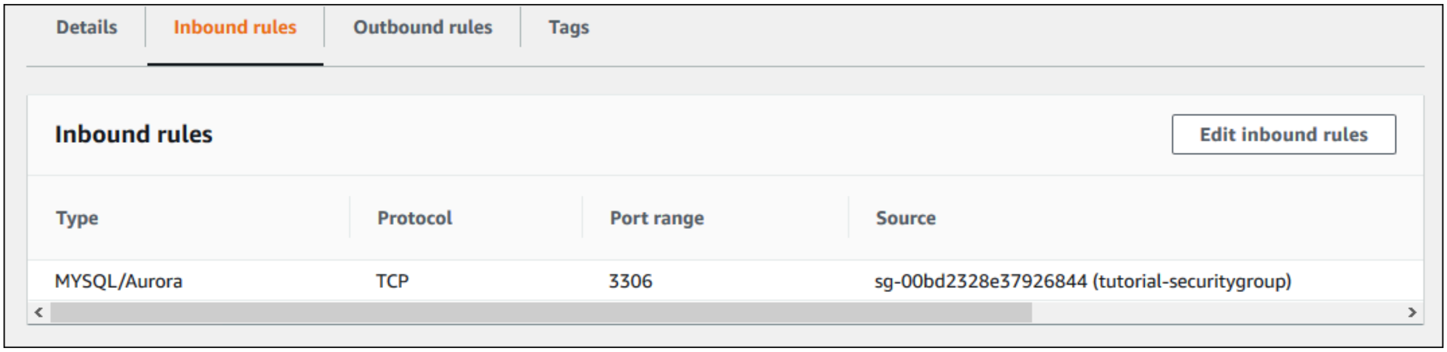
有关说明如何为此方案创建包含公有子网和私有子网的 VPC 的教程，请参阅教程：创建 VPC 以用于数据库实例（仅限 IPv4）。

> **Tip**
>
> 创建数据库实例时，您可以在 Amazon EC2 实例和数据库实例之间自动设置网络连接。有关更多信息，请参阅 配置与 EC2 实例的自动网络连接。

要在 VPC 安全组中创建允许从另一安全组连接的规则，请执行以下操作：

1. 登录到Amazon Web Services Management Console并打开 Amazon VPC 控制台，网址：https://console.aws.amazon.com/vpc。
2. 在导航窗格中，选择 Security groups（安全组）。
3. 选择或创建要允许另一个安全组的成员访问的安全组。在前面的场景中，这是您用于数据库实例的安全组。选择 Inbound rules（入站规则）选项卡，然后选择 Edit inbound rules（编辑入站规则）。
4. 在 Edit inbound rules（编辑入站规则）页面上，选择 Add rule（添加规则）。
5. 对于类型，选择与创建数据库实例时使用的端口相对应的条目，如 MYSQL/Aurora。
6. 在源框中，开始键入安全组的 ID，其中列出了匹配的安全组。选择您希望其成员可以访问此安全组保护的资源的安全组。在前面的方案中，这是您用于 EC2 实例的安全组。
7. 如果需要，可通过在源框中创建类型为所有 TCP 的规则以及安全组，对 TCP 协议重复这些步骤。如果您打算使用 UDP 协议，请在 Source（源）框中创建 Type（类型）为 All UDP（所有 UDP）的规则以及安全组。
8. 选择保存规则。

以下屏幕显示了包含其来源的安全组的入站规则。

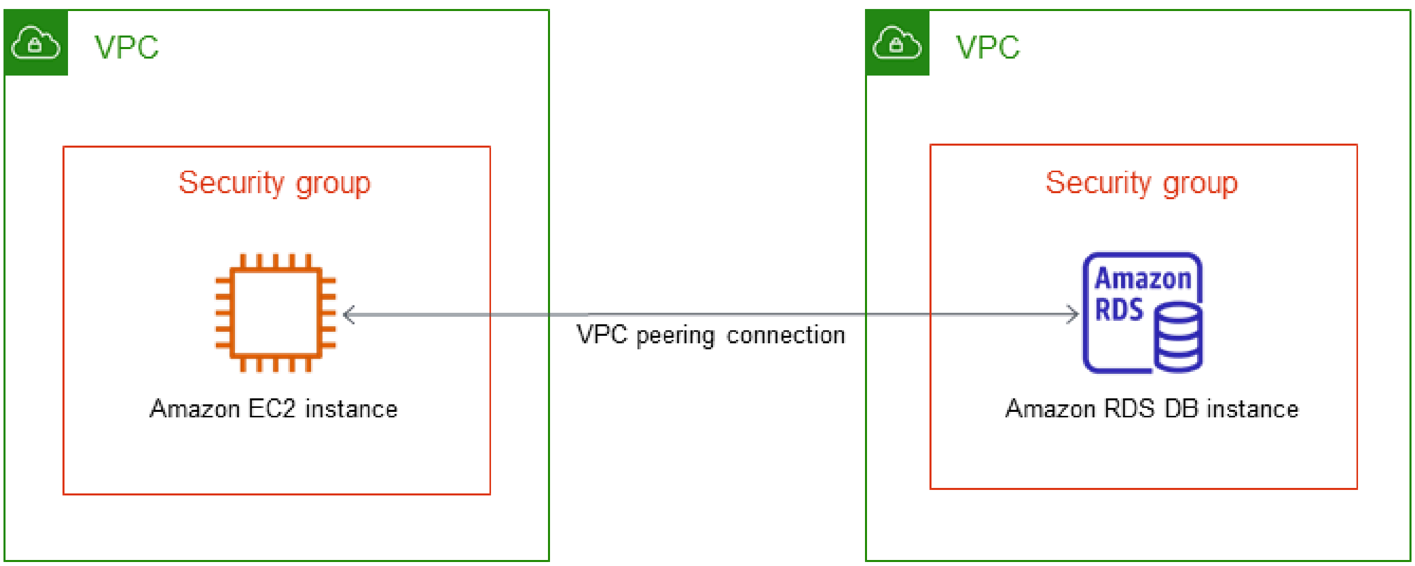


有关从 EC2 实例连接到数据库实例的更多信息，请参阅[连接到 Amazon RDS 数据库实例]。

VPC 中的数据库实例由另一 VPC 中的 EC2 实例访问

当您的数据库实例与您用来访问它的 EC2 实例位于不同的 VPC 中时，可使用 VPC 对等连接来访问数据库实例。

下图说明了此情形。

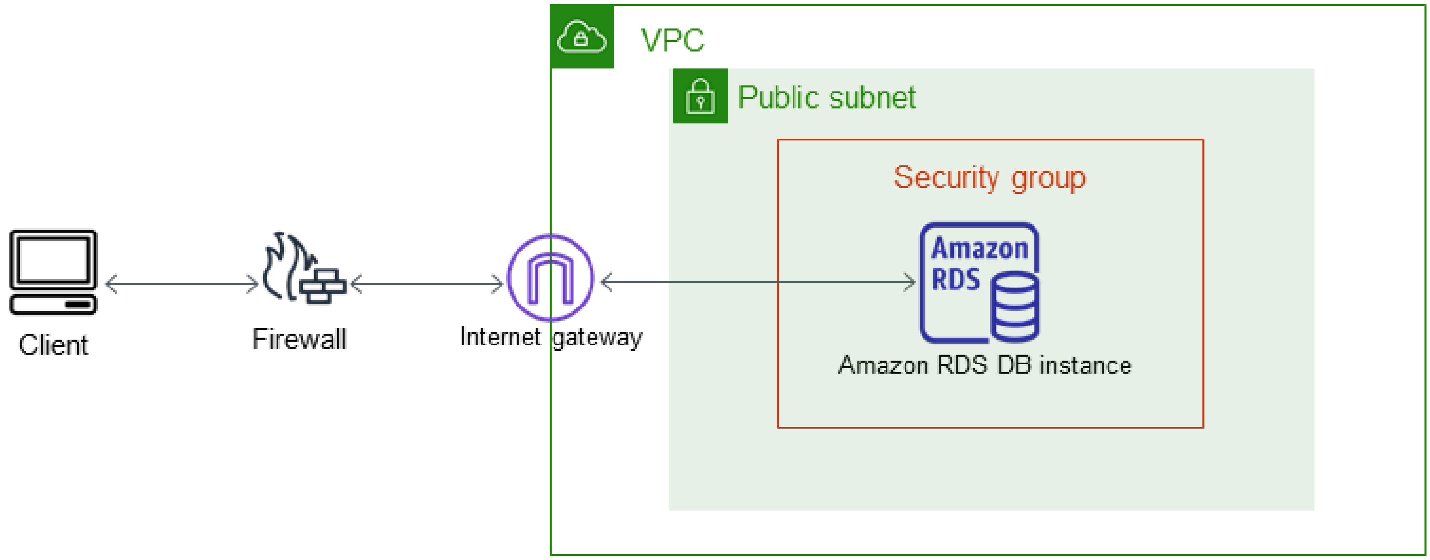


VPC 对等连接是两个 VPC 之间的网络连接，通过此连接，您可以使用私有 IP 地址在这两个 VPC 之间路由流量。这两个 VPC 中的资源可以彼此通信，就像它们在同一网络中一样。您可以在自己的 VPC 之间、自己的 VPC 与另一个 Amazon 账户中的 VPC 或与其他 Amazon Web Services 区域中的 VPC 之间创建 VPC 对等连接。要了解有关 VPC 对等的更多信息，请参阅 Amazon Virtual Private Cloud 用户指南 中的 VPC 对等。

VPC 中的数据库实例由客户端应用程序通过互联网访问

要从客户端应用程序通过互联网访问 VPC 中的数据库实例，您可配置包含单个公有子网的 VPC 以及一个互联网网关，以实现通过互联网通信。

下图说明了此情形。



推荐以下配置：

- 大小为 /16 的 VPC (例如，CIDR：10.0.0.0/16)。此大小提供了 65536 个私有 IP 地址。
- 大小为 /24 的子网 (例如，CIDR：10.0.0.0/24)。此大小提供了 256 个私有 IP 地址。
- 与 VPC 和子网关联的 Amazon RDS 数据库实例。Amazon RDS 将子网内的 IP 地址分配给您的数据库实例。
- 将 VPC 连接到 Internet 和其他 Amazon 产品的互联网网关。
- 与数据库实例关联的安全组。安全组的入站规则允许客户端应用程序访问数据库实例。

有关在 VPC 中创建数据库实例的信息，请参阅[在 VPC 中创建数据库实例]。

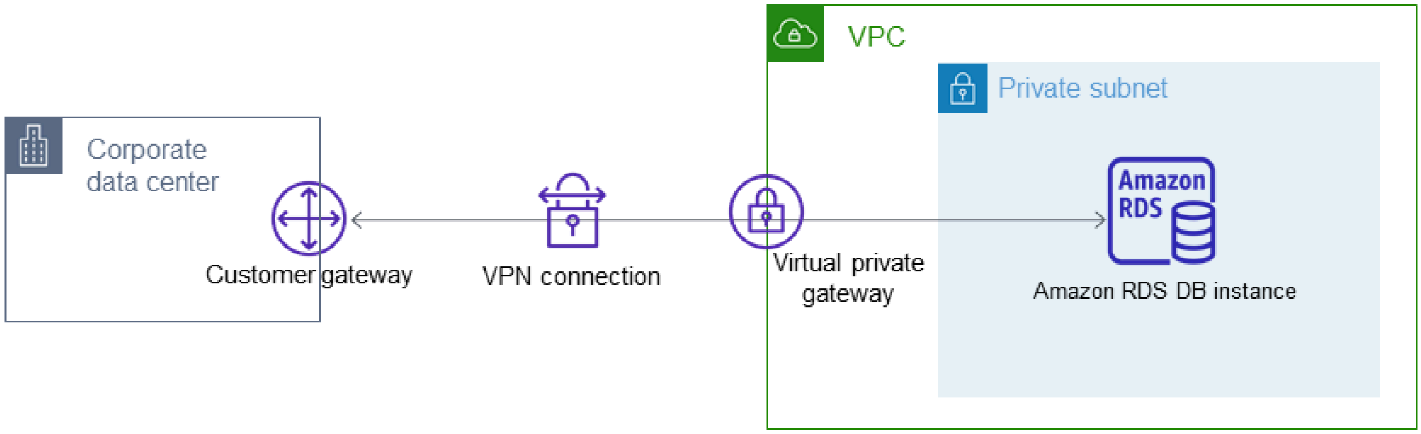
VPC 中由私有网络访问的数据库实例

如果您的数据库实例不可公开访问，可通过以下选项从私有网络进行访问：

- 一个Amazon Site-to-Site VPN 连接。有关更多信息，请参阅[什么是 Amazon Site-to-Site VPN？]
- Amazon Direct Connect 连接。有关更多信息，请参阅[什么是 Amazon Direct Connect？]

- Amazon Client VPN 连接。有关更多信息，请参阅[什么是 Amazon Client VPN？](#)

下图显示了一个具有Amazon Site-to-Site VPN 连接的场景。

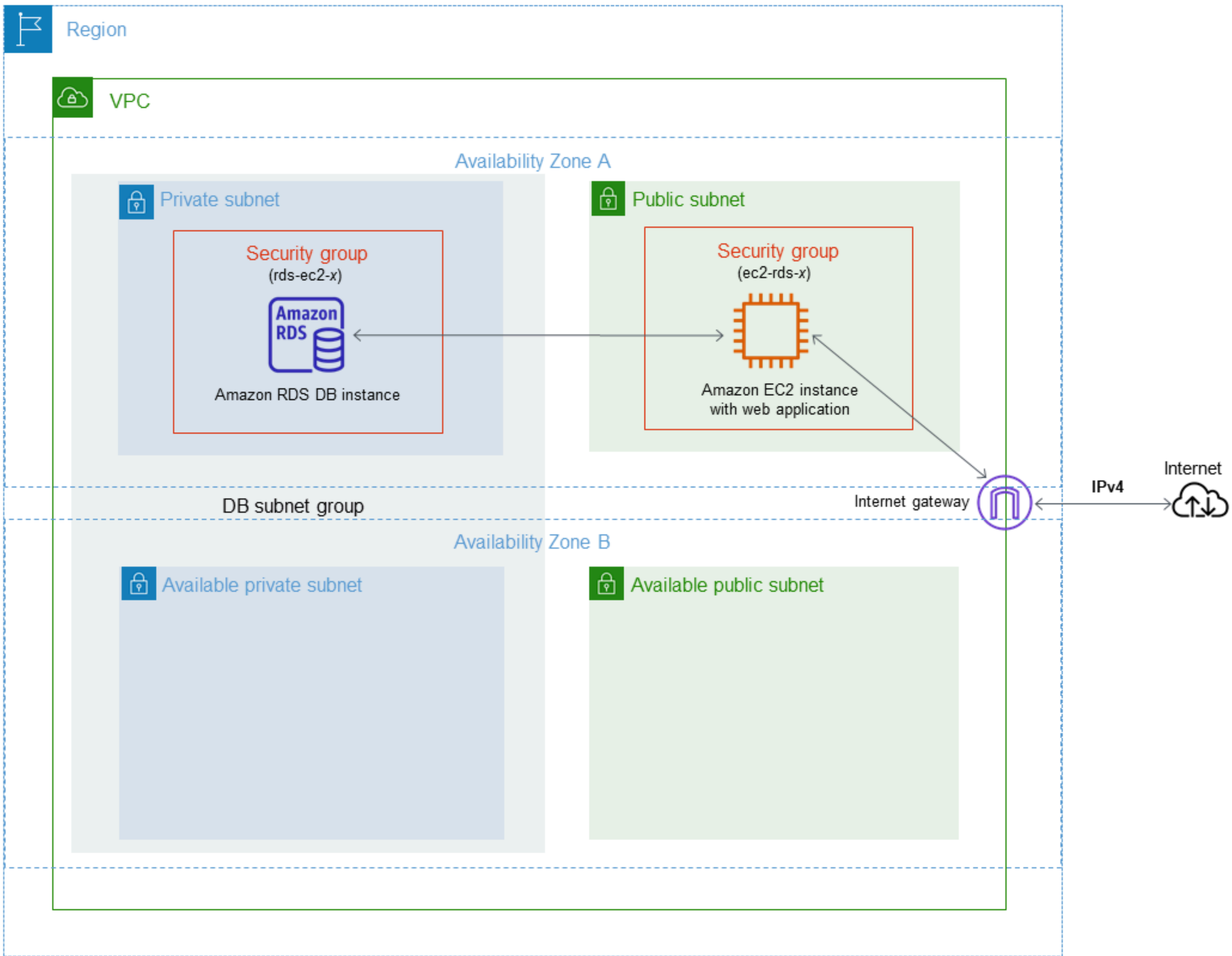


有关更多信息，请参阅 [互联网络流量隐私](#)。

教程：创建 VPC 以用于数据库实例（仅限 IPv4）

一种常见的场景包括虚拟私有云（VPC）中基于 Amazon VPC 服务的数据库实例。此 VPC 与在同一 VPC 中运行的 Web 服务器共享数据。在本教程中，针对此场景创建 VPC。

下图说明了此情形。有关其他方案的信息，请参阅在 VPC 中访问数据库实例的场景。



数据库实例只需对 Web 服务器可用，而无需对公共互联网可用。因此，请创建包含公有子网和私有子网的 VPC。Web 服务器托管在公有子网中，以便它可访问公共互联网。数据库实例托管于私有子网中。Web 服务器可以连接到数据库实例，因为它托管在同一 VPC 内。但是，数据库实例不可用于公共互联网，从而提高了安全性。

本教程在单独的可用区中配置额外的公有和私有子网。本教程未使用这些子网。RDS 数据库子网组需要位于至少两个可用区中的一个子网。额外的子网使将来更容易切换到多可用区数据库实例部署。

本教程介绍为 Amazon RDS 数据库实例配置 VPC。有关向您说明如何为此 VPC 方案创建 Web 服务器的教程，请参阅教程：创建 Web 服务器和 Amazon RDS 数据库实例。有关 Amazon VPC 的更多信息，请参阅 Amazon VPC 入门指南和 Amazon VPC 用户指南。

> **Tip**
>
> 创建数据库实例时，您可以在 Amazon EC2 实例和数据库实例之间自动设置网络连接。网络配置类似于本教程中描述的配置。有关更多信息，请参阅配置与 EC2 实例的自动网络连接。

创建包含公有子网和私有子网的 VPC

使用以下步骤创建包含公有和私有子网的 VPC。

创建 VPC 和子网

1. 通过 https://console.aws.amazon.com/vpc/ 打开 Amazon VPC 控制台。
2. 在 Amazon Web Services Management Console 的右上角，选择要在其中创建 VPC 的区域。此示例使用 美国西部（俄勒冈） 区域。
3. 在左上角，选择 VPC Dashboard（VPC 控制面板）。要开始创建 VPC，请选择 Create VPC（创建 VPC）。
4. 对于 VPC Settings（VPC 设置）下的 Resources to create（要创建的资源），选择 VPC and more（VPC 及更多）。
5. 对于 VPC settings（VPC 设置），请设置以下值：

   - Name tag auto-generation（名称标签自动生成）– **`tutorial`**
   - IPv4 CIDR block（IPv4 CIDR 块）– **`10.0.0.0/16`**
   - IPv6 CIDR block（IPv6 CIDR 块）– No IPv6 CIDR block（无 IPv6 CIDR 块）
   - Tenancy（租赁）– Default（原定设置）
   - Number of Availability Zones (AZs) [可用区（AZ）数量] – 2
   - Customize AZs（自定义可用区）– 保留原定设置值。
   - Number of public subnet（公有子网的数量）– 2
   - Number of private subnets（私有子网的数量）– 2
   - Customize subnets CIDR blocks（自定义子网 CIDR 块）– 保留原定设置值。
   - NAT gateways ($) [NAT 网关（$）]– None（无）

- VPC endpoints（VPC 端点）– None（无）
- DNS options（DNS 选项）– 保留原定设置值。

> **Note**
>
> Amazon RDS 至少需要位于两个不同可用区中的两个子网，才能支持多可用区数据库实例部署。本教程创建了单可用区部署，但这项要求使得将来可以更轻松地转换为多可用区数据库实例部署。

6. 选择创建 VPC。

为公共 Web 服务器创建 VPC 安全组

接下来创建安全组以便公共访问。要连接到 VPC 中的公有 EC2 实例，请将入站规则添加到 VPC 安全组。这些规则允许流量从互联网进行连接。

创建 VPC 安全组

1. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
2. 依次选择 VPC 控制面板)、安全组和创建安全组。
3. 在创建安全组页面上，设置以下值：
   - 安全组名称：**`tutorial-securitygroup`**
   - 说明：**`Tutorial Security Group`**
   - VPC：选择您之前创建的 VPC，例如：vpc-***`identifier`*** (tutorial-vpc)
4. 将入站规则添加到安全组。

   a. 确定要用来通过 Secure Shell（SSH）连接到 VPC 中的 EC2 实例的 IP 地址。要确定您的公有 IP 地址，请在新的浏览器窗口或标签页中，使用 https://checkip.amazonaws.com 上的服务。IP 地址的一个示例为 `203.0.113.25/32`。

      在许多情况下，您可能通过互联网服务提供商（ISP）进行连接，或者在不使用静态 IP 地址的情况下从防火墙之后进行连接。如果是这样，请找出客户端计算机使用的 IP 地址范围。

> **Warning**
>
> 如果您使用 `0.0.0.0/0` 进行 SSH 访问，则所有 IP 地址可能能够使用 SSH 访问您的公有实例。在测试环境下短时间内，此方法尚可接受，但它对于生产环境并不安全。在生产环境中，将仅向特定 IP 地址或地址范围授权使用 SSH 访问您的实例。

b. 在入站规则部分中，选择添加规则。

c. 为新入站规则设置以下值，以允许 SSH 访问 Amazon EC2 实例。如果这样做，您就可以连接到 Amazon EC2 实例，以便安装 Web 服务器和其他实用程序。您还可以连接到 EC2 实例，以便上载 Web 服务器的内容。

   - 类型： **`SSH`**
   - 源：步骤 a 中的 IP 地址或范围，例如：**`203.0.113.25/32`**。

d. 选择 添加规则。

e. 为新入站规则设置以下值以允许针对 Web 服务器的 HTTP 访问：

   - 类型： **`HTTP`**
   - 源：**`0.0.0.0/0`**

5. 请选择 Create security group（创建安全组）以创建安全组。

   请记下安全组 ID，因为本教程的后面将需要它。

为私有数据库实例创建 VPC 安全组

要保持您的数据库实例为私有，请创建第二个安全组进行私有访问。要连接到 VPC 中的私有数据库实例，请将入站规则添加到 VPC 安全组，以仅支持来自 Web 服务器的流量。

创建 VPC 安全组

1. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
2. 依次选择 VPC 控制面板)、安全组和创建安全组。
3. 在创建安全组页面上，设置以下值：

   - 安全组名称：**`tutorial-db-securitygroup`**
   - 说明：**`Tutorial DB Instance Security Group`**
   - VPC：选择您之前创建的 VPC，例如：vpc-***`identifier`*** (tutorial-vpc)

4. 将入站规则添加到安全组。

   a. 在入站规则部分中，选择添加规则。

   b. 为新入站规则设置以下值，以允许 Amazon EC2 实例中端口 3306 上的 MySQL 流量。如果这样做，您就可以从 Web 服务器连接到数据库实例。这样，您就可以从 Web 应用程序将数据存储和检索到数据库。

      - 类型：**MySQL/Aurora**
      - Source（源）：您在本教程的前面部分创建的 tutorial-securitygroup 安全组的标识符，例如 sg-9edd5cfb。

5. 请选择 Create security group（创建安全组）以创建安全组。

创建数据库子网组

数据库子网组 是您在 VPC 中创建并随后指定给数据库实例的子网集合。通过数据库子网组，您可能能够在创建数据库实例时指定特定的 VPC。

创建数据库子网组

1. 在 VPC 中识别数据库的私有子网。

   a. 通过 https://console.aws.amazon.com/vpc/ 打开 Amazon VPC 控制台。

   b. 选择 VPC Dashboard（VPC 控制面板），然后选择 Subnets（子网）。

   c. 记下名为 tutorial-subnet-private1-us-west-2a 和 tutorial-subnet-private2-us-west-2b 的子网的子网 ID。

      创建数据库子网组时需要子网 ID。

2. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。

   确保您连接到 Amazon RDS 控制台，而不是 Amazon VPC 控制台。

3. 在导航窗格中，选择子网组。

4. 选择 Create DB subnet group（创建数据库子网组）。

5. 在创建数据库子网组页面的子网组详细信息中设置以下值：

   - 名称：**tutorial-db-subnet-group**
   - 说明：**Tutorial DB Subnet Group**
   - VPC：tutorial-vpc (vpc-***identifier***)

6. 在添加子网部分中，选择可用区和子网。

   对于本教程，请为 Availability Zones（可用区）选择 us-west-2a 和 us-west-2b。对于 Subnets（子网），选择您在上一步中确定的私有子网。

7. 选择创建。

   您的新数据库子网组显示在 RDS 控制台的数据库子网组列表中。可选择该数据库子网组，以在窗口底部的详细信息窗格中查看详细信息。这些详细信息包括与该组关联的所有子网。

> **Note**
>
> 如果您已创建此 VPC 以完成 教程：创建 Web 服务器和 Amazon RDS 数据库实例，请按照 创建 Amazon RDS 数据库实例 中的说明创建数据库实例。

删除 VPC

为本教程创建 VPC 和其他资源后，如果不再需要 VPC 和其他资源，可以将其删除。

> **Note**
>
> 如果您在为本教程创建的 VPC 中添加了资源，则可能需要先删除这些资源，然后才能删除 VPC。例如，这些资源可能包括 Amazon EC2 实例或 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon VPC 用户指南中的您的 VPC 的安全性。

删除 VPC 和相关资源

1. 删除数据库子网组。

   a. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
   b. 在导航窗格中，选择子网组。
   c. 选择要删除的数据库子网组，例如 tutorial-db-subnet-group。
   d. 选择 Delete (删除)，然后在确认窗口中选择 Delete (删除)。

2. 记下 VPC ID。

   a. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
   b. 选择 VPC 控制面板，然后选择 VPC。

c. 在列表中，确定您创建的 VPC，例如 tutorial-vpc。

d. 记下所创建 VPC 的 VPC ID。在后面的步骤中，您需要此 VPC ID。

3. 删除安全组。

   a. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。

   b. 选择 VPC 控制面板，然后选择安全组。

   c. 选择 Amazon RDS 数据库实例的安全组，例如 tutorial-db-securitygroup。

   d. 对于 Actions（操作），请选择 Delete security groups（删除安全组），然后在确认页面上选择 Delete（删除）。

   e. 在安全组页面上，选择 Amazon EC2 实例的安全组，例如 tutorial-securitygroup。

   f. 对于 Actions（操作），请选择 Delete security groups（删除安全组），然后在确认页面上选择 Delete（删除）。

4. 删除 VPC

   a. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。

   b. 选择 VPC 控制面板，然后选择 VPC。

   c. 选择要删除的 VPC，例如 tutorial-vpc。

   d. 对于 Actions (操作)，请选择 Delete VPC (删除 VPC)。

      确认页面显示与 VPC 关联的其他资源，这些资源也将被删除，包括与其关联的子网。

   e. 在确认页面上，输入 **`delete`** 并选择 Delete (删除)。

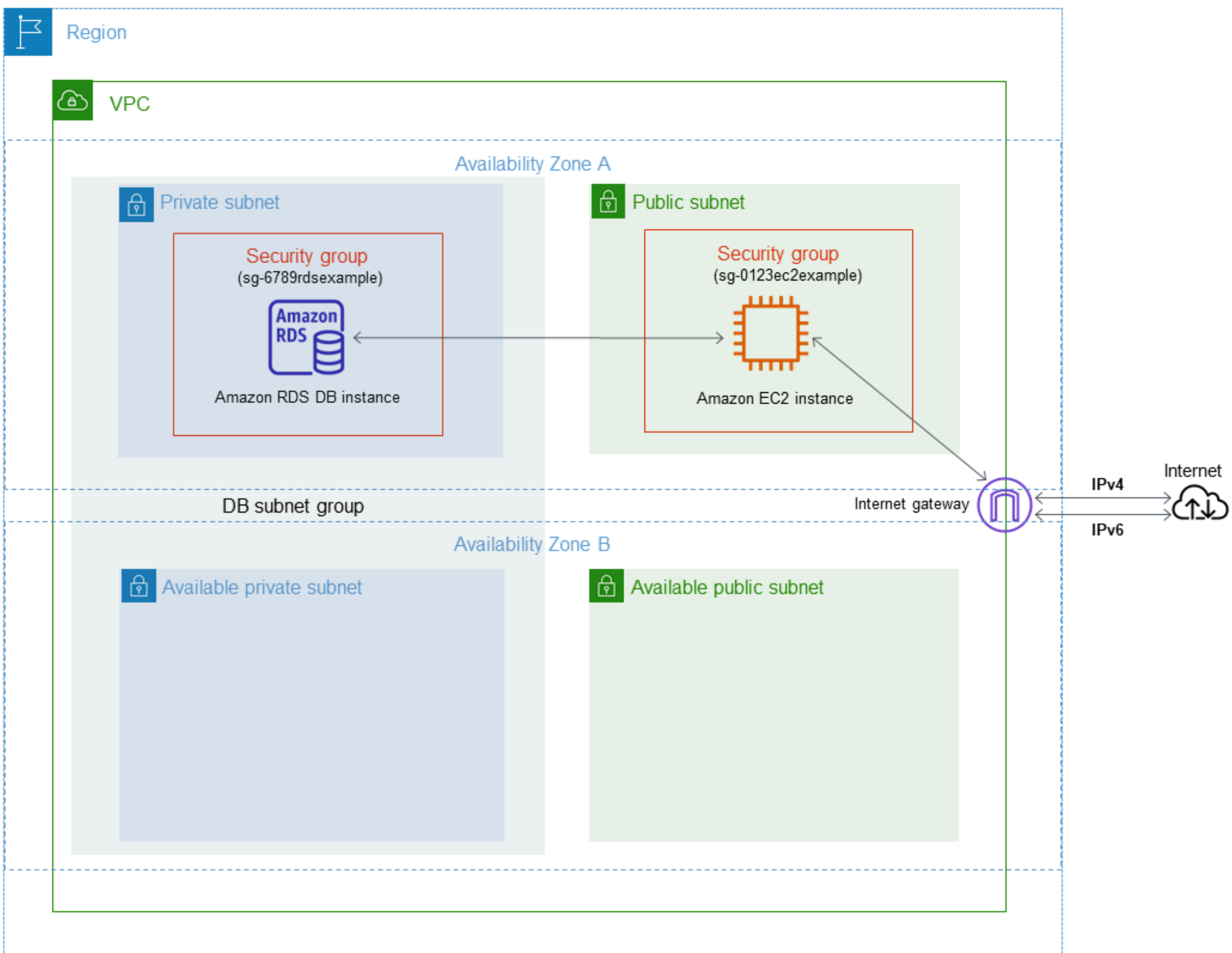
教程：创建 VPC 以用于数据库实例（双堆栈模式）

一种常见的场景包括虚拟私有云（VPC）中基于 Amazon VPC 服务的数据库实例。此 VPC 与在同一 VPC 中运行的公有 Amazon EC2 实例共享数据。

在本教程中，您将为此场景创建 VPC，该 VPC 与在双堆栈模式下运行的数据库一起使用。双堆栈模式，用于启用基于 IPv6 寻址协议的连接。有关 IP 地址的更多信息，请参阅 Amazon RDS IP 寻址。

大多数区域支持双堆栈网络实例。有关更多信息，请参阅区域和版本可用性。要查看双堆栈模式的限制，请参阅 双堆栈网络数据库实例的限制。

下图说明了此情形。



有关其他方案的信息，请参阅在 VPC 中访问数据库实例的场景。

数据库实例只需对 Amazon EC2 实例可用，而无需对公共互联网可用。因此，请创建包含公有子网和私有子网的 VPC。Amazon EC2 实例托管在公有子网中，以便它可访问公共 Internet。数据库实例托管于私有子网中。Amazon EC2 实例可以连接到数据库实例，因为它托管在同一 VPC 内。但是，数据库实例不可用于公共互联网，从而提高了安全性。

本教程在单独的可用区中配置额外的公有和私有子网。本教程未使用这些子网。RDS 数据库子网组需要位于至少两个可用区中的一个子网。额外的子网使将来很容易切换到多可用区数据库实例部署。

要创建使用双堆栈模式的数据库实例，请为 Network type（网络类型）设置指定 Dual-stack mode（双堆栈模式）。您也可以使用相同的设置修改数据库实例。有关更多信息，请参阅创建 Amazon RDS 数据库实例 和修改 Amazon RDS 数据库实例。

本教程介绍为 Amazon RDS 数据库实例配置 VPC。有关 Amazon VPC 的更多信息，请参阅 Amazon VPC 用户指南。

创建包含公有子网和私有子网的 VPC

使用以下步骤创建包含公有和私有子网的 VPC。

创建 VPC 和子网

1. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
2. 在 Amazon Web Services Management Console 的右上角，选择要在其中创建 VPC 的区域。此示例使用美国东部（俄亥俄州）区域。
3. 在左上角，选择 VPC Dashboard（VPC 控制面板）。要开始创建 VPC，请选择 Create VPC（创建 VPC）。
4. 对于 VPC Settings（VPC 设置）下的 Resources to create（要创建的资源），选择 VPC and more（VPC 及更多）。
5. 对于剩下的 VPC settings（VPC 设置），请设置这些值：

   - Name tag auto-generation（名称标签自动生成）– **tutorial-dual-stack**
   - IPv4 CIDR block（IPv4 CIDR 块）– **10.0.0.0/16**
   - IPv6 CIDR block（IPv6 CIDR 块）– Amazon-provided IPv6 CIDR block（Amazon 提供的 IPv6 CIDR 块）
   - Tenancy（租赁）– Default（原定设置）
   - Number of Availability Zones (AZs) [可用区（AZ）数量] – 2
   - Customize AZs（自定义可用区）– 保留原定设置值。
   - Number of public subnet（公有子网的数量）– 2

- Number of private subnets（私有子网的数量）– 2
- Customize subnets CIDR blocks（自定义子网 CIDR 块）– 保留原定设置值。
- NAT gateways ($) [NAT 网关（$）]– None（无）
- Egress only internet gateway（仅限出口的互联网网关）– No（否）
- VPC endpoints（VPC 端点）– None（无）
- DNS options（DNS 选项）– 保留原定设置值。

> **Note**
>
> Amazon RDS 至少需要位于两个不同可用区中的两个子网，才能支持多可用区数据库实例部署。本教程创建了单可用区部署，但这项要求使得将来可以轻松转换为多可用区数据库实例部署。

6. 选择创建 VPC。

为公有 Amazon EC2 实例创建 VPC 安全组

接下来创建安全组以便公共访问。要连接到 VPC 中的公有 EC2 实例，请将入站规则添加到 VPC 安全组，以允许流量从互联网连接。

创建 VPC 安全组

1. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
2. 依次选择 VPC 控制面板)、安全组和创建安全组。
3. 在创建安全组页面上，设置以下值：
   - 安全组名称：**`tutorial-dual-stack-securitygroup`**
   - 说明：**`Tutorial Dual-Stack Security Group`**
   - VPC：选择您之前创建的 VPC，例如：vpc-***`identifier`*** (tutorial-dual-stack-vpc)
4. 将入站规则添加到安全组。
   a. 确定要用来通过 Secure Shell（SSH）连接到 VPC 中的 EC2 实例的 IP 地址。

      互联网协议版本 4（IPv4）地址的示例为 `203.0.113.25/32`。互联网协议版本 6（IPv6）地址范围的示例为 `2001:db8:1234:1a00::/64`。

在许多情况下，您可能通过互联网服务提供商（ISP）进行连接，或者在不使用静态 IP 地址的情况下从防火墙之后进行连接。如果是这样，请找出客户端计算机使用的 IP 地址范围。

> **Warning**
>
> 如果对 IPv4 使用 `0.0.0.0/0` 或对 IPv6 使用 `::0`，则使所有 IP 地址可能能够使用 SSH 访问您的公有实例。在测试环境下短时间内，此方法尚可接受，但它对于生产环境并不安全。在生产环境中，请仅授权特定 IP 地址或地址范围访问您的实例。

b. 在入站规则部分中，选择添加规则。

c. 为新入站规则设置以下值，以允许安全外壳 (SSH) 访问 Amazon EC2 实例。如果这样做，则可以连接到 EC2 实例来安装 SQL 客户端和其他应用程序。指定一个 IP 地址，以便您可以访问 EC2 实例：

- 类型：**`SSH`**
- Source（源）：步骤 a 中的 IP 地址或范围。IPv4 IP 地址的一个示例为 **`203.0.113.25/32`**。IPv6 IP 地址的一个示例为 **`2001:DB8::/32`**。

5. 请选择 Create security group（创建安全组）以创建安全组。

   请记下安全组 ID，因为本教程的后面将需要它。

为私有数据库实例创建 VPC 安全组

要保持您的数据库实例为私有，请创建第二个安全组进行私有访问。要连接到 VPC 中的私有数据库实例，请将入站规则添加到 VPC 安全组。它们仅允许来自 Amazon EC2 实例的流量。

创建 VPC 安全组

1. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
2. 依次选择 VPC 控制面板)、安全组和创建安全组。
3. 在创建安全组页面上，设置以下值：
   - 安全组名称：**`tutorial-dual-stack-db-securitygroup`**
   - 说明：**`Tutorial Dual-Stack DB Instance Security Group`**
   - VPC：选择您之前创建的 VPC，例如：vpc-***`identifier`*** (tutorial-dual-stack-vpc)
4. 将入站规则添加到安全组：

a. 在入站规则部分中，选择添加规则。

b. 为新入站规则设置以下值，以允许 Amazon EC2 实例中端口 3306 上的 MySQL 流量。如果这样做，您就可以从 EC2 实例连接到您的数据库实例。这样做意味着您可以将数据从 EC2 实例发送到数据库。

- Type（类型）：MySQL/Aurora
- Source（源）：您在本教程的前面部分创建的 tutorial-dual-stack-securitygroup 安全组的标识符，例如 sg-9edd5cfb。

5. 要创建安全组，请选择创建安全组。

创建数据库子网组

数据库子网组 是您在 VPC 中创建并随后指定给数据库实例的子网集合。使用数据库子网组，您可以在创建数据库实例时指定特定的 VPC。要创建与 `DUAL` 兼容的数据库子网组，所有子网都必须与 `DUAL` 兼容。要与 `DUAL` 兼容，子网必须具有与之关联的 IPv6 CIDR。

创建数据库子网组

1. 在 VPC 中识别数据库的私有子网。

   a. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。

   b. 选择 VPC Dashboard（VPC 控制面板），然后选择 Subnets（子网）。

   c. 记下名为 tutorial-dual-stack-subnet-private1-us-west-2a 和 tutorial-dual-stack-subnet-private2-us-west-2b 的子网的子网 ID。

   创建数据库子网组时需要子网 ID。

2. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。

   确保您连接到 Amazon RDS 控制台，而不是 Amazon VPC 控制台。

3. 在导航窗格中，选择子网组。

4. 选择 Create DB subnet group（创建数据库子网组）。

5. 在创建数据库子网组页面的子网组详细信息中设置以下值：

- 名称：**`tutorial-dual-stack-db-subnet-group`**
- 说明：**`Tutorial Dual-Stack DB Subnet Group`**
- VPC：tutorial-dual-stack-vpc (vpc-***`identifier`***)

6. 在 Add subnets（添加子网）部分中，选择 Availability Zones（可用区）和 Subnets（子网）选项的值。

   对于本教程，请为 Availability Zones（可用区）选择 us-east-2a 和 us-east-2b。对于 Subnets（子网），选择您在上一步中确定的私有子网。

7. 选择创建。

您的新数据库子网组显示在 RDS 控制台的数据库子网组列表中。您可以选择数据库子网组以查看其详细信息。其中包括支持的寻址协议、与该组关联的所有子网以及数据库子网组支持的网络类型。

在双堆栈模式下创建 Amazon EC2 实例

要创建 Amazon EC2 实例，请按照《Amazon EC2 用户指南》中的使用新启动实例向导启动实例中的说明操作。

在 Configure Instance Details（配置实例详细信息）页面上，设置以下值并将其他值保留为其原定设置值：

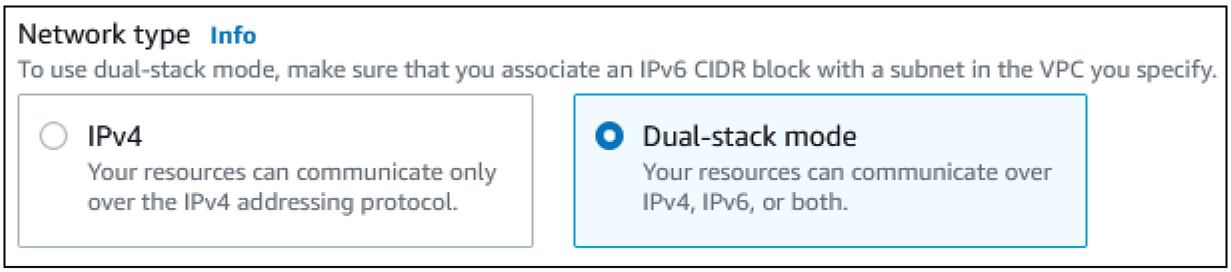
- 网络 – 选择同时具有公有子网和私有子网的现有 VPC，如在创建包含公有子网和私有子网的 VPC中创建的 tutorial-dual-stack-vpc（vpc-*identifier*）。
- Subnet（子网）– 选择一个现有的公有子网，如在为公有 Amazon EC2 实例创建 VPC 安全组中创建的 subnet-***identifier*** | tutorial-dual-stack-subnet-public1-us-east-2a | us-east-2a。
- Auto-assign Public IP（自动分配公有 IP）– 选择 Enable（启用）。
- Auto-assign IPv6 IP（自动分配 IPv6 IP）– 选择 Enable（启用）。
- Firewall (security groups) [防火墙（安全组）] – 选择 Select an existing security group（选择现有安全组）。
- Common security groups（常用安全组）– 选择一个现有的安全组，例如在为公有 Amazon EC2 实例创建 VPC 安全组中创建的 `tutorial-securitygroup`。确保您选择的安全组包括 Secure Shell (SSH) 和 HTTP 访问的入站规则。

在双堆栈模式下创建数据库实例

在此步骤中，您将创建在双堆栈模式下运行的数据库实例。

创建数据库实例

1. 登录 Amazon Web Services Management Console 并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
2. 在控制台的右上角，选择要在其中创建数据库实例的 Amazon Web Services 区域。此示例使用美国东部（俄亥俄州）区域。
3. 在导航窗格中，选择 Databases（数据库）。
4. 选择创建数据库。
5. 在 Create database（创建数据库）页面上，确保选择了 Standard Create（标准创建）选项，然后选择 MySQL 数据库引擎类型。
6. 在 Connectivity（连接）部分中，设置以下值：
   - Network type（网络类型）– 选择 Dual-stack mode（双堆栈模式）。

   - Virtual Private Cloud (VPC) [虚拟私有云（VPC）]– 选择具有公有子网和私有子网的现有 VPC，如在创建包含公有子网和私有子网的 VPC中创建的 tutorial-dual-stack-vpc (vpc-*identifier*)。

     VPC 的子网必须位于不同的可用区中。
   - DB subnet group（数据库子网组）– VPC 的数据库子网组，如在创建数据库子网组中创建的 tutorial-dual-stack-db-subnet-group。
   - Public access（公有访问权限）– 选择 No（否）。
   - VPC security group (firewall) [VPC 安全组（防火墙）] – 选择 Choose existing（选择现有）。
   - Existing VPC security groups（现有 VPC 安全组）– 选择为私有访问配置的现有 VPC 安全组，如 为私有数据库实例创建 VPC 安全组 中创建的 tutorial-dual-stack-db-securitygroup。

     通过选择与其他每个安全组关联的 X 来删除该安全组，如默认安全组。
   - Availability Zone（可用区）– 选择 us-west-2a。

     为避免跨可用区的流量，请确保数据库实例和 EC2 实例位于同一个可用区内。
7. 对于其余部分，请指定数据库实例设置。有关每项设置的信息，请参阅 数据库实例的设置。

连接到 Amazon EC2 实例和数据库实例

在双堆栈模式下创建 Amazon EC2 实例和数据库实例后，您可以使用 IPv6 协议连接到每个实例。要使用 IPv6 协议连接到 Amazon EC2 实例，请按照《Amazon EC2 用户指南》中的连接到 Linux 实例中的说明操作。

要从 Amazon EC2 实例连接到 RDS for MySQL 数据库实例，请按照连接到 MySQL 数据库实例中的说明进行操作。

删除 VPC

为本教程创建 VPC 和其他资源后，如果不再需要 VPC 和其他资源，可以将其删除。

如果您在为本教程创建的 VPC 中添加了资源，则可能需要先删除这些资源，然后才能删除 VPC。资源示例包括 Amazon EC2 实例或数据库实例。有关更多信息，请参阅 Amazon VPC 用户指南中的您的 VPC 的安全性。

删除 VPC 和相关资源

1. 删除数据库子网组：
   a. 通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
   b. 在导航窗格中，选择子网组。
   c. 选择要删除的数据库子网组，例如 tutorial-db-subnet-group。
   d. 选择 Delete (删除)，然后在确认窗口中选择 Delete (删除)。
2. 记下 VPC ID：
   a. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
   b. 选择 VPC 控制面板，然后选择 VPC。
   c. 在列表中，标识您创建的 VPC，例如 tutorial-dual-stack-vpc。
   d. 记下所创建 VPC 的 VPC ID 值。在后续步骤中，您将需要此 VPC ID。
3. 删除安全组：
   a. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
   b. 选择 VPC 控制面板，然后选择安全组。
   c. 选择 Amazon RDS 数据库实例的安全组，例如 tutorial-dual-stack-db-securitygroup。
   d. 对于 Actions（操作），请选择 Delete security groups（删除安全组），然后在确认页面上选择 Delete（删除）。

e. 在 Security Groups（安全组）页面上，选择 Amazon EC2 实例的安全组，例如 tutorial-dual-stack-securitygroup。
f. 对于 Actions（操作），请选择 Delete security groups（删除安全组），然后在确认页面上选择 Delete（删除）。

4. 删除 NAT 网关：

   a. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
   b. 选择 VPC 控制面板，然后选择 NAT 网关。
   c. 选择您创建的 VPC 的 NAT 网关。使用 VPC ID 标识正确的 NAT 网关。
   d. 对于 Actions（操作），请选择 Delete NAT gateway（删除 NAT 网关）。
   e. 在确认页面上，输入 **delete** 并选择删除。

5. 删除 VPC：

   a. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
   b. 选择 VPC 控制面板，然后选择 VPC。
   c. 选择要删除的 VPC，例如 tutorial-dual-stack-vpc。
   d. 对于 Actions (操作)，请选择 Delete VPC (删除 VPC)。

      确认页面显示与 VPC 关联的其他资源，这些资源也将被删除，包括与其关联的子网。
   e. 在确认页面上，输入 **delete** 并选择 Delete (删除)。

6. 释放弹性 IP 地址：

   a. 通过以下网址打开 Amazon EC2 控制台：https://console.aws.amazon.com/ec2/。
   b. 选择 EC2 控制面板，然后选择弹性 IP。
   c. 选择要释放的弹性 IP 地址。
   d. 对于 Actions（操作），请选择 Release Elastic IP addresses（释放弹性 IP 地址）。
   e. 在确认页面上，请选择释放。

将不在 VPC 中的数据库实例移动到 VPC 中

EC2-Classic 平台上一些旧的数据库实例不位于 VPC 中。如果数据库实例未在 VPC 中，可以使用 Amazon Web Services Management Console轻松将数据库实例移至 VPC 中。必须先创建 VPC，然后才能将未在 VPC 中的数据库实例移至 VPC 中。

EC2-Classic 已于 2022 年 8 月 15 日停用。如果您尚未从 EC2-Classic 迁移到 VPC，建议您尽快迁移。有关更多信息，请参阅《Amazon EC2 用户指南》中的从 EC2-Classic 迁移到 VPC 和博客 EC2-Classic Networking 即将停用 – 以下是准备方法。

> **Important**
>
> 如果您是新 Amazon RDS 客户，您之前从未创建数据库实例或您正在以前未使用的 Amazon 区域中创建数据库实例，则几乎在所有情况下，您都将位于 EC2-VPC 平台上并且具有默认 VPC。有关使用位于 VPC 中的数据库实例的更多信息，请参阅 在 VPC 中使用数据库实例。

执行以下步骤可为数据库实例创建 VPC。

- 步骤 1：创建 VPC
- 步骤 2：创建数据库子网组
- 步骤 3：创建 VPC 安全组

创建 VPC 后，执行以下步骤可将数据库实例移至 VPC 中。

- 更新数据库实例的 VPC

我们强烈建议您在就要迁移之前创建数据库实例的备份。这样做可确保在迁移失败时可以恢复数据。有关更多信息，请参阅“备份、还原和导出数据”。

以下是将数据库实例移至 VPC 中时的一些限制。

- 上一代数据库实例类 – 在 VPC 平台上可能不支持上一代数据库实例类。将数据库实例移到 VPC 时，请选择 db.m3 或 db.r3 数据库实例类。将数据库实例移到 VPC 后，您可以扩展数据库实例以使用更高版本的数据库实例类。有关 VPC 支持的实例类的完整列表，请参阅 Amazon RDS 实例类型。

- 多可用区 – 当前不支持将未在 VPC 中的多可用区数据库实例移至 VPC 中。要将数据库实例移到 VPC，请首先修改数据库实例，使其成为单可用区部署。将多可用区部署设置更改为 No (否)。将数据库实例移到 VPC 后，再次对其进行修改，使其成为多可用区部署。有关更多信息，请参阅“修改 Amazon RDS 数据库实例”。
- 只读副本 – 当前不支持将未在 VPC 中的带只读副本的数据库实例移至 VPC 中。要将数据库实例移到 VPC，请首先删除其所有只读副本。将数据库实例移到 VPC 后，重新创建只读副本。有关更多信息，请参阅“使用数据库实例只读副本”。
- 选项组 – 如果您将数据库实例移到 VPC，并且数据库实例正在使用自定义选项组，请更改与数据库实例关联的选项组。选项组是特定于平台的，并且移动到 VPC 的操作是一种平台切换。要在这种情况下使用自定义选项组，请将默认 VPC 选项组分配给数据库实例，分配由作为移动目标的 VPC 中的其他数据库实例使用的选项组，或者创建新选项组并将它分配给数据库实例。有关更多信息，请参阅“使用选项组”。

将不在 VPC 中的数据库实例移动到具最短停机时间的 VPC 中的替代方案

使用以下替代方案，您可以将不在 VPC 中的数据库实例移到具有最短停机时间的 VPC 中。这些替代方案对源数据库实例造成的中断最小，并允许它在迁移期间为用户流量提供服务。但是，迁移到 VPC 所需的时间会因数据库大小和实时工作负载特性而异。

- Amazon Database Migration Service (Amazon DMS) – Amazon DMS 支持数据的实时迁移，同时保持源数据库实例全面运行，但它只复制一组有限的 DDL 语句。Amazon DMS 不传播索引、用户、权限、存储过程等项目以及与表数据不直接相关的其他数据库更改。此外，Amazon DMS 不会自动使用 RDS 快照创建初始数据库实例，这会增加迁移时间。有关更多信息，请参阅 Amazon Database Migration Service。
- 数据库快照还原或时间点恢复 – 您可以通过还原数据库实例的快照或将数据库实例还原到某个时间点，将数据库实例移到 VPC。有关更多信息，请参阅 还原到数据库实例 和 将 Amazon RDS 的数据库实例还原到指定时间。

Amazon RDS 的配额和限制

接下来，您可以查找 Amazon RDS 的资源配额和命名约束的说明。

主题

- Amazon RDS 中的配额
- Amazon RDS 中的命名约束
- 最大数据库连接数
- Amazon RDS 中的文件大小限制

Amazon RDS 中的配额

每个Amazon区域的每个Amazon账户都有关于可以创建的 Amazon RDS 资源数量的配额。达到某一资源的配额时，再进行创建该资源的调用就会失败并引发异常。

下表列出了每个 Amazon 区域的资源及其配额。

名称	默认值	可调整	描述
每个数据库安全组的授权	每个受支持的区域：20 个	否	每个数据库安全组的安全组授权数
每个数据库集群的自定义终端节点	每个受支持的区域：5 个	是	在当前区域的此账户中，您可为每个 Aurora 数据库集群创建的自定义端点的最大数量。此值体现的是账户中数据库集群内自定义端点的最大数。账户中的其他数据库集群可能具有更少的自定义端点数。
自定义引擎版本	每个受支持的区域：40 个	是	当前区域中此账户中允许的自定义引擎版本的最大数目

名称	默认值	可调整	描述
数据库集群参数组	每个受支持的区域：50 个	是	数据库集群参数组的最大数目
数据库集群	每个受支持的区域：40 个	是	当前区域中此账户中允许的 Aurora 集群的最大数目
数据库实例数	ap-south-1：20 个 每个其他支持的区域：40 个	是	当前区域中此账户中允许的数据库实例的最大数目
数据库分片组	每个受支持的区域：5 个	是	当前区域中此账户中 Aurora Limitless 数据库分片组的最大数
数据库子网组	ap-south-1：20 个 每个其他支持的区域：50 个	是	数据库子网组的最大数目
数据 API HTTP 请求体大小	每个受支持的区域：4MB	否	HTTP 请求正文允许的最大大小。
数据 API 最大并发集群密钥对数	每个受支持的区域：30 个	否	当前 Amazon 区域中对此账户的并发数据 API 请求中 Aurora Serverless v1 数据库集群和密钥的唯一对的最大数量。

名称	默认值	可调整	描述
数据 API 最大并发请求数	每个受支持的区域：500 个	否	针对 Aurora Serverless v1 数据库集群的数据 API 请求的最大数量，这些请求使用相同密钥且可以同时处理。当处理中的请求完成时，将对其他请求进行排队和处理。
Data API 最大结果集大小	每个受支持的区域：1MB	否	数据 API 可以返回的数据库结果集的最大大小。
数据 API 的 JSON 响应字符串的最大大小	每个受支持的区域：10 MB	否	RDS 数据 API 返回的简化 JSON 响应字符串的最大大小。
每秒 Data API 请求数	每个受支持的区域：每秒 1000 个	否	当前 Amazon 区域中此账户每秒允许的针对数据 API 的最大请求数。此限额仅适用于 Amazon Aurora Serverless v1 集群。
事件订阅	每个受支持的区域：20 个	是	事件订阅的最大数目
每个数据库集群的 IAM 角色	每个受支持的区域：5 个	是	与数据库集群关联的 IAM 角色的最大数目
每个数据库实例的 IAM 角色	每个受支持的区域：5 个	是	与数据库实例关联的 IAM 角色的最大数目
集成	每个受支持的区域：100 个	否	当前 Amazon 区域内此账户中支持的最大集成数量

名称	默认值	可调整	描述
手动数据库集群快照	每个受支持的区域：100 个	是	手动数据库集群快照的最大数目
手动数据库实例快照数	每个受支持的区域：100 个	是	手动数据库实例快照的最大数目
选项组	每个受支持的区域：20 个	是	选项组的最大数目
参数组	ap-south-1：20 个 每个其他支持的区域：50 个	是	参数组的最大数目
代理	每个受支持的区域：20 个	是	当前 Amazon 区域中此账户中允许的代理的最大数目
每个主数据库实例的只读副本数	每个受支持的区域：15 个	是	每个主数据库实例的只读副本的最大数量。Amazon Aurora 的此限额无法调整。
预留数据库实例	每个受支持的区域：40 个	是	当前 Amazon 区域中此账户中允许的预留数据库实例的最大数目
安全组	ap-south-1：20 个 每个其他支持的区域：25 个	是	数据库安全组的最大数目
每个数据库子网组的子网数	每个受支持的区域：20 个	否	每个数据库子网组的子网的最大数目

名称	默认值	可调整	描述
所有数据库实例的总存储空间	每个受支持的区域：100,000 GB	是	EBS 卷上所有 Amazon RDS 数据库实例的最大总存储空间（以 GB 为单位）此限额不适用于 Amazon Aurora，每个数据库集群的最大集群容量为 128TiB。

Note

默认情况下，您最多可以有 40 个数据库实例。RDS 数据库实例、Aurora 数据库实例、Amazon Neptune 实例和 Amazon DocumentDB 实例适用于此配额。
以下限制适用于 Amazon RDS 数据库实例：

- 每个 SQL Server 版本（Enterprise、Standard、Web 和 Express）具有 10 个处于“附带许可”模式的实例
- Oracle 具有 10 个处于“附带许可”模式的实例
- Db2 具有 40 个处于“自带许可”（BYOL）许可模式的实例
- MySQL、MariaDB 或 PostgreSQL 具有 40 个实例
- Oracle 具有 40 个处于“自带许可”(BYOL) 许可模式的实例

如果您的应用程序需要更多数据库实例，则可以通过打开 Service Quotas 控制台请求其他数据库实例。在导航窗格中，选择 Amazon 服务。选择 Amazon Relational Database Service (Amazon RDS) (Amazon 关系数据库服务 (Amazon RDS))，选择配额，然后按照说明请求增加配额。有关更多信息，请参阅《Service Quotas 用户指南》中的请求增加配额。
对于 RDS for Oracle，在每个区域中，您可以为每个源数据库实例创建最多 15 个只读副本，但我们建议将副本数量限制为 5 个，以尽可能减少复制延迟。
由 Amazon Backup 管理的备份被视为手动数据库快照，但不计入手动快照限额。有关 Amazon Backup 的更多信息，请参阅《Amazon Backup 开发人员指南》。

请注意，一个 Amazon Web Services 账户的跨区域自动备份的默认配额为 20，对于一个 Amazon Web Services 账户，每个区域的并发快照复制请求数量的默认配额为 20。RDS 无法针对以下 Amazon Web Services 区域调整此配额：

- 非洲（开普敦）
- 亚太地区（香港）
- 亚太地区（海得拉巴）
- 亚太地区（墨尔本）
- 加拿大西部（卡尔加里）
- 欧洲（苏黎世）

如果您使用任何 RDS API 操作并超过每秒调用数的默认限额，Amazon RDS API 会发出如下所示的错误。

ClientError: An error occurred (ThrottlingException) when calling the *API_name* operation: Rate exceeded.

此处，请减少每秒调用数。配额旨在涵盖大多数使用案例。如果需要更高的配额，可以使用以下任一选项请求增加配额：

- 在控制台中，打开服务配额控制台。
- 从 Amazon CLI 中，使用 request-service-quota-increase Amazon CLI 命令。

有关更多信息，请参阅《Service Quotas 用户指南》。

Amazon RDS 中的命名约束

Amazon RDS 中的命名约束如下所示：

- 数据库实例标识符：
  - 必须包含 1–63 个字母数字字符或连字符。
  - 第一个字符必须是字母。
  - 不能以连字符结尾，也不能包含两个连续连字符。
  - 对于每个 Amazon 区域的每个 Amazon 账户的所有数据库实例必须是唯一的。
- 初始数据库名称：

  - 数据库名称约束因数据库引擎而异。有关更多信息，请参阅创建每个数据库实例时的可用设置。
  - SQL Server – 在创建数据库实例后创建数据库。
- 主用户名 – 主用户名约束因每个数据库引擎而异。有关更多信息，请参阅创建数据库实例时的可用设置。
- 主密码：
  - 数据库主用户的密码可以包括除 /、'、"、@ 或空格之外的任意可打印 ASCII 字符。

    对于 Oracle，& 是额外的字符限制。
  - 密码可能包含以下数量的可打印 ASCII 字符，具体取决于数据库引擎：
    - Db2：8–255
    - MariaDB 和 MySQL：8–41
    - Oracle：8–30
    - SQL Server 和 PostgreSQL：8–128
- 数据库参数组：
  - 必须包含 1–255 个字母数字字符。
  - 第一个字符必须是字母。
  - 允许使用连字符，但名称不能以连字符结束或包含两个连续的连字符。
- 数据库子网组：
  - 必须包含 1–255 个字符。
  - 允许使用字母数字字符、空格、连字符、下划线和句点。

最大数据库连接数

数据库的同时最大连接数量因数据库引擎类型和数据库实例类的内存分配情况而异。最大连接数通常在与数据库实例关联的参数组中设置。例外情况是 Microsoft SQL Server，此时，此参数是在 SQL Server Management Studio（SSMS）中数据库实例的服务器属性中设置的。

数据库连接占用内存。将其中一个参数设置得太高可能会导致内存不足的情况，从而可能会导致数据库实例处于 incompatible-parameters 状态。有关更多信息，请参阅 诊断并解决内存限制的不兼容参数状态。

如果您的应用程序经常打开和关闭连接，或使大量长期连接保持打开，建议您使用 Amazon RDS 代理。RDS 代理是一种完全托管的高可用性数据库代理，它使用连接池安全有效地共享数据库连接。要了解有关 RDS 代理的更多信息，请参阅 的 Amazon RDS 代理。

> **Note**
>
> 对于 Oracle，您可以设置用户进程以及用户和系统会话的最大数量。
>
> 对于 Db2，您无法设置最大连接数。限制为 64000。

下表显示了有关不同数据库引擎的最大数据库连接数的信息。

数据库引擎	参数	允许的值	默认值	描述
MariaDB	`max_connections`	1–100000	• 对于 MariaDB 10.5 及更高版本，默认值为： LEAST({DBInstanceClassMemory/25165760},12000) 该公式实际上等同于 MB/25。 如果默认值计算得出的值大于 12,000，Amazon RDS 会将限制设置为 12,000。 • 对于 MariaDB 10.4 版本： {DBInstanceClassMemory/12582880} 该公式实际上等同于 MB/12。	允许的客户端同时连接数
MySQL	`max_connections`	1–100000	{DBInstanceClassMemory/12582880} 该公式实际上等同于 MB/12。	允许的客户端同时连接数

数据库引擎	参数	允许的值	默认值	描述
Oracle	`processes`	80–20000	LEAST({DBInstanceClassMemory/9868951}, 20000)	用户进程
Oracle	`sessions`	100–65535	不适用	用户和系统会话
PostgreSQL	`max_connections`	6–262143	LEAST({DBInstanceClassMemory/9531392}, 5000)	最大并发连接数
SQL Server	`user connections`	0–32767	0（无限制）	最大并发连接数。有关更多信息，请参阅配置 user connections（服务器配置选项）。

`DBInstanceClassMemory` 以字节为单位。有关如何计算此值的详细信息，请参阅 指定数据库参数。由于为操作系统和 RDS 管理进程预留了内存，因此，此内存大小小于数据库实例类的硬件规格中所示的值（以 GiB 为单位）。

例如，某些数据库实例类有 8GiB 的内存，即 8,589,934,592 字节。对于在内存为 8GiB 的数据库实例类（例如 db.m7g.large）上运行的 MySQL 数据库实例，使用总内存的公式为 `8589934592/12582880=683`。但是，变量 `DBInstanceClassMemory` 会自动减去为操作系统和管理数据库实例的 RDS 进程保留的内存量。然后，将减法的余数除以 12,582,880。此计算的结果是：`max_connections` 的值约为 630，而不是 683。此值取决于数据库实例类和数据库引擎。

当 MariaDB 或 MySQL 数据库实例在小型数据库实例类（例如 db.t3.micro 或 db.t3.small）上运行时，可用内存总量会很低。对于这些数据库实例类，RDS 会预留很大一部分可用内存，这会影响值 `max_connections`。例如，对于在 db.t3.micro 数据库实例类上运行的 MySQL 数据库实例，默认的最大连接数约为 60。您可以通过连接数据库 MariaDB 或 MySQL 数据库实例并运行以下 SQL 命令来确定相应实例的 `max_connections` 值：

```
SHOW GLOBAL VARIABLES LIKE 'max_connections';
```

Amazon RDS 中的文件大小限制

文件大小限制适用于某些 Amazon RDS 数据库实例。有关更多信息，请参阅以下特定于引擎的限制：

- Amazon RDS 中的 MariaDB 文件大小限制
- Amazon RDS 中的 MySQL 文件大小限制
- Amazon RDS 中的 Oracle 文件大小限制

Amazon RDS 故障排除

利用以下部分帮助排查您使用 Amazon RDS 和 Amazon Aurora 中的数据库实例时遇到的问题。

主题

有关使用 Amazon RDS API 调试问题的信息，请参阅对 Amazon RDS 上的应用程序进行故障排除。

无法连接到 Amazon RDS 数据库实例

当您无法连接到数据库实例时，常见原因如下：

- 入站规则 – 由本地防火墙强制执行的访问规则与您授权用于访问数据库实例的 IP 地址可能不匹配。该问题很可能是由安全组中的入站规则所致。

  默认情况下，数据库实例不允许访问。通过与 VPC 关联的安全组授予访问权限，该安全组允许流量进出数据库实例。如有必要，请将针对您具体情况的入站和出站规则添加到安全组。您可以指定一个 IP 地址、IP 地址范围或另一个 VPC 安全组。

  > Note
  >
  > 添加新入站规则时，您可以为源选择我的 IP，以允许从浏览器中检测到的 IP 地址访问数据库实例。

有关设置安全组的更多信息，请参阅通过创建安全组提供对 VPC 中的数据库实例的访问。

> **Note**
>
> 不允许来自 169.254.0.0/16 范围内的 IP 地址的客户端连接。这是自动私有 IP 寻址范围 (APIPA)，它用于本地链路寻址。

- 公开可用性 – 要从 VPC 外部连接到您的数据库实例（例如通过使用客户端应用程序），实例必须具有向其分配的公有 IP 地址。

  要使实例公开可用，请修改它，在公开可用性下选择是。有关更多信息，请参阅 对互联网隐藏 VPC 中的数据库实例。
- 端口 – 由于您的本地防火墙限制，无法使用您在创建数据库实例时指定的端口来发送和接收通信。如需确定您的网络是否允许指定端口用于入站和出站通信，请咨询您的网络管理员。
- 可用性 – 对于新创建的数据库实例，数据库实例具有 `creating` 状态，直到该数据库实例可供使用。当状态变为 `available` 时，您可以连接到该数据库实例。根据数据库实例的大小，可能要用最长 20 分钟，实例才可用。
- 互联网网关 – 对于希望可公开访问的数据库实例，其数据库子网组中的子网必须具有互联网网关。

  为子网配置互联网网关

  1. 登录Amazon Web Services Management Console并通过以下网址打开 Amazon RDS 控制台：https://console.aws.amazon.com/rds/。
  2. 在导航窗格中，选择 Databases (数据库)，然后选择数据库实例的名称。
  3. 在 Connectivity & security (连接和安全) 选项卡中，记录 VPC 下 VPC ID 的值以及 Subnets (子网) 下子网 ID 的值。
  4. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
  5. 在导航窗格中，选择 Internet Gateways（互联网网关）。验证是否有 Internet 网关附加到您的 VPC。否则，选择 Create Internet Gateway 以创建 Internet 网关。选择 Internet 网关，然后选择 Attach to VPC 并按照说明将其附加到您的 VPC。
  6. 在导航窗格中，选择 Subnets，然后选择您的子网。
  7. 在 Route Table 选项卡上，验证 `0.0.0.0/0` 的路由是否为目的地以及您的 VPC 的 Internet 网关是否为目标。

如果您使用实例的 IPv6 地址连接到实例，请检查是否有一个路由可以将所有 IPv6 流量 (`::/0`) 指向 Internet 网关。否则请执行以下操作：

a. 选择路由表的 ID (rtb-xxxxxxxx) 以导航到路由表。

b. 在 Routes（路由）选项卡上，选择 Edit routes（编辑路由）。选择 Add route (添加路由)，将 `0.0.0.0/0` 用作目的地并将 Internet 网关用作目标。

   对于 IPv6，选择 Add route (添加路由)，将 `::/0` 用作目的地并将 Internet 网关用作目标。

c. 选择 Save routes（保存路由）。

此外，如果您尝试连接到 IPv6 端点，请确保客户端 IPv6 地址范围已获得连接到数据库实例的授权。

有关更多信息，请参阅 在 VPC 中使用数据库实例。

有关引擎特定的连接问题，请参阅以下主题：

- 排除与 SQL Server 数据库实例的连接故障
- 排除与 Oracle 数据库实例的连接故障
- 排除与您的 RDS for PostgreSQL 实例的连接故障
- 最大 MySQL 和 MariaDB 连接

测试与数据库实例的连接

您可以使用常见 Linux 或 Microsoft Windows 工具测试与数据库实例的连接。

从 Linux 或 Unix 终端，您可以通过输入以下命令来测试连接。将 *DB-instance-endpoint* 替换为端点，并将 *port* 替换为数据库实例的端口。

```
nc -zv DB-instance-endpoint port
```

例如，下面显示了示例命令和返回值。

```
nc -zv postgresql1.c6c8mn7fake0.us-west-2.rds.amazonaws.com 8299
```

```
  Connection to postgresql1.c6c8mn7fake0.us-west-2.rds.amazonaws.com 8299 port [tcp/
vvr-data] succeeded!
```

Windows 用户可使用 Telnet 测试与数据库实例的连接。Telnet 操作不支持除测试连接之外的用途。如果连接成功，则该操作不返回任何消息。如果连接失败，则您将收到一条错误消息，诸如以下内容。

```
C:\>telnet sg-postgresql1.c6c8mntfake0.us-west-2.rds.amazonaws.com 819

  Connecting To sg-postgresql1.c6c8mntfake0.us-west-2.rds.amazonaws.com...Could not
 open
  connection to the host, on port 819: Connect failed
```

如果 Telnet 操作返回成功值，则您的安全组配置正确。

> **Note**
>
> Amazon RDS 不接受 Internet 控制消息协议 (ICMP) 流量，包括 ping。

排查连接身份验证错误

在某些情况下，您可以连接到数据库实例，但会收到身份验证错误。在这些情况下，您可能需要重置数据库实例的主用户密码。您可以通过修改 RDS 实例来做到这一点。

有关修改 数据库实例的更多信息，请参阅 修改 Amazon RDS 数据库实例。

Amazon RDS 安全性问题

为避免出现安全性问题，切勿为用户账户使用您的 Amazon Web Services 账户根用户电子邮件地址和密码。最佳做法是使用根用户来创建用户，并将这些用户分配给数据库用户账户。还可使用根用户创建其他用户账户（如有必要）。

有关创建用户的信息，请参阅在 Amazon Web Services 账户 中创建 IAM 用户。有关在 Amazon IAM Identity Center 中创建用户的信息，请参阅管理 IAM Identity Center 中的身份。

错误消息“无法检索账户属性，某些控制台功能可能受损。”

您可能会由于几个原因而收到此错误。这可能是因为您的账户缺少权限，或者您的账户尚未正确设置。如果您的账户是新账户，可能该账户还没准备好。如果是现有账户，您的访问策略中可能缺少执行特定

操作的权限（如，创建数据库实例）。要修复该问题，您的管理员需要为您的账户提供必要的角色。有关更多信息，请参阅 IAM 文档。

排查网络不兼容状态问题

网络不兼容状态意味着可能仍然可以在数据库级别访问数据库，但您无法对其进行修改或重启。

原因

您的数据库实例的网络不兼容状态可能是以下操作之一造成的：

- 修改数据库实例类。
- 修改数据库实例以使用多可用区数据库集群部署。
- 因维护事件更换主机。
- 启动替换数据库实例。
- 从快照备份进行还原。
- 启动已停止的数据库实例。

解决方案

使用 start-db-instance 命令

要修复处于网络不兼容状态的数据库，请按照以下说明进行操作：

1. 打开 https://console.aws.amazon.com/rds/，然后从导航窗格中选择数据库。
2. 从连接和安全选项卡中选择处于网络不兼容状态的数据库实例，并记下数据库实例标识符、VPC ID 和子网 ID。
3. 使用 Amazon CLI 运行 `start-db-instance` 命令。指定 `--db-instance-identifier` 值。

> **Note**
>
> 当数据库处于不兼容模式时运行此命令可能会导致一些停机时间。
>
> `start-db-instance` 命令无法解决 RDS for SQL Server 数据库实例的这一问题。

如果命令成功执行，则您的数据库状态更改为可用。

如果您的数据库重新启动，则数据库实例可能会在实例变为网络不兼容状态之前，执行上一次在实例上运行的操作。这可能会将实例移回网络不兼容状态。

如果 `start-db-instance` 命令未成功或实例移回网络不兼容状态，请在 RDS 控制台中打开数据库页面并选择表。导航到日志和事件部分。近期事件部分显示了要遵循的进一步解决步骤。消息分类如下：

- 内部资源检查：您的内部资源可能存在问题。
- DNS 检查：在 VPC 控制台中检查 VPC 的 DNS 解析和主机名。
- ENI 检查：数据库的弹性网络接口（ENI）可能不存在。
- 网关检查：公开可用的数据库的互联网网关未连接到 VPC。
- IP 检查：子网中没有可用的 IP 地址。
- 安全组检查：没有与您的数据库关联的安全组或安全组无效。
- 子网检查：您的数据库子网组中没有有效的子网，或者您的子网存在问题。
- VPC 检查：与您的数据库关联的 VPC 无效。

执行时间点恢复

最佳做法是进行备份（快照或逻辑），以防数据库进入网络不兼容状态。请参阅备份简介。如果启用了自动备份，则暂时停止对数据库的任何写入并执行时间点恢复。

> **Note**
> 实例进入网络不兼容状态后，可能无法访问该数据库实例来执行逻辑备份。

如果您没有开启自动备份，请创建新的数据库实例。然后，使用 Amazon Database Migration Service（Amazon DMS）或使用备份和还原工具迁移数据。

如果这仍无法解决问题，请联系 Amazon Web Services 支持 寻求进一步的帮助。

重置数据库实例拥有者密码

如果您被锁定在数据库实例之外，则可以以主用户身份登录。然后，您可以重置其他管理用户或角色的凭证。如果您无法以主用户身份登录，Amazon 账户所有者可以重置主用户密码。有关可能需要重置哪些管理账户或角色的详细信息，请参阅主用户账户权限。

您可以使用 Amazon RDS 控制台、Amazon CLI 命令 modify-db-instance，或者使用 ModifyDBInstance API 操作更改数据库实例密码。有关修改 数据库实例的更多信息，请参阅 修改 Amazon RDS 数据库实例。

Amazon RDS 数据库实例中断或重新引导

重新启动数据库实例时可能会发生数据库实例中断。在将数据库实例置于某种禁止对其访问的状态，以及重新启动数据库时会出现这种情况。当您手动重启数据库实例时，可能会发生重启。当您更改需要在重启后才生效的数据库实例设置时，也可能会发生重启。

当您更改需要重启的设置或手动引起重启时，才会发生数据库实例重启。如果您更改某个设置并请求此更改立即生效，则可能会立即发生重启。或者，可能在数据库实例的维护时段内发生重启。

在出现下列情况之一时，会立即发生数据库实例重启：

- 您将数据库实例的备份保留期从 0 更改为非零值或从非零值更改为 0。然后，将 Apply Immediately（立即应用）设置为 `true`。
- 更改数据库实例类，并将立即应用设置为 `true`。
- 将存储类型从磁性（标准）更改为通用型 (SSD) 或预置 IOPS (SSD)，或从预置 IOPS (SSD) 或通用型 (SSD) 更改为磁性（标准）。

在维护时段内，如果出现下列情况之一，则会发生数据库实例重启：

- 将数据库实例的备份保留期从 0 更改为非零值或从非零值更改为 0 并将立即应用设置为 `false`。
- 更改数据库实例类，并将立即应用设置为 `false`。

当您更改数据库参数组中的静态参数时，在与该参数组关联的数据库实例重启之前，此更改将不会生效。更改需要手动重新启动。在维护时段内，数据库实例不会自动重新启动。

要查看显示数据库实例操作的表以及设置立即应用值的效果，请参阅修改 Amazon RDS 数据库实例。

Amazon RDS 数据库参数更改未生效

在某些情况下，您可能会更改数据库参数组中的参数，但未看到更改生效。如果是这样，您可能需要重新启动与数据库参数组关联的数据库实例。当您更改动态参数时，更改将立即生效。当您更改静态参数时，在与该参数组关联的数据库实例重启之前，此更改将不会生效。

您可以使用 RDS 控制台重启数据库实例。或者，您可以显式调用 RebootDBInstance API 操作。如果数据库实例处于多可用区部署中，则可以在不进行失效转移的情况下重启。在静态参数更改后重新引导关联的数据库实例的要求，可帮助缓解影响 API 调用的参数误配置的风险。调用 ModifyDBInstance 以更改数据库实例类就是这样的一个例子。有关更多信息，请参阅 在 Amazon RDS 中修改数据库参数组中的参数。

Amazon RDS 数据库实例用完存储空间

如果数据库实例用完存储空间，则该实例可能不再可用。我们强烈建议您持续监控 CloudWatch 内发布的 FreeStorageSpace 指标，以确保数据库实例具有足够的可用存储空间。

如果数据库实例的存储空间不足，其状态将更改为 storage-full。例如，为存储空间已用完的数据库实例调用 DescribeDBInstances API 操作将输出以下内容。

```
aws rds describe-db-instances --db-instance-identifier mydbinstance

DBINSTANCE  mydbinstance  2009-12-22T23:06:11.915Z  db.m5.large  mysql8.0  50  sa
storage-full  mydbinstance.clla4j4jgyph.us-east-1.rds.amazonaws.com  3306
us-east-1b  3
 SECGROUP  default  active
 PARAMGRP  default.mysql8.0  in-sync
```

要从这种情况下恢复，请使用 ModifyDBInstance API 操作或以下 Amazon CLI 命令为实例添加更多存储空间。

对于 Linux、macOS 或 Unix：

```
aws rds modify-db-instance \
    --db-instance-identifier mydbinstance \
    --allocated-storage 60 \
    --apply-immediately
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier mydbinstance ^
    --allocated-storage 60 ^
    --apply-immediately
```

```
DBINSTANCE  mydbinstance  2009-12-22T23:06:11.915Z  db.m5.large  mysql8.0  50  sa
storage-full  mydbinstance.clla4j4jgyph.us-east-1.rds.amazonaws.com  3306
us-east-1b  3  60
 SECGROUP  default  active
 PARAMGRP  default.mysql8.0  in-sync
```

现在，当您描述您的数据库实例时，您将看到数据库实例处于 modifying 状态，这表示正在扩展存储。

```
aws rds describe-db-instances --db-instance-identifier mydbinstance
```

```
DBINSTANCE  mydbinstance  2009-12-22T23:06:11.915Z  db.m5.large  mysql8.0  50  sa
modifying  mydbinstance.clla4j4jgyph.us-east-1.rds.amazonaws.com
3306  us-east-1b  3  60
 SECGROUP  default  active
 PARAMGRP  default.mysql8.0  in-sync
```

存储扩展完成后，数据库实例的状态将更改为 available。

```
aws rds describe-db-instances --db-instance-identifier mydbinstance
```

```
DBINSTANCE  mydbinstance  2009-12-22T23:06:11.915Z  db.m5.large  mysql8.0  60  sa
available  mydbinstance.clla4j4jgyph.us-east-1.rds.amazonaws.com  3306
us-east-1b  3
 SECGROUP  default  active
 PARAMGRP  default.mysql8.0  in-sync
```

使用 DescribeEvents 操作，您可在存储空间用完后接收通知。例如，在这种情况下，如果在完成这些操作后执行 DescribeEvents 调用，您将看到以下输出。

```
aws rds describe-events --source-type db-instance --source-identifier mydbinstance
```

```
2009-12-22T23:44:14.374Z  mydbinstance  Allocated storage has been exhausted db-
instance
2009-12-23T00:14:02.737Z  mydbinstance  Applying modification to allocated storage db-
instance
2009-12-23T00:31:54.764Z  mydbinstance  Finished applying modification to allocated
 storage
```

Amazon RDS 可用数据库实例不足

在您尝试创建、启动或修改数据库实例时，可能返回 InsufficientDBInstanceCapacity 错误。在您尝试从数据库快照还原数据库实例时，也可能返回此错误。返回此错误时，常见原因是特定的数据库实例类在请求的可用区中不可用。您可以尝试以下方法之一来解决问题：

- 使用不同的数据库实例类重试请求。
- 使用不同的可用区返回请求。
- 重试请求而不明确指定可用区。

有关排查 Amazon EC2 的实例容量问题的信息，请参阅《Amazon EC2 用户指南》中的实例容量不足。

有关修改数据库实例的信息，请参阅修改 Amazon RDS 数据库实例。

Amazon RDS 中的可用内存问题

可用内存 是数据库实例上可供数据库引擎使用的总随机存取内存 (RAM)。这是可用操作系统 (OS) 内存以及可用缓冲区和页面缓存的总和。数据库引擎使用主机上的大部分内存，但操作系统进程也使用一些 RAM。当前分配给数据库引擎或操作系统进程使用的内存不包括在可用内存中。当数据库引擎内存不足时，数据库实例可以使用通常用于缓冲和缓存的临时空间。如前所述，这个临时空间包含在可用内存中。

您可以使用 Amazon CloudWatch 中的 FreeableMemory 指标来监控可用内存。有关更多信息，请参阅 Amazon RDS 的监控工具。

如果数据库实例始终可用内存不足或使用交换空间，则考虑纵向扩展至更大的数据库实例类。有关更多信息，请参阅数据库实例类。

您还可以更改内存设置。例如，在 RDS for MySQL 上，您可以调整 innodb_buffer_pool_size 参数的大小。默认情况下，此参数设置为物理内存的 75%。有关更多 MySQL 故障排除提示，请参阅如何排查 Amazon RDS for MySQL 数据库中的可用内存不足问题？

MySQL 和 MariaDB 问题

可以使用 MySQL 和 MariaDB 数据库实例诊断和纠正问题。

主题

最大 MySQL 和 MariaDB 连接

允许与 RDS for MySQL 或 RDS for MariaDB 数据库实例的最大连接数，基于可用于其数据库实例类的内存量。可用于数据库实例类的内存量越大，允许的连接数就越多。有关数据库实例类的更多信息，请参阅数据库实例类。

默认情况下，数据库实例的连接限制设置为数据库实例类的最大数量。您可以将并行连接的数量限制为不超过所允许的最大连接数的任何值。在参数组中为数据库实例使用 `max_connections` 参数。有关更多信息，请参阅“最大数据库连接数”和“Amazon RDS 的参数组”。

您可以通过运行以下查询来检索 MySQL 或 MariaDB 数据库实例允许的最大连接数。

```
SELECT @@max_connections;
```

您可以通过运行以下查询来检索与 MySQL 或 MariaDB 数据库实例的活动连接数。

```
SHOW STATUS WHERE `variable_name` = 'Threads_connected';
```

诊断并解决内存限制的不兼容参数状态

当满足以下条件时，可以将 MariaDB 或 MySQL 数据库实例置于 incompatible-parameters 状态以获得内存限制：

- 当数据库实例状态为可用时，数据库实例在一小时内至少重新启动三次，或在一天内至少重新启动五次。

- 尝试重新启动数据库实例失败，因为维护操作或监控过程无法重新启动数据库实例。
- 数据库实例的潜在内存使用量超过分配给其数据库实例类的内存的 1.2 倍。

当数据库实例在一小时内第三次重新启动或一天内第五次重新启动时，它将执行内存使用情况检查。该检查将计算数据库实例的潜在内存使用情况。计算返回的值是以下值的总和：

- 值 1 – 以下参数的总和：
  - innodb_additional_mem_pool_size
  - innodb_buffer_pool_size

    您可以修改 innodb_buffer_pool_size 的值。但是，该值并不总是与您输入的值相匹配。出现这种不匹配有以下几个原因。首先，如果数据库实例是微型数据库实例，我们会覆盖默认值并将其设置为 256 MB。有关更多信息，请参阅 [覆盖 innodb_buffer_pool_size]。

    其次，我们确保在数据库实例上为主机管理器、引擎、操作系统和内核预留 500 MB 的内存。

    最后，我们通过将 innodb_buffer_pool_size 划分为单位来对其进行优化。主机管理器向下舍入到最接近这些单位的倍数。单位是通过将 innodb_buffer_pool_chunk_size 乘以 innodb_buffer_pool_instances 计算得出的。有关更多信息，请参阅 MySQL 文档中的[配置 InnoDB 缓冲池大小]。

    innodb_buffer_pool_instances 的默认值为 8，除非 innodb_buffer_pool_size 小于 1 GB。如果 innodb_buffer_pool_size 小于 1 GB，则 innodb_buffer_pool_instances 的默认值为 1。innodb_buffer_pool_chunk_size 的默认值为 128 MB。
  - innodb_log_buffer_size
  - key_buffer_size
  - query_cache_size（仅限 MySQL 版本 5.7）
  - tmp_table_size
- 值 2 – max_connections 参数乘以下参数的总和：
  - binlog_cache_size
  - join_buffer_size
  - read_buffer_size
  - read_rnd_buffer_size
  - sort_buffer_size

  - thread_stack
- 值 3 – 如果 performance_schema 参数已启用，则将 max_connections 参数乘以 429498。

  如果 performance_schema 参数被禁用，则此值为零。

因此，计算返回的值如下：

Value 1 + Value 2 + Value 3

当此值超过分配给数据库实例使用的数据库实例类的内存的 1.2 倍时，数据库实例将处于 incompatible-parameters 状态。有关分配给数据库实例类的内存的信息，请参阅 数据库实例类的硬件规格。

计算将参数的值乘以多个 max_connections 参数的总和。如果 max_connections 参数设置为较大的值，则可能会导致检查返回过高的数据库实例潜在内存使用量值。在这种情况下，请考虑降低 max_connections 参数的值。

要解决此问题，请完成以下步骤：

1. 调整与数据库实例关联的数据库参数组中的内存参数。为此，使潜在的内存使用量低于分配给其数据库实例类的内存的 1.2 倍。

   有关设置参数的信息，请参阅在 Amazon RDS 中修改数据库参数组中的参数。
2. 重新启动数据库实例。

   有关设置参数的信息，请参阅 启动之前停止的 Amazon RDS 数据库实例。

诊断并解决只读副本之间的滞后

在创建一个 MySQL 或 MariaDB 只读副本且该只读副本可用后，Amazon RDS 首先将复制自只读副本创建操作启动以来对源数据库实例所做的更改。在此期间，只读副本的复制滞后时间将大于 0。您可以在 Amazon CloudWatch 中通过查看 Amazon RDS ReplicaLag 指标来监控此滞后时间。

ReplicaLag 指标报告 MariaDB 或 MySQL Seconds_Behind_Master 命令的 SHOW REPLICA STATUS 字段的值。有关更多信息，请参阅 MySQL 文档中的 SHOW REPLICA STATUS 语句。

当 ReplicaLag 指标达到 0 时，即表示副本已赶上源数据库实例进度。如果 ReplicaLag 指标返回 -1，则副本可能为未激活状态。要纠正复制错误，请参阅诊断并解决 MySQL 或 MariaDB 读取复制故障。ReplicaLag 值为 -1 还可能意味着 Seconds_Behind_Master 值无法确定或为 NULL。

> **Note**
>
> 以前的 MariaDB 版本使用的是 `SHOW SLAVE STATUS`，而不是 `SHOW REPLICA STATUS`。如果您使用的 MariaDB 版本低于 10.5，请使用 `SHOW SLAVE STATUS`。

`ReplicaLag` 指标将在网络中断期间或在维护时段内应用修补程序时返回 -1。在这种情况下，在再次检查 `ReplicaLag` 指标之前，需等待网络连接恢复或维护时段结束。

MySQL 和 MariaDB 只读复制技术是异步的。因此，预计源数据库实例上的 `BinLogDiskUsage` 指标和只读副本上的 `ReplicaLag` 指标偶尔会增加。例如，请考虑对源数据库实例并行进行大量写入操作的情况。同时，对只读副本的写入操作会使用单个 I/O 线程序列化。这种情况可能会导致源实例与只读副本之间的滞后。

有关只读副本和 MySQL 的更多信息，请参阅 MySQL 文档中的[复制实施详细信息]。有关只读副本和 MariaDB 的更多信息，请参阅 MariaDB 文档中的[复制概述]。

您可降低对源数据库实例的更新与对只读副本的后续更新之间的滞后，方式如下：

- 将只读副本的数据库实例类的存储大小设置为与源数据库实例的类似。
- 确保源数据库实例和只读副本使用的数据库参数组中的参数设置兼容。有关更多信息和示例，请参阅下一部分中有关 `max_allowed_packet` 参数的讨论。
- 禁用查询缓存。对于经常修改的表，使用查询缓存可能会加大副本滞后，因为缓存已锁定且会频繁刷新。如果是这样的话，您可能会发现在禁用查询缓存的情况下副本滞后较少。您可通过在数据库实例的数据库参数组中将 `query_cache_type parameter` 设置为 0 来禁用查询缓存。有关查询缓存的更多信息，请参阅[查询缓存配置]。
- 为 MySQL 或 MariaDB 的 InnoDB 预热只读副本上的缓冲池。例如，假设您有少量经常更新的表，并且您正在使用 InnoDB 或 XtraDB 表架构。在这种情况下，将这些表转储到只读副本上。这样做将促使数据库引擎从磁盘扫描这些表的行，然后将它们缓存到缓冲池中。此方法可以减少副本滞后。下面是一个示例。

  对于 Linux、macOS 或 Unix：

  ```
  PROMPT> mysqldump \
      -h <endpoint> \
      --port=<port> \
      -u=<username> \
      -p <password> \
  ```

```
    database_name table1 table2 > /dev/null
```

对于 Windows：

```
PROMPT> mysqldump ^
    -h <endpoint> ^
    --port=<port> ^
    -u=<username> ^
    -p <password> ^
    database_name table1 table2 > /dev/null
```

诊断并解决 MySQL 或 MariaDB 读取复制故障

Amazon RDS 会监控只读副本的复制状态。如果由于任何原因停止复制，则 RDS 将只读副本实例的 Replication State（复制状态）字段更新为 `Error`。您可通过查看复制错误字段，检查 MySQL 或 MariaDB 引擎引发的关联错误的详细信息。还生成指示只读副本状态的事件，包括 RDS-EVENT-0045、RDS-EVENT-0046 和 RDS-EVENT-0057。有关这些事件和事件订阅的详细信息，请参阅 使用 Amazon RDS 事件通知。如果返回 MySQL 错误消息，则在 MySQL 错误消息文档中查看错误。如果返回 MariaDB 错误消息，则在 MariaDB 错误消息文档中查看错误。

可导致复制出错的常见情况包括：

- 只读副本的 `max_allowed_packet` 参数的值小于源数据库实例的 `max_allowed_packet` 参数。

  `max_allowed_packet` 参数是您可以在数据库参数组中设置的自定义参数。`max_allowed_packet` 参数用于指定可在数据库上运行的数据操作语言 (DML) 的最大大小。在某些情况下，源数据库实例的 `max_allowed_packet` 值可能大于只读副本的 `max_allowed_packet` 值。如果是这样，复制过程可能会引发错误并停止复制。最常见的错误是 `packet bigger than 'max_allowed_packet' bytes`。通过将源和只读副本设置为使用具有相同 `max_allowed_packet` 参数值的数据库参数组，即可更正此错误。
- 对只读副本上的表进行写入操作。如果是在只读副本上创建索引，则需要将 `read_only` 参数设置为 0 才能创建索引。如果对只读副本上的表进行写入操作，则会中断复制。
- 使用 MyISAM 等非事务性存储引擎。只读副本需要使用事务性存储引擎。只有以下存储引擎支持复制：适用于 MySQL 或 MariaDB 的 InnoDB。

  您可以使用以下命令将 MyISAM 表转换为 InnoDB：

  ```
  alter table <schema>.<table_name> engine=innodb;
  ```

- 使用不安全的不确定性查询，如 SYSDATE()。有关更多信息，请参阅 MySQL 文档中的确定二进制日志记录中的安全和不安全语句。

下列步骤可帮助您纠正复制错误：

- 如果您遇到逻辑错误并且您可安全跳过该错误，则可执行跳过 RDS for MySQL 的当前复制错误中所述的步骤。您的 MySQL 或 MariaDB 数据库实例必须运行包括 mysql_rds_skip_repl_error 过程的版本。有关更多信息，请参阅 mysql.rds_skip_repl_error。
- 如果您遇到二进制日志（binlog）位置问题，则可使用 mysql.rds_next_source_log（RDS for MySQL 主要版本 8.4 及更高版本） 或 mysql.rds_next_master_log（RDS for MariaDB 和 RDS for MySQL 主要版本 8.0 及更低版本） 命令更改副本重放位置。
- 由于 DML 负载高，您可能会遇到暂时的性能问题。如果是这样，您可以在只读副本上的数据库参数组中将 innodb_flush_log_at_trx_commit 参数设置为 2。这样做有助于只读副本保持同步，但这会临时降低原子性、一致性、隔离和持久性（ACID）属性。
- 您可以删除只读副本，使用相同的数据库实例标识符创建实例。如果您这样做，则端点将保持与旧只读副本的端点相同。

如果复制错误得到纠正，则复制状态将更改为正在复制。有关更多信息，请参阅“排查 MySQL 只读副本问题”。

在启用二进制日志记录的情况下创建触发器需要 SUPER 权限

当尝试在 RDS for MySQL 或 RDS for MariaDB 数据库实例中创建触发器时，您可能会收到以下错误。

```
"You do not have the SUPER privilege and binary logging is enabled"
```

在启用了二进制日志记录时使用触发器需要 SUPER 权限（对于 RDS for MySQL 和 RDS for MariaDB 数据库实例会限制该权限）。您可以通过将 log_bin_trust_function_creators 参数设置为 true，在启用了二进制日志记录时创建触发器而无需 SUPER 权限。要将 log_bin_trust_function_creators 设置为 true，请创建新的数据库参数组或修改现有数据库参数组。

您可以创建一个新的数据库参数组，这样，您就可以在启用了二进制日志记录的情况下，在 RDS for MySQL 或 RDS for MariaDB 数据库实例中创建触发器。为此，请使用以下 CLI 命令。要修改现有参数组，请从步骤 2 开始。

使用 CLI 创建新参数组以允许在启用了二进制日志记录时使用触发器

1. 创建新的参数组。

   对于 Linux、macOS 或 Unix：

   ```
   aws rds create-db-parameter-group \
       --db-parameter-group-name allow-triggers \
       --db-parameter-group-family mysql8.0 \
       --description "parameter group allowing triggers"
   ```

   对于 Windows：

   ```
   aws rds create-db-parameter-group ^
       --db-parameter-group-name allow-triggers ^
       --db-parameter-group-family mysql8.0 ^
       --description "parameter group allowing triggers"
   ```

2. 修改数据库参数组以允许使用触发器。

   对于 Linux、macOS 或 Unix：

   ```
   aws rds modify-db-parameter-group \
       --db-parameter-group-name allow-triggers \
       --parameters "ParameterName=log_bin_trust_function_creators,
    ParameterValue=true, ApplyMethod=pending-reboot"
   ```

   对于 Windows：

   ```
   aws rds modify-db-parameter-group ^
       --db-parameter-group-name allow-triggers ^
       --parameters "ParameterName=log_bin_trust_function_creators,
    ParameterValue=true, ApplyMethod=pending-reboot"
   ```

3. 修改数据库实例以使用新的数据库参数组。

   对于 Linux、macOS 或 Unix：

   ```
   aws rds modify-db-instance \
       --db-instance-identifier mydbinstance \
       --db-parameter-group-name allow-triggers \
   ```

```
    --apply-immediately
```

对于 Windows：

```
aws rds modify-db-instance ^
    --db-instance-identifier mydbinstance ^
    --db-parameter-group-name allow-triggers ^
    --apply-immediately
```

4. 要使更改生效，请手动重启数据库实例。

```
aws rds reboot-db-instance --db-instance-identifier mydbinstance
```

诊断并解决时间点还原故障

还原包含临时表的数据库实例

在尝试对 MySQL 或 MariaDB 数据库实例进行时间点还原 (PITR) 时，您可能遇到以下错误。

```
Database instance could not be restored because there has been incompatible database
 activity for restore
functionality. Common examples of incompatible activity include using temporary tables,
 in-memory tables,
or using MyISAM tables. In this case, use of Temporary table was detected.
```

PITR 依靠 MySQL 或 MariaDB 中的备份快照和二进制日志 (binlog) 将数据库实例还原为特定时间的状态。二进制日志中的临时表信息不可靠，并且可能导致 PITR 失败。如果在 MySQL 或 MariaDB 数据库实例中使用临时表，则可通过执行更频繁的备份来降低 PITR 失败的可能性。在创建临时表和下一次备份快照之间的时间点最可能发生 PITR 失败。

还原包含内存表的数据库实例

在还原具有内存表的数据库时，您可能会遇到问题。重新启动过程中将清空内存表。因此，您的内存表在重新启动后可能为空。我们建议在使用内存表时，构建解决方案以便在重新启动时处理空表。如果您将内存中表与复制的数据库实例一起使用，则可能需要在重新启动后重新创建只读副本。如果只读副本重新启动并且无法从空内存表中还原数据，则可能需要执行此操作。

有关备份和 PITR 的更多信息，请参阅备份简介和将 Amazon RDS 的数据库实例还原到指定时间。

复制已停止错误

调用 `mysql.rds_skip_repl_error` 命令时，您可能会收到一条错误消息，指出副本已关闭或禁用。

出现该错误消息是由于复制已停止且无法重新启动。

如果您需要跳过大量错误，复制滞后时间可能会超出二进制日志文件的默认保留期。在这种情况下，您可能会遇到一个因二进制日志文件在副本上进行重放之前被清除而引发的严重错误。此清除会导致复制停止，而您将无法再调用 `mysql.rds_skip_repl_error` 命令以跳过复制错误。

您可以通过增加二进制日志文件在复制源上保留的小时数来缓解该问题。在增加二进制日志保留时间后，您可以重新启动复制进程，并根据需要调用 `mysql.rds_skip_repl_error` 命令。

若要设置 binlog 保留时间，请使用 mysql.rds_set_configuration 过程。指定二进制日志保留小时数的配置参数以及要在数据库集群上保留二进制日志文件的小时数（最多 720 个小时，即 30 天）。以下示例将 binlog 文件的保留期设置为 48 个小时。

```
CALL mysql.rds_set_configuration('binlog retention hours', 48);
```

只读副本创建失败或复制中断，出现严重错误 1236

在更改 MySQL 或 MariaDB 数据库实例的默认参数值后，您可能会遇到下列问题之一：

- 您无法为数据库实例创建只读副本。
- 复制失败，出现 `fatal error 1236`。

MySQL 和 MariaDB 数据库实例的一些默认参数值有助于确保数据库符合 ACID，并且只读副本是崩溃安全的。它们在提交之前将事务写入二进制日志来确保每个提交完全同步，从而实现这一点。在尚未将事务写入二进制日志的情况下，如果更改这些参数的默认值来提高性能，则可能会导致复制失败。

要解决该问题，请设置以下参数值：

- `sync_binlog = 1`
- `innodb_support_xa = 1`
- `innodb_flush_log_at_trx_commit = 1`

只读副本复制未能初始化元数据结构

在您尝试启动复制时，收到了以下错误消息：

```
Read Replica Replication Error - SQLError: 13124, reason: Replica failed to initialize
 applier metadata structure from the repository
```

当副本的元数据结构有问题时，就会出现此错误。要修复元数据结构，您必须创建新副本。

为了防止以后出现此问题，请执行以下操作之一：

- 如果可能，请在副本上禁用多线程。从 MySQL 8.0.27 开始，默认情况下启用了多线程。
- 如果您需要在副本上使用多线程，建议您使用基于 GTID 的复制。有关更多信息，请参阅 使用基于 GTID 的复制。

无法将备份保留期设置为 0

您可能出于多个原因需要将备份保留期设置为 0。例如，您可以通过将保留期设置为 0 来立即禁用自动备份。

在某些情况下，您可能会将值设置为 0 并收到一条消息，说明保留期必须介于 1 到 35 之间。在这些情况下，请检查以确保您没有为实例设置只读副本。只读副本需要备份来管理只读副本日志，因此不能将保留期设置为 0。

Amazon RDS API 参考

除了 Amazon Web Services Management Console 和 Amazon Command Line Interface（Amazon CLI）之外，Amazon RDS 还提供了一个 API。您可以使用 API 自动执行管理 Amazon RDS 中的数据库实例和其他对象的任务。

- 有关 API 操作的字母顺序列表，请参阅操作。
- 有关数据类型的字母顺序列表，请参阅数据类型。
- 有关常用查询参数的列表，请参阅常用参数。
- 有关错误代码的描述，请参阅常见错误。

有关 Amazon CLI 的更多信息，请参阅适用于 Amazon RDS 的 Amazon Command Line Interface 参考。

主题

- 使用查询 API
- 对 Amazon RDS 上的应用程序进行故障排除

使用查询 API

以下各节简要讨论用于查询 API 的参数和请求身份验证。

有关查询 API 工作原理的一般信息，请参阅 Amazon EC2 API Reference 中的 查询请求。

查询参数

HTTP 基于查询的请求是指使用 HTTP 动作 GET 或 POST 的 HTTP 请求，查询参数的名称为 `Action`。

每个查询请求必须包括一些通用参数，以处理操作的身份验证和选择事宜。

有些操作会使用参数列表。这些列表都是使用 `param.`*n* 表示法指定的。*n* 值是从 1 开始的整数。

有关 Amazon RDS 区域和端点的信息，请转至《Amazon Web Services 一般参考》的“区域和端点”部分中的 Amazon Relational Database Service（RDS）。

查询请求身份验证

您只能通过 HTTPS 发送查询请求，并且每个查询请求中必须包含签名。您必须使用 Amazon 签名版本 4 或者签名版本 2。有关更多信息，请参阅签名版本 4 签名流程和签名版本 2 签名流程。

对 Amazon RDS 上的应用程序进行故障排除

Amazon RDS 提供具体的描述性错误，以帮助您在与 Amazon RDS API 互动时排查问题。

主题

- 检索错误
- 故障排除技巧

有关对 Amazon RDS 数据库实例进行故障排除的信息，请参阅 Amazon RDS 故障排除。

检索错误

通常，在您花费任何时间处理错误结果之前，您都会希望您的应用程序检查某个请求是否生成错误。查明是否出现错误的最简单方法是寻找 Amazon RDS API 中做出响应的 `Error` 节点。

XPath 语法提供了一种搜索 `Error` 节点存在的简单方法。它还提供了一种相对简单的方法来检索错误代码和消息。下面的代码片段采用 Perl 和 XML::XPath 模块来确定在请求期间是否出现错误。如果出现了错误，那么代码会刊载第一个错误代码和响应信息。

```
use XML::XPath;
    my $xp = XML::XPath->new(xml =>$response);
    if ( $xp->find("//Error") )
    {print "There was an error processing your request:\n", " Error code: ",
    $xp->findvalue("//Error[1]/Code"), "\n", " ",
    $xp->findvalue("//Error[1]/Message"), "\n\n"; }
```

故障排除技巧

我们建议采用下列流程来诊断和解决 Amazon RDS API 的问题：

- 通过检查 http://status.aws.amazon.com 来验证 Amazon RDS 在您的目标 Amazon 区域中正常运行。
- 检查您的请求结构。

每个 Amazon RDS 操作都在 Amazon RDS API Reference 中有一个参考页面。复查您正在使用的参数是否正确。要了解关于潜在错误内容的意见，请查看示例请求或用户场景，以查看这些示例是否执行类似操作。

- 检查 Amazon re:Post。

  Amazon RDS 有一个开发社区，您可以在其中搜索他人在开发过程中遇到的问题以及解决方案。如要查看主题，请转至 Amazon re:Post。

文档历史记录

当前 API 版本：2014-10-31

下表描述了在 2018 年 5 月后每次发布 Amazon RDS 用户指南时进行的重要更改。如需对此文档更新的通知，您可以订阅 RSS 源。

> **Note**
>
> 您可以在数据库的新增功能？页面上筛选新的 Amazon RDS 特征。对于 Products (产品)，请选择 Amazon RDS。然后使用 **RDS Proxy** 或 **Oracle 2023** 之类的关键词进行搜索。

变更	说明	日期
Amazon RDS 在数据库预览环境中支持 MySQL 9.3	MySQL 9.3 现已在美国东部（俄亥俄州）Amazon Web Services 区域的数据库预览环境中提供。有关更多信息，请参阅数据库预览环境中的 MySQL 版本 9.3。	2025 年 6 月 16 日
Amazon RDS for Db2 支持备用模式下的跨区域副本	现在，您可以在备用模式下创建 RDS for Db2 跨区域副本以进行灾难恢复。有关更多信息，请参阅 Working with replicas for RDS for Db2。	2025 年 6 月 11 日
Amazon RDS 支持 MariaDB 10.11.13 和 11.4.7	您现在可以创建运行 MariaDB 版本 10.11.13 和 11.4.7 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MariaDB 版本。	2025 年 6 月 4 日
性能详情的生命周期终止信息	Amazon 已宣布性能详情的生命周期终止日期为：2025 年 11 月 30 日。在此日期之后，	2025 年 5 月 30 日

	Amazon RDS 将不再支持性能详情控制台体验、灵活的保留期（1-24 个月）及其关联定价。我们建议在此日期之前将数据库实例升级到数据库洞察。有关更多信息，请参阅在 Amazon RDS 上使用性能详情监控数据库负载。	
适用于 RDS for MySQL 的 Amazon RDS 扩展支持版本 5.7.44-RDS.20250508	适用于 RDS for MySQL 的 RDS 扩展支持版本 5.7.44-RDS.20250508 现已发布。有关更多信息，请参阅适用于 RDS for MySQL 的 Amazon RDS 扩展支持版本。	2025 年 5 月 27 日
RDS Custom for Oracle 支持 db.m7i 和 db.r7i 实例类	现在可以将 db.m7i 和 db.r7i 实例类用于 RDS Custom for Oracle。有关更多信息，请参阅 RDS Custom for Oracle 的数据库实例类支持。	2025 年 5 月 22 日
查看开源引擎主要版本的支持日期	现在，您可以使用 Amazon CLI 和 RDS API，查看有关 RDS 标准支持和 RDS 扩展支持中开源引擎主要版本的开始日期和结束日期的信息。有关更多信息，请参阅查看 Amazon RDS 扩展支持中引擎版本的支持日期。	2025 年 5 月 20 日
Amazon RDS 支持 MariaDB 10.5.29 和 10.6.22	您现在可以创建运行 MariaDB 版本 10.5.29 和 10.6.22 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MariaDB 版本。	2025 年 5 月 20 日

Amazon RDS 支持 MySQL 8.0.42	您现在可以创建运行 MySQL 8.0.42 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2024 年 5 月 12 日
Amazon RDS 支持 MySQL 8.4.5	您现在可以创建运行 MySQL 8.4.5 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2025 年 5 月 1 日
Amazon RDS 支持亚太地区（香港）、亚太地区（墨尔本）、亚太地区（马来西亚）、加拿大西部（卡尔加里）和欧洲（苏黎世）的备份复制	现在可对亚太地区（香港）、亚太地区（墨尔本）、亚太地区（马来西亚）、加拿大西部（卡尔加里）和欧洲（苏黎世）的数据库进行备份复制。有关可用区域的更多信息，请参阅将自动备份复制到其他 Amazon Web Services 区域。	2025 年 4 月 30 日
Amazon RDS 在 Oracle 多租户架构中支持托管式主用户密码	您可以使用 Amazon Secrets Manager 来管理 RDS for Oracle 容器数据库（CDB）和可插拔数据库（PDB）的主用户密码。CDB 架构的单租户配置和多租户配置均支持托管式密码。有关更多信息，请参阅 CDB 中的用户账户和权限。	2025 年 4 月 24 日

Amazon RDS for Oracle 对于 BYOL 支持 m6id 和 r6id 实例类	现在，可以在 BYOL 模型上使用 m6id 和 r6id 实例类创建 RDS for Oracle 数据库实例。同时支持 Enterprise 和 Standard Edition 2。有关更多信息，请参阅 RDS for Oracle 中支持的版本、实例类和许可组合。	2025 年 4 月 9 日
Amazon RDS 在数据库预览环境中支持 MariaDB 11.8	MariaDB 11.8 现已在美国东部（俄亥俄州）Amazon Web Services 区域的数据库预览环境中推出。有关更多信息，请参阅数据库预览环境中的 MariaDB 版本 11.8。	2025 年 4 月 2 日
IAM 服务相关角色权限更新	`AmazonRDSCustomServiceRolePolicy` 策略现在授予了管理 EC2 密钥对和与 Amazon SQS 集成的额外权限。有关更多信息，请参阅 Amazon RDS 对 Amazon 托管策略的更新。	2025 年 3 月 25 日
Amazon RDS 在数据库预览环境中支持 MySQL 9.2	MySQL 9.2 现已在美国东部（俄亥俄州）Amazon Web Services 区域的数据库预览环境中提供。有关更多信息，请参阅数据库预览环境中的 MySQL 版本 9.2。	2025 年 3 月 24 日

更新现有策略	Amazon RDS 向托管式策略 `AmazonRDSCustomInstanceProfileRolePolicy` 添加了新的权限，以支持在 RDS Custom 实例上使用 RDS Custom 托管式密钥。有关更多信息，请参阅Amazon 托管策略：AmazonRDSCustomInstanceProfileRolePolicy。	2025 年 3 月 20 日
适用于 RDS for MySQL 的 Amazon RDS 扩展支持版本 5.7.44-RDS.20250213	适用于 RDS for MySQL 的 RDS 扩展支持版本 5.7.44-RDS.20250213 现已发布。有关更多信息，请参阅适用于 RDS for MySQL 的 Amazon RDS 扩展支持版本。	2025 年 3 月 12 日
IAM 服务相关角色权限更新	`AmazonRDSCustomServiceRolePolicy` 策略现在授予额外的权限来列出和还原 Secrets Manager 密钥。有关更多信息，请参阅 Amazon RDS 对 Amazon 托管策略的更新。	2025 年 3 月 6 日
Amazon RDS for Db2 支持 db.m7i 和 db.r7i 实例类	现在可以将 db.m7i 和 db.r7i 实例类用于 RDS for Db2。有关更多信息，请参阅数据库实例类类型。	2025 年 2 月 27 日

Amazon RDS for Oracle 支持 Spatial 补丁包（SPB）	RDS for Oracle 现已发布适用于 Oracle Database 19c 的 Spatial 补丁包（SPB）。SPB 引擎版本包含版本更新（RU）补丁以及特定于 Oracle Spatial 的补丁。通常，SPB 在其相应的 RU 发布后的 2-3 周内发布。您可以将数据库实例从标准 RU 升级到相同或更高数据库引擎版本上的 SPB。有关更多信息，请参阅 Oracle 次要版本升级。	2025 年 2 月 25 日
Amazon RDS 支持 MariaDB 11.4.5、10.11.11、10.6.21 和 10.5.28	您现在可以创建运行 MariaDB 版本 11.4.5、10.11.11、10.6.21 和 10.5.28 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MariaDB 版本。	2025 年 2 月 24 日
Amazon RDS 支持 CloudWatch 数据库洞察	现在，可以使用数据库洞察来监控 Amazon RDS 数据库。有关更多信息，请参阅 Monitoring Amazon RDS databases with CloudWatch Database Insights。	2025 年 2 月 20 日
Amazon RDS 支持 MySQL 8.4.4 和 MySQL 8.0.41	现在可以创建运行 MySQL 8.4.4 和 8.0.41 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2025 年 2 月 19 日

适用于 RDS for MySQL 的 Amazon RDS 扩展支持版本 5.7.44-RDS.20250103	适用于 RDS for MySQL 的 RDS 扩展支持版本 5.7.44-RDS.20250103 现已推出。有关更多信息，请参阅适用于 RDS for MySQL 的 Amazon RDS 扩展支持版本。	2025 年 2 月 13 日
Amazon RDS 在数据库预览环境中支持 MariaDB 11.7	MariaDB 11.7 现已在美国东部（俄亥俄州）Amazon Web Services 区域的数据库预览环境中提供。有关更多信息，请参阅数据库预览环境中的 MariaDB 版本 11.7。	2025 年 1 月 9 日
Amazon RDS 支持 MariaDB 11.4.4、10.11.10、10.6.20 和 10.5.27	您现在可以创建运行 MariaDB 版本 11.4.4、10.11.10、10.6.20 和 10.5.27 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MariaDB 版本。	2024 年 12 月 20 日
Amazon RDS for Db2 支持多个数据库	最多可以向一个 RDS for Db2 数据库实例添加 50 个 Db2 数据库。有关更多信息，请参阅 Amazon RDS for Db2 数据库实例上的多个数据库。	2024 年 12 月 20 日
Amazon RDS 在数据库预览环境中支持 MySQL 9.1	MySQL 9.1 现已在美国东部（俄亥俄州）Amazon Web Services 区域的数据库预览环境中提供。有关更多信息，请参阅数据库预览环境中的 MySQL 版本 9.1。	2024 年 12 月 19 日

Amazon RDS 支持 db.m8g 和 db.r8g 实例类	您现在可以将 db.m8g 和 db.r8g 实例类用于 RDS for MySQL、RDS for PostgreSQL 和 RDS for MariaDB。有关更多信息，请参阅数据库实例类类型。	2024 年 11 月 21 日
Amazon RDS 支持 MySQL 8.4	您现在可以创建运行 MySQL 版本 8.4 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2024 年 11 月 21 日
Amazon Relational Database Service 支持自动迁移 EC2 数据库	您可以使用 RDS 控制台将 EC2 数据库迁移到 Amazon RDS。Amazon RDS 使用 Amazon Database Migration Service（Amazon DMS）迁移您的源 EC2 数据库。Amazon DMS 允许您将关系数据库迁移到 Amazon Web Services 云中。有关更多信息，请参阅使用 Amazon Database Migration Service 将 EC2 数据库自动迁移至 Amazon Relational Database Service。	2024 年 11 月 20 日
Amazon RDS 蓝/绿部署支持扩展存储	您现在可以使用 Amazon RDS 蓝/绿部署来调整绿色环境中的存储设置，以优化资源分配。有关更多信息，请参阅创建蓝/绿部署时指定更改。	2024 年 11 月 20 日

Amazon RDS 蓝/绿部署增加了存储初始化	Amazon RDS 现在支持在蓝/绿部署的绿色环境中进行存储初始化，从而在不影响可用性的情况下从首次使用起即可提高卷性能。有关更多信息，请参阅蓝/绿部署的延迟加载和存储初始化。	2024 年 11 月 20 日
Amazon RDS for Oracle 支持用于 BYOL 的 db.m7i 和 db.r7i 实例类	您可以在 BYOL 许可模式下将 db.m7i 和 db.r7i 实例类用于 RDS for Oracle。支持所有版本的 Oracle 数据库。有关更多信息，请参阅数据库实例类类型和支持的 RDS for Oracle 数据库实例类。	2024 年 11 月 18 日
Amazon RDS 代理支持 RDS for PostgreSQL 17	现在，您可以使用 RDS 代理为 RDS for PostgreSQL 17 数据库实例创建代理。有关更多信息，请参阅使用 Amazon RDS Proxy。	2024 年 11 月 15 日
Amazon RDS 支持 MySQL 8.0.40	您现在可以创建运行 MySQL 版本 8.0.40 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2024 年 11 月 13 日
Amazon RDS 支持非洲（开普敦）和亚太地区（海得拉巴）的备份复制	备份复制现在可用于非洲（开普敦）和亚太地区（海得拉巴）区域的数据库。有关可用区域的更多信息，请参阅将自动备份复制到其他 Amazon Web Services 区域。	2024 年 11 月 1 日

Amazon RDS 支持 db.m7i 和 db.r7i 实例类	您现在可以将 db.m7i 和 db.r7i 实例类用于 RDS for MySQL、RDS for PostgreSQL 和 RDS for MariaDB。有关更多信息，请参阅数据库实例类类型。	2024 年 10 月 29 日
Amazon RDS for Oracle 支持 Oracle APEX 版本 24.1.v1	您可以将 Oracle APEX 24.1.v1 与 Oracle Database 19c 及更高版本一起使用。有关更多信息，请参阅 Oracle Application Express。	2024 年 10 月 22 日
多可用区数据库集群支持 IAM 数据库身份验证	您现在可以使用 Amazon Identity and Access Management（IAM）数据库身份验证对多可用区数据库集群进行身份验证。有关更多信息，请参阅创建多可用区数据库集群的设置。	2024 年 10 月 17 日
Amazon RDS 支持 MariaDB 11.4	您现在可以创建运行 MariaDB 版本 11.4 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MariaDB 版本。	2024 年 10 月 15 日
Amazon RDS 支持 Console-to-Code	现在，可以使用 Console-to-Code 从您在 RDS 控制台中执行的操作生成代码。生成的代码有助于您编写代码来自动使用其它 Amazon 服务。有关更多信息，请参阅 Use Console-to-Code to generate code for your Amazon RDS console actions。	2024 年 10 月 3 日

Amazon RDS for Oracle 支持 Oracle Management Agent 13.5.0.0.v2	RDS 支持 Oracle Management Agent 13.5.0.0.v2，后者需要 Oracle Management Service（OMS）13.5.0.23。有关更多信息，请参阅 Management Agent 的要求。	2024 年 9 月 25 日
与 Amazon Redshift 的零 ETL 集成正式发布	零 ETL 集成使事务数据在写入 RDS for MySQL 数据库实例后的几秒钟内，即可在 Amazon Redshift 中使用。该功能现已正式发布。有关更多信息，请参阅使用 Amazon RDS 与 Amazon Redshift 的零 ETL 集成。	2024 年 9 月 12 日
Amazon RDS 支持 MariaDB 10.11.9、10.6.19 和 10.5.26	您现在可以创建运行 MariaDB 版本 10.11.9、10.6.19 和 10.5.26 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Management Agent 的要求。	2024 年 9 月 4 日
Amazon RDS for Oracle 对于 CDB 架构支持 OEM、OEMAGENT 和 OLS 选项	现在，您可以将 Oracle Enterprise Manager 和 Oracle Label Security 与 RDS for Oracle CDB 实例结合使用。有关更多信息，请参阅 Oracle Enterprise Manager 和 Oracle Label Security。	2024 年 9 月 4 日

RDS Custom for SQL Server 在其他区域中推出	RDS Custom for SQL Server 区域现已在美国西部（北加利福尼亚）区域、亚太地区（大阪）区域和欧洲地区（巴黎）区域推出。有关更多信息，请参阅支持 RDS Custom 的区域和数据库引擎。	2024 年 8 月 29 日
适用于 RDS for MySQL 的 Amazon RDS 扩展支持版本 5.7.44-RDS.20240808	适用于 RDS for MySQL 的 RDS 扩展支持版本 5.7.44-RDS.20240808 现已推出。有关更多信息，请参阅适用于 RDS for MySQL 的 Amazon RDS 扩展支持版本。	2024 年 8 月 29 日
Amazon RDS 在亚太地区（马来西亚）区域中推出	Amazon RDS 现已在亚太地区（马来西亚）区域中推出。有关更多信息，请参阅区域和可用区。	2024 年 8 月 22 日
Amazon RDS 支持 MySQL 8.0.39	您现在可以创建运行 MySQL 版本 8.0.39 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2024 年 8 月 12 日
更新现有策略	Amazon RDS 从 `AWSServiceRoleForRDSPreview` 服务相关角色的 `AmazonRDSPreviewServiceRolePolicy` 中删除了 `sns:Publish` 权限。有关更多信息，请参阅 Amazon RDS 对 Amazon 托管策略的更新。	2024 年 8 月 7 日

更新现有策略	Amazon RDS 从 `AWSServiceRoleForRDSBeta` 服务相关角色的 `AmazonRDSBetaServiceRolePolicy` 中删除了 `sns:Publish` 权限。有关更多信息，请参阅 Amazon RDS 对 Amazon 托管策略的更新。	2024 年 8 月 7 日
Amazon RDS 在数据库预览环境中支持 MySQL 8.4	MySQL 8.4 现已在美国东部（俄亥俄州）Amazon Web Services 区域的数据库预览环境中提供。有关更多信息，请参阅数据库预览环境中的 MySQL 版本 8.4。	2024 年 8 月 1 日
IAM 服务相关角色权限更新	`AmazonRDSCustomServiceRolePolicy` 策略现在授予与其他 Amazon Web Services 区域中的 Amazon RDS 服务进行通信并复制 EC2 映像的附加权限。有关更多信息，请参阅 Amazon RDS 对 Amazon 托管策略的更新。	2024 年 7 月 18 日
Amazon RDS 在数据库预览环境中支持 MariaDB 11.4	MariaDB 11.4 现已在美国东部（俄亥俄州）Amazon Web Services 区域的数据库预览环境中提供。有关更多信息，请参阅数据库预览环境中的 MariaDB 版本 11.4。	2024 年 7 月 18 日

适用于 MySQL 的 Amazon ODBC 驱动程序已正式发布	适用于 MySQL 的 Amazon Web Services（Amazon）ODBC 驱动程序是专为实现 RDS for MySQL 的高可用性而设计的客户端驱动程序。有关更多信息，请参阅使用适用于 MySQL 的 Amazon Web Services（Amazon）ODBC 驱动程序连接到 RDS for MySQL。	2024 年 7 月 18 日
更新现有策略	Amazon RDS 从 `AWSServiceRoleForRDS` 服务相关角色的 `AmazonRDSServiceRolePolicy` 中删除了 `sns:Publish` 权限。有关更多信息，请参阅 Amazon 托管式策略：AmazonRDSServiceRolePolicy。	2024 年 7 月 2 日
Db2 的 Amazon Web Services Marketplace 专属优惠	Amazon Web Services Marketplace 现在支持通过 Amazon Web Services Marketplace 为 Amazon RDS for Db2 提供 Db2 许可证的专属优惠。有关更多信息，请参阅获取专属优惠。	2024 年 7 月 1 日
将多可用区数据库集群快照数据导出到 Amazon S3	您现在可以将多可用区数据库集群快照数据导出到 Amazon S3。有关更多信息，请参阅将数据库快照数据导出到 Amazon S3。	2024 年 6 月 27 日

Amazon RDS for Oracle 支持预调配的 r6i 内存优化型实例类	db.r6i Oracle 数据库实例类已针对每个 vCPU 需要更多内存、存储和输入/输出的工作负载进行了优化。例如，db.r6i.8xlarge.tpc2.mem4x 已启用多线程技术，并且提供的内存是 db.r6i.8xlarge 的 4 倍。有关更多信息，请参阅 RDS for Oracle 实例类。	2024 年 6 月 21 日
适用于 RDS for MySQL 的 Amazon RDS 扩展支持版本 5.7.44-RDS.20240529	适用于 RDS for MySQL 的 RDS 扩展支持版本 5.7.44-RDS.20240529 现已推出。有关更多信息，请参阅适用于 RDS for MySQL 的 Amazon RDS 扩展支持版本。	2024 年 6 月 20 日
Amazon RDS 支持 MySQL 8.0.37	您现在可以创建运行 MySQL 版本 8.0.37 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2024 年 6 月 18 日
Amazon RDS 支持 MariaDB 10.11.8、10.6.18、10.5.25 和 10.4.34	您现在可以创建运行 MariaDB 版本 10.11.8、10.6.18、10.5.25 和 10.4.34 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MariaDB 版本。	2024 年 6 月 14 日

Amazon RDS 即将结束对 db.m4、db.r4 和 db.t2 数据库实例类的支持	对于 RDS for MariaDB、RDS for MySQL 和 RDS for PostgreSQL 数据库引擎，您无法再创建使用 db.m4、db.r4 和 db.t2 实例类的数据库实例。RDS 会自动将使用这些类的现有数据库实例升级到较新一代实例。有关更多信息，请参阅数据库实例类类型。	2024 年 6 月 4 日
多可用区数据库集群在更多 Amazon Web Services 区域中提供	您可以在更多 Amazon Web Services 区域中创建多可用区数据库集群。有关显示所有受支持区域的表，请参阅支持 Amazon RDS 中多可用区数据库集群的区域和数据库引擎。	2024 年 5 月 29 日
Amazon Python 驱动程序现已正式发布	Amazon Web Services（Amazon）驱动程序设计为高级 Python 包装器。这款包装器是对开源 Psycopg 驱动程序的各项功能的补充和扩展。有关更多信息，请参阅借助 Amazon 驱动程序连接到数据库实例。	2024 年 5 月 23 日
RDS 代理现已在更多区域推出	RDS 代理现已在亚太地区（海得拉巴）、亚太地区（墨尔本）、中东（阿联酋）、以色列（特拉维夫）、加拿大西部（卡尔加里）和欧洲（苏黎世）区域推出。有关 RDS 代理的更多信息，请参阅使用 Amazon RDS 代理。	2024 年 5 月 21 日

通过 Amazon Web Services Marketplace 获得 Db2 许可证	借助"通过 Amazon Web Services Marketplace 获得 Db2 许可证"模式，您现在可以按小时费率为 Amazon RDS for Db2 订阅 Db2 许可证。有关更多信息，请参阅 Amazon RDS for Db2 许可选项。	2024 年 5 月 21 日
Amazon RDS 支持性能详情精细访问	现在，您可以在性能详情中允许或拒绝对各个维度的访问。这种精细访问控制功能可用于 `GetResourceMetrics`、`DescribeDimensionKeys` 和 `GetDimensionKeyDetails` 操作。有关更多信息，请参阅授予 Performance Insights 精细访问权限。	2024 年 5 月 21 日
适用于 RDS for MySQL 的 Amazon RDS 扩展支持版本	您可以查看适用于 RDS for MySQL 版本的所有 RDS Extended Support 发行版。有关更多信息，请参阅适用于 RDS for MySQL 的 Amazon RDS Extended Support 版本。	2024 年 5 月 16 日
Amazon RDS 在数据库预览环境中支持 MySQL 8.3	MySQL 8.3 现已在美国东部（俄亥俄州）Amazon Web Services 区域的数据库预览环境中提供。有关更多信息，请参阅数据库预览环境中的 MySQL 版本 8.3。	2024 年 4 月 30 日

Amazon RDS for Db2 支持时区	RDS for Db2 现在支持为新的 RDS for Db2 数据库实例设置本地时区。有关更多信息，请参阅适用于 Db2 数据库实例的 Amazon RDS 本地时区。	2024 年 4 月 25 日
IAM 服务相关角色权限更新	`AmazonRDSCustomServiceRolePolicy` 策略现在授予了额外权限，用于将服务角色作为实例配置文件关联到 RDS Custom 实例。有关更多信息，请参阅 Amazon RDS 对 Amazon 托管策略的更新。	2024 年 4 月 19 日
Amazon RDS for Oracle 在所有 Amazon Web Services 区域中支持 Oracle Data Guard 切换	现在，您可以在所有支持的区域中使用 Oracle Data Guard 切换。有关更多信息，请参阅 Oracle Data Guard 切换概述。	2024 年 4 月 16 日
RDS Custom for Oracle 支持 Oracle Standard Edition 2	现在，您可以在 Oracle Database 12c 发行版 1（12.1）、12c 发行版 2（12.2）、18c 和 19c 上使用 Standard Edition 2 创建数据库实例。您可以同时创建 CDB 和非 CDB。有关更多信息，请参阅 RDS Custom for Oracle 的版本和许可支持。	2024 年 4 月 11 日
Amazon RDS for Oracle 支持 Oracle APEX 版本 23.2.v1	您可以将 Oracle APEX 23.2.v1 与 Oracle Database 19c 及更高版本一起使用。有关更多信息，请参阅 Oracle Application Express。	2024 年 4 月 11 日

更新 RDS Custom 服务相关角色权限	`AmazonRDSCustomServiceRolePolicy` 现在授予额外的权限，以允许 RDS Custom for SQL Server 获取 EC2 实例类型信息并修改数据库主机实例类型。有关更多信息，请参阅 Amazon 托管式策略的更新。	2024 年 4 月 8 日
Amazon RDS Custom for Oracle 支持 db.x2iezn 数据库实例类	您现在可以将 db.x2iezn 实例类用于 RDS Custom for Oracle 数据库实例。有关更多信息，请参阅 RDS Custom for Oracle 的数据库实例类支持。	2024 年 3 月 26 日
Amazon RDS 对于多可用区数据库集群支持 db.c6gd 实例类	现在，您可以将 db.c6gd 实例类用于多可用区数据库集群部署。有关更多信息，请参阅 Instance class availability for Multi-AZ DB clusters。	2024 年 3 月 21 日
Amazon RDS 扩展支持	现在，创建或还原 RDS for MySQL 5.7 或 RDS for PostgreSQL 11 数据库会自动将该数据库注册到 Amazon RDS 扩展支持，这样您的现有应用程序就可以继续按原样运行。您可以选择退出 RDS Extended Support，以避免在数据库引擎的 RDS 标准支持终止日期后支付费用。有关更多信息，请参阅使用 Amazon RDS 扩展支持。	2024 年 3 月 21 日

RDS for Db2 与 Amazon License Manager 集成	RDS for Db2 现已与 Amazon License Manager 集成。如果您使用“自带许可”模型，则 Amazon License Manager 集成有助于监控组织内的 Db2 许可证使用情况。有关更多信息，请参阅与 Amazon License Manager 集成。	2024 年 3 月 20 日
多可用区数据库集群的 CA 证书轮换	您现在可以轮换多可用区数据库集群的 CA 证书。考虑使用新的 CA 证书之一：rds-ca-rsa2048-g1、rds-ca-rsa4096-g1 或 rds-ca-ecc384-g1。有关更多信息，请参阅 Rotating your SSL/TLS certificate。	2024 年 3 月 6 日
Amazon RDS 支持 io2 Block Express 存储	您现在可以创建使用 io2 Block Express 存储类型的 RDS 数据库实例。有关更多信息，请参阅 io2 Block Express storage。	2024 年 3 月 6 日
RDS Custom for SQL Server 支持 db.r5b 和 db.x2iedn 数据库实例类	您现在可以将 db.r5b 和 db.x2iedn 实例类用于 RDS Custom for SQL Server 数据库实例。有关更多信息，请参阅 DB instance class support for RDS Custom for SQL Server。	2024 年 3 月 4 日

RDS Custom for Oracle 在中东（阿联酋）区域中推出	您可以在中东（阿联酋）区域中创建 RDS Custom for Oracle 数据库实例。要查看显示所有受支持 Amazon Web Services 区域的表，请参阅支持 RDS Custom for Oracle 的区域和数据库引擎。	2024 年 3 月 4 日
新 Amazon 托管策略	Amazon RDS 添加了一个名为 `AmazonRDSCustomInstanceProfileRolePolicy` 的新托管式策略，允许 RDS Custom 通过 EC2 实例配置文件执行自动化操作和数据库管理任务。有关更多信息，请参阅 Amazon RDS 对 Amazon 托管策略的更新。	2024 年 2 月 27 日
Amazon RDS 支持 MariaDB 10.11.7、10.6.17、10.5.24 和 10.4.33	您现在可以创建运行 MariaDB 版本 10.11.7、10.6.17、10.5.24 和 10.4.33 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MariaDB 版本。	2024 年 2 月 26 日
Amazon RDS Multi-AZ 数据库集群支持 Amazon EBS gp3 存储卷	多可用区数据库集群现在支持基于 gp3 SSD 的 EBS 卷。有关更多信息，请参阅 gp3 存储。	2024 年 2 月 26 日
Amazon RDS 在以色列（特拉维夫）区域支持 Amazon Secrets Manager	Amazon RDS 在以色列（特拉维夫）区域支持 Secrets Manager。有关更多信息，请参阅使用 Amazon RDS 和 Amazon Secrets Manager 管理密码。	2024 年 2 月 21 日

Amazon RDS for Db2 支持审核日志记录	RDS for Db2 现在支持数据库级别的审核日志记录。当您为 RDS for Db2 数据库启用审核日志记录时，Amazon RDS 会记录数据库活动并将审核日志存储在 Amazon S3 中。有关更多信息，请参阅 Db2 audit logging。	2024 年 2 月 15 日
Amazon RDS 扩展支持	当您的数据库实例和多可用区数据库集群中的 RDS for MySQL 和 RDS for PostgreSQL 主要引擎版本到达 RDS 标准支持终止日期时，Amazon RDS 将自动启用 Amazon RDS 扩展支持。有关更多信息，请参阅使用 Amazon RDS 扩展支持。	2024 年 2 月 15 日
Amazon RDS 支持 MySQL 8.0.36	您现在可以创建运行 MySQL 版本 8.0.36 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2024 年 2 月 12 日
Amazon RDS 对于 RDS for Db2 支持 EBCDIC 排序规则	现在，您可以创建使用 EBCDIC 排序规则序列对数据库中的内容进行排序的 Db2 数据库。有关更多信息，请参阅 EBCDIC collation for Db2 databases on Amazon RDS。	2024 年 1 月 29 日
更新为默认 CA 证书	默认 CA 证书设置为 `rds-ca-rsa2048-g1` 。有关更多信息，请参阅 Using SSL/TLS to encrypt a connection to a DB instance。	2024 年 1 月 26 日

Amazon RDS for PostgreSQL 支持两个适用于 pl/Rust 的新 crate，即 croaring-rs 和 num-bigint	您可以在 Amazon RDS for PostgreSQL 中使用这两个新的 crate。有关更多信息，请参阅将 crate 与 PL/Rust 结合使用。	2024 年 1 月 24 日
Amazon RDS for PostgreSQL 支持 TLS 版本 1.3	您可以在 RDS for PostgreSQL 中使用传输层安全性协议（TLS）版本 1.3。有关更多信息，请参阅将 SSL 用于 PostgreSQL 数据库实例。	2024 年 1 月 24 日
RDS Custom for SQL Server 支持 Microsoft SQL Server 2022	您现在可以创建使用 SQL Server 2022 的 RDS Custom for SQL Server 数据库实例。有关更多信息，请参阅使用 RDS Custom for SQL Server。	2024 年 1 月 22 日
更新 Amazon 托管策略权限	`AWSServiceRoleForRDS` 服务相关角色的 `AmazonRDSServiceRolePolicy` 具有新的语句 ID。有关更多信息，请参阅 Amazon RDS 对 Amazon 托管策略的更新。	2024 年 1 月 19 日
RDS Custom for Oracle 支持欧洲地区（巴黎）区域	您可以在欧洲地区（巴黎）区域中创建 RDS Custom for Oracle 数据库实例。有关更多信息，请参阅支持 RDS Custom for Oracle 的区域和数据库引擎。	2024 年 1 月 18 日

Amazon RDS for MySQL 支持多源复制	您现在可以在 RDS for MySQL 数据库实例上使用多源复制。有关更多信息，请参阅 Configuring multi-source replication on RDS for MySQL。	2024 年 1 月 16 日
Amazon RDS 在数据库预览环境中支持 MySQL 8.2	MySQL 8.2 现已在美国东部（俄亥俄州）Amazon Web Services 区域的数据库预览环境中提供。有关更多信息，请参阅数据库预览环境中的 MySQL 版本 8.2。	2024 年 1 月 11 日
RDS 代理在欧洲（西班牙）区域推出	RDS 代理现已在欧洲（西班牙）区域推出。有关 RDS 代理的更多信息，请参阅使用 Amazon RDS 代理。	2024 年 1 月 8 日
Amazon RDS 在加拿大西部（卡尔加里）区域推出	Amazon RDS 现已在加拿大西部（卡尔加里）区域推出。有关更多信息，请参阅区域和可用区。	2023 年 12 月 20 日
Amazon RDS for Db2 支持 5000 个本地用户	现在，您最多可以将 5000 个本地用户添加到授权列表中。有关更多信息，请参阅 rdsadmin.add_user。	2023 年 12 月 20 日
Amazon RDS 支持查看和响应建议	Amazon RDS 建议现在包括适用于 RDS for PostgreSQL 的基于阈值的主动建议和基于机器学习的被动建议。有关更多信息，请参阅来自 Amazon RDS 的建议。	2023 年 12 月 19 日

Amazon RDS 支持 MariaDB 10.11.6、10.6.16、10.5.23 和 10.4.32	您现在可以创建运行 MariaDB 版本 10.11.6、10.6.16、10.5.23 和 10.4.32 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MariaDB 版本。	2023 年 12 月 12 日
Amazon RDS 推出与 Amazon Redshift 的零 ETL 集成（预览版）	零 ETL 集成提供了一个完全托管式解决方案，使事务数据在写入 RDS for MySQL 数据库实例后的几秒钟内即可在 Amazon Redshift 中可用。有关更多信息，请参阅使用 Amazon RDS 与 Amazon Redshift 的零 ETL 集成（预览版）。	2023 年 11 月 28 日
Amazon RDS 支持 IBM Db2 数据库引擎	现在，您可以在 Amazon RDS 中运行 IBM Db2 数据库引擎。有关更多信息，请参阅 Amazon RDS for Db2。	2023 年 11 月 27 日
RDS for PostgreSQL 支持到 PostgreSQL 16.1 的主要版本升级以及到 15.5、14.10、13.13、12.17 和 11.22 的次要版本升级	使用 RDS for PostgreSQL，您现在可以将数据库引擎升级到主要版本 16.1，并将次要版本升级到 15.5、14.10、13.13、12.17 和 11.22。有关更多信息，请参阅升级 Amazon RDS 的 PostgreSQL 数据库引擎。	2023 年 11 月 17 日
RDS Custom for Oracle 支持选项组	您可以创建或修改选项组并将其与 RDS Custom for Oracle 数据库实例关联。现在支持 `Timezone`。有关更多信息，请参阅使用 RDS Custom for Oracle 中的选项组。	2023 年 11 月 17 日

Amazon RDS for MySQL 支持组复制插件	现在，您可以使用由 MySQL 社区开发和维护的组复制插件，通过 RDS for MySQL 版本 8.0.35 或更高版本的数据库实例来设置主动-主动集群。有关更多信息，请参阅 Configuring active-active clusters for RDS for MySQL。	2023 年 11 月 17 日
Amazon RDS 代理支持 RDS for PostgreSQL 16.1	现在，您可以使用 RDS 代理为 RDS for PostgreSQL 16.1 数据库实例创建代理。有关更多信息，请参阅使用 Amazon RDS Proxy。	2023 年 11 月 17 日
RDS Custom for SQL Server 支持 Microsoft SQL Server 2019 Developer 版本	您可以创建使用 SQL Server 2019 Developer 版本的 RDS Custom for SQL Server 数据库实例。有关更多信息，请参阅对于 RDS Custom for SQL Server 使用自带媒体。	2023 年 11 月 16 日
多可用区数据库集群的次要版本升级，停机时间最短	当您对多可用区数据库集群执行次要版本升级时，Amazon RDS 现在会在写入器实例之前升级读取器数据库实例，从而大大减少停机时间。您可以使用 RDS 代理将停机时间进一步缩短到一秒或更短。有关更多信息，请参阅升级多可用区数据库集群的引擎版本。	2023 年 11 月 16 日
RDS for SQL Server 支持 Microsoft SQL Server 2022	您现在可以创建使用 SQL Server 2022 的 RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 Microsoft SQL Server 版本。	2023 年 11 月 15 日

RDS for MySQL 支持将快照从版本 5.7 升级到 8.0	您现在可以将 RDS for MySQL 快照的引擎版本从版本 5.7 升级到版本 8.0。您可以使用 Amazon Web Services Management Console、RDS API 的 `ModifyDBSnapshot` 操作或 Amazon CLI 达成这一目的。有关更多信息，请参阅升级 MySQL 数据库快照引擎版本。	2023 年 11 月 15 日
RDS Custom for SQL Server 支持 10000 个数据库的时间点恢复	现在，您可以在 RDS Custom for SQL Server 数据库实例上创建多达 1000 个符合完整备份和时间点恢复条件的数据库。有关更多信息，请参阅将 RDS Custom for SQL Server 实例还原到某个时间点。	2023 年 11 月 15 日
RDS Custom for SQL Server 支持使用服务主密钥	RDS Custom for SQL Server 现在支持使用服务主密钥（SMK）。SMK 允许您加密诸如凭证之类的对象和使用 SQL Server 功能，例如 TDE 和列加密。有关更多信息，请参阅在 RDS Custom for SQL Server 中使用服务主密钥。	2023 年 11 月 13 日
Amazon RDS 在数据库预览环境中支持 MySQL 8.1	MySQL 8.1 现已在美国东部（俄亥俄州）Amazon Web Services 区域的数据库预览环境中提供。有关更多信息，请参阅数据库预览环境中的 MySQL 版本 8.1。	2023 年 11 月 10 日

RDS 支持 MySQL 8.0.35 和 5.7.44	您现在可以创建运行 MySQL 版本 8.0.35 和 5.7.44 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2023 年 11 月 9 日
RDS 代理支持多可用区数据库集群	RDS 代理现在支持连接到多可用区数据库集群。有关更多信息，请参阅使用 Amazon RDS 代理端点。	2023 年 11 月 9 日
RDS Custom for Oracle 可在 Amazon GovCloud (US) Regions中使用	Amazon RDS 现在已在 Amazon GovCloud (US) Regions中推出。有关更多信息，请参阅支持 RDS Custom for Oracle 的区域和数据库引擎。	2023 年 11 月 9 日
Amazon RDS 优化写入功能支持 db.m5 数据库实例类	Amazon RDS 优化写入功能现在支持 db.m5 数据库实例类。有关更多信息，请参阅使用适用于 MariaDB 的 Amazon RDS 优化型写入功能提高写入性能和使用适用于 MySQL 的 Amazon RDS 优化型写入功能提高写入性能。	2023 年 11 月 9 日

Amazon RDS for Oracle支持 CDB 架构的多租户配置	借助 RDS for Oracle 多租户功能，RDS 为您的 Oracle 数据库提供了完全托管的 Oracle 多租户架构和体验。您可以使用 RDS API 在 CDB 中创建多个 PDB（称为租户数据库）。RDS 提供 CDB 架构的多租户配置，作为传统单租户配置的替代方案。有关更多信息，请参阅 CDB 架构的多租户配置。	2023 年 11 月 8 日
Amazon RDS 将性能详情指标导出到 Amazon CloudWatch	性能详情允许您将预配置或自定义的指标控制面板导出到 Amazon CloudWatch。导出的指标控制面板可在 CloudWatch 控制台中查看。您还可以导出选定的性能详情指标小组件，并在 CloudWatch 控制台中查看指标数据。有关更多信息，请参阅将 Performance Insights 指标导出到 CloudWatch。	2023 年 11 月 8 日
Amazon RDS Custom for Oracle 允许您升级数据库实例上的操作系统	现在，您可以使用 CLI 命令 `modify-db-instance` 升级 RDS Custom for Oracle 数据库实例的数据库或操作系统（OS）。有关更多信息，请参阅升级 Amazon RDS Custom for Oracle 的数据库实例。	2023 年 11 月 7 日
RDS 代理支持 RDS for PostgreSQL 的扩展协议	现在，您可以在 RDS for PostgreSQL 数据库实例上执行扩展的查询协议。有关更多信息，请参阅使用 Amazon RDS Proxy。	2023 年 11 月 6 日

RDS 蓝绿部署的 RDS for PostgreSQL 支持	您现在可以从 RDS for PostgreSQL 数据库实例上创建蓝绿部署。有关更多信息，请参阅使用 Amazon RDS 蓝绿部署进行数据库更新。	2023 年 10 月 26 日
对 Amazon 托管策略的更新	`AmazonRDSPerformanceInsightsReadOnly` 和 `AmazonRDSPerformanceInsightsFullAccess` 托管策略现在将 `Sid`（语句 ID）作为标识符包括在策略语句中。有关更多信息，请参阅 Amazon RDS 对 Amazon 托管策略的更新。	2023 年 10 月 23 日
RDS Custom for Oracle 支持欧洲地区（米兰）区域	有关更多信息，请参阅支持 RDS Custom for Oracle 的区域和数据库引擎。	2023 年 10 月 23 日
在现有数据库上启用 RDS 优化写入	现在，您可以对现有数据库实例启用 RDS 优化写入，即使该实例是使用不支持该功能的引擎版本、数据库实例类或文件系统配置创建的亦是如此。有关更多信息，请参阅 RDS for MySQL 的在现有数据库上启用 RDS 优化写入，以及 RDS for MariaDB 的在现有数据库上启用 RDS 优化写入。	2023 年 10 月 19 日

Amazon RDS 支持使用专用日志卷（DLV）。	您现在可以将专用日志卷（DLV）与 RDS for MariaDB、RDS for MySQL 以及 RDS for PostgreSQL 一起使用。DLV 非常适合分配了较大存储空间、每秒 I/O（IOPS）要求较高或工作负载对延迟敏感的数据库。有关更多信息，请参阅使用专用日志卷（DLV）。	2023 年 10 月 17 日
Amazon RDS for PostgreSQL、MySQL 和 MariaDB 支持新的数据库实例类	您可以创建运行 PostgreSQL、MySQL 和 MariaDB 的 Amazon RDS 数据库实例，这些数据库实例使用 db.m6.in、db.m6idn、db.r6.in 和 db.r6.idn 数据库实例类。有关更多信息，请参阅所有可用数据库实例类支持的数据库引擎。	2023 年 10 月 12 日
Amazon RDS for PostgreSQL 支持 pgactive	pgactive 扩展已在 Amazon RDS for PostgreSQL 中推出。有关更多信息，请参阅将 PostgreSQL 扩展与 Amazon RDS for PostgreSQL 结合使用。	2023 年 10 月 9 日
RDS Custom for Oracle 已在亚太地区（雅加达）区域推出	您可以在亚太地区（雅加达）区域中创建 RDS Custom for Oracle 数据库实例。有关更多信息，请参阅支持 RDS Custom for Oracle 的区域和数据库引擎。	2023 年 10 月 5 日

RDS Custom for SQL Server 支持新的服务器级排序规则	RDS Custom for SQL Server 现在支持各种服务器排序规则，包括传统编码和 UTF-8 编码，适用于 SQL_Latin、日语、德语和阿拉伯语区域设置。有关更多信息，请参阅 RDS Custom for SQL Server 数据库实例的排序规则和字符支持。	2023 年 9 月 26 日
更新 Amazon 托管策略权限	`AWSServiceRoleForRDSCustom` 服务相关角色的 `AmazonRDSCustomServiceRolePolicy` 拥有的新权限允许 RDS Custom 创建、修改和删除 EventBridge 托管式规则。有关更多信息，请参阅 Amazon RDS 对 Amazon 托管策略的更新。	2023 年 9 月 20 日
Amazon RDS 向 Amazon CloudWatch 发布了性能详情计数器指标	CloudWatch 控制台中的 DB_PERF_INSIGHTS 指标数学函数允许您查询 Amazon RDS 以获取性能详情计数器指标。有关更多信息，请参阅创建 CloudWatch 警报以监控 Amazon RDS。	2023 年 9 月 20 日
性能详情支持 SQL Server 的摘要级统计数据	使用性能详情时，您可以查看 Amazon RDS for SQL Server 的语句和摘要级别的 SQL 统计信息。有关更多信息，请参阅分析正在 SQL Server 中运行的查询。	2023 年 9 月 18 日

Amazon RDS for PostgreSQL、MySQL 和 MariaDB 支持 db.m6.id 和 db.r6.id 数据库实例类类型	现在，您可以创建运行 PostgreSQL、MySQL 和 MariaDB 的 Amazon RDS 数据库实例，这些实例使用内存优化型 db.m6.id 和 db.r6.id 数据库实例类类型。这些类型提供基于 NVMe 的本地 SSD 存储。有关更多信息，请参阅所有可用数据库实例类支持的数据库引擎。	2023 年 9 月 11 日
RDS for PostgreSQL 多可用区数据库集群的主要版本升级支持	现在，您可以执行 RDS for PostgreSQL 多可用区数据库集群的主要版本升级。有关更多信息，请参阅升级多可用区数据库集群的引擎版本。	2023 年 9 月 7 日
Amazon RDS 支持 MariaDB 10.11.5、10.6.15、10.5.22 和 10.4.31	您现在可以创建运行 MariaDB 版本 10.11.5、10.6.15、10.5.22 和 10.4.31 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MariaDB 版本。	2023 年 9 月 7 日
Amazon RDS 扩展支持	Amazon RDS 宣布，在 RDS 标准支持终止日期后，将能够在您的数据库实例中继续运行 RDS for MySQL 和 RDS for PostgreSQL 主要引擎版本。有关更多信息，请参阅使用 Amazon RDS 扩展支持。	2023 年 9 月 1 日

RDS Custom 支持启动和停止 RDS Custom for SQL Server 数据库实例	RDS Custom 现在支持启动和停止 RDS Custom for SQL Server 数据库实例。有关更多信息，请参阅启动和停止 RDS Custom for SQL Server 数据库实例。	2023 年 8 月 31 日
Amazon RDS 优化型写入功能支持 db.r5 数据库实例类	Amazon RDS 优化型写入功能现在支持 db.r5 数据库实例类。有关更多信息，请参阅使用适用于 MariaDB 的 Amazon RDS 优化型写入功能提高写入性能和使用适用于 MySQL 的 Amazon RDS 优化型写入功能提高写入性能。	2023 年 8 月 31 日
Amazon RDS for Oracle 对于 CDB 支持时区文件自动升级	使用 `TIMEZONE_FILE_AUTOUPGRADE` 选项，您可以将 RDS for Oracle 容器数据库（CDB）上的当前时区文件升级到最新版本。有关更多信息，请参阅 Oracle 时区文件自动升级。	2023 年 8 月 29 日
Amazon RDS 优化型写入功能支持 db.m6g 和 db.m6i 数据库实例类	Amazon RDS 优化型写入功能现在支持 db.m6g 和 db.m6i 数据库实例类。有关更多信息，请参阅使用适用于 MariaDB 的 Amazon RDS 优化型写入功能提高写入性能和使用适用于 MySQL 的 Amazon RDS 优化型写入功能提高写入性能。	2023 年 8 月 28 日

Amazon RDS 支持 MariaDB 10.11	您现在可以创建运行 MariaDB 版本 10.11 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MariaDB 版本。	2023 年 8 月 21 日
更新 Amazon 托管策略权限	AWSServiceRoleForRDSCustom 服务相关角色的 AmazonRDSCustomServiceRolePolicy 拥有的新权限允许 RDS Custom 创建网络接口。有关更多信息，请参阅 Amazon RDS 对 Amazon 托管策略的更新。	2023 年 8 月 18 日
更新 Amazon 托管策略权限	AmazonRDSFullAccess 托管策略具有新权限，允许您生成、查看和删除时间段内的性能分析报告。有关更多信息，请参阅 Amazon RDS 对 Amazon 托管策略的更新。	2023 年 8 月 17 日
更新 Amazon 托管策略权限	向 AmazonRDSPerformanceInsightsReadOnly 托管策略添加新权限和添加新的托管策略 AmazonRDSPerformanceInsightsFullAccess ，可让您生成时间段内的数据库负载分析报告。有关更多信息，请参阅 Amazon RDS 对 Amazon 托管策略的更新。	2023 年 8 月 16 日

Amazon RDS 支持时间段内的性能分析	性能详情允许您创建和查看特定时间段的性能分析报告。该报告提供已确定的见解和解决性能问题的建议。有关更多信息，请参阅分析时间段内的数据库负载。	2023 年 8 月 16 日
Amazon RDS Custom for Oracle 支持 db.r5b 和 db.x2iedn 数据库实例类	您现在可以将 db.r5b 和 db.x2iedn 实例类用于 RDS Custom for Oracle 数据库实例。有关更多信息，请参阅 RDS Custom for Oracle 的数据库实例类支持。	2023 年 8 月 16 日
Amazon RDS Custom for Oracle 支持 db.m6i、db.r6i 和 db.t3 数据库实例类	您现在可以将 db.m6i、db .r6i 和 db.t3 实例类用于 RDS Custom for Oracle 数据库实例。有关更多信息，请参阅 RDS Custom for Oracle 的数据库实例类支持。	2023 年 8 月 15 日
在数据库预览环境中，Amazon RDS for PostgreSQL 现在支持 PostgreSQL 版本 16 测试版 3	PostgreSQL 版本 16 测试版 3 现已在美国东部（俄亥俄州）Amazon Web Services 区域的数据库预览环境中提供。有关更多信息，请参阅使用数据库预览环境。	2023 年 8 月 11 日
Amazon RDS 支持 MySQL 8.0.34 和 5.7.43	您现在可以创建运行 MySQL 版本 8.0.34 和 5.7.43 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2023 年 8 月 9 日

RDS for SQL Server 支持备用副本的操作系统指标视图	现在，您可以查看 RDS for SQL Server 的备用副本的操作系统指标。有关更多信息，请参阅在 RDS 控制台中查看操作系统指标。	2023 年 8 月 3 日
RDS for Oracle 对于 CDB 支持 Oracle Data Guard	RDS for Oracle 对于 Oracle Database 19c 和 21c 容器数据库（CDB）支持 Data Guard 只读副本。您可以使用现有 RDS API 在 CDB 中创建、管理和提升只读副本，就像在非 CDB 中一样。有关更多信息，请参阅多租户只读副本。	2023 年 8 月 1 日
Amazon RDS 在以色列（特拉维夫）区域推出	Amazon RDS 现已在以色列（特拉维夫）区域推出。有关更多信息，请参阅区域和可用区。	2023 年 8 月 1 日
Amazon RDS 支持 Oracle APEX 版本 23.1.v1	您可以将 Oracle APEX 23.1.v1 与 Oracle Database 19c 及更高版本一起使用。有关更多信息，请参阅 Oracle Application Express。	2023 年 7 月 26 日
Amazon RDS Custom for Oracle 支持非默认 Oracle SID	当您使用 Oracle Database 19c 创建 RDS Custom for Oracle 数据库实例时，您可以指定非默认 Oracle 系统标识符（Oracle SID)。此值也是 CDB 的名称。有关更多信息，请参阅多租户架构注意事项。	2023 年 7 月 21 日

RDS for SQL Server 支持自托管式 Active Directory	您现在可以使用自托管式 Active Directory 将 RDS for SQL Server 数据库实例直接加入您的 Microsoft Active Directory（AD）域。自托管式 AD 域可以在本地或云中。有关更多信息，请参阅使用自托管式 Active Directory。	2023 年 7 月 7 日
多可用区数据库集群的 PostgreSQL 逻辑复制支持	您现在可以将 PostgreSQL 逻辑复制用于多可用区数据库集群，以复制和同步各个表，而不是整个数据库实例。有关更多信息，请参阅使用多可用区数据库集群为 Amazon RDS 设置 PostgreSQL 逻辑复制。	2023 年 7 月 6 日
在数据库预览环境中，Amazon RDS for PostgreSQL 现在支持 PostgreSQL 版本 16 测试版 2	PostgreSQL 版本 16 测试版 2 现已在美国东部（俄亥俄州）Amazon Web Services 区域的数据库预览环境中提供。有关更多信息，请参阅使用数据库预览环境。	2023 年 7 月 6 日
更新 Amazon 托管策略权限	`AWSServiceRoleForRDSCustom` 服务相关角色的 `AmazonRDSCustomServiceRolePolicy` 拥有的新权限允许 RDS Custom for Oracle 使用快照。有关更多信息，请参阅 Amazon RDS 对 Amazon 托管策略的更新。	2023 年 6 月 23 日

RDS 支持 MariaDB 10.6.14、10.5.21 和 10.4.30	您现在可以创建运行 MariaDB 版本 10.6.14、10.5.21 和 10.4.30 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MariaDB 版本。	2023 年 6 月 22 日
RDS 支持 MySQL 8.0.33 和 5.7.42	您现在可以创建运行 MySQL 版本 8.0.33 和 5.7.42 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2023 年 6 月 15 日
RDS 支持 MariaDB 10.6.13、10.5.20、10.4.29 和 10.3.39	您现在可以创建运行 MariaDB 版本 10.6.13、10.5.20、10.4.29 和 10.3.39 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MariaDB 版本。	2023 年 6 月 15 日
RDS for Oracle 支持可传输的表空间	您可以使用可传输的表空间，将数据从本地 Oracle 数据库迁移到 RDS for Oracle 数据库实例。此技术不需要额外的许可，并且是停机时间最少的迁移技术。有关更多信息，请参阅使用 Oracle 可传输表空间进行迁移。	2023 年 6 月 15 日
Amazon RDS 支持使用 RDS for MariaDB 版本 10.6 的 RDS 代理	现在，您可以使用 RDS for MariaDB 版本 10.6 数据库来创建 RDS 代理。有关 RDS 代理的更多信息，请参阅使用 Amazon RDS 代理。	2023 年 6 月 15 日

RDS Custom for SQL Server 支持自带媒体（BYOM）	现在，您可以使用自己的 SQL Server 媒体创建自定义引擎版本（CEV）。有关更多信息，请参阅对于 RDS Custom for SQL Server 使用自带媒体。	2023 年 6 月 8 日
RDS for Oracle 可以将 Oracle Database 19c 非 CDB 转换为 CDB	如果您的数据库实例运行 Oracle Database 19c（带有 2021 年 4 月或更高版本的 RU），则可以将非 CDB 转换为 CDB（容器数据库）。转换架构后，您可以将 19c CDB 升级到 21c CDB。此步骤是必要的，因为您无法使用单个命令升级数据库和转换架构。有关更多信息，请参阅将 RDS for Oracle 非 CDB 转换为 CDB。	2023 年 5 月 31 日
中国区域推出了多可用区数据库集群	多可用区数据库集群现已在 Amazon Web Services 区域中国（北京）和中国（宁夏）推出。有关更多信息，请参阅支持 Amazon RDS 中多可用区数据库集群的区域和数据库引擎。	2023 年 5 月 30 日
Amazon RDS 优化型读取功能支持多可用区数据库集群	Amazon RDS 优化型读取功能现在支持多可用区数据库集群。有关更多信息，请参阅使用 Amazon RDS 优化型读取功能提高 RDS for MySQL 的查询性能和使用 Amazon RDS 优化型读取功能提高 RDS for PostgreSQL 的查询性能。	2023 年 5 月 30 日

RDS Custom for Oracle 支持亚太地区（雅加达）区域	有关更多信息，请参阅支持 RDS Custom for Oracle 的区域和数据库引擎。	2023 年 5 月 29 日
使用源 RDS for PostgreSQL 多可用区数据库集群来创建数据库实例只读副本	您现在可以使用 RDS for PostgreSQL 多可用区数据库集群作为源来创建数据库实例只读副本。以前，仅支持 RDS for MySQL。有关更多信息，请参阅从多可用区数据库集群创建数据库实例只读副本。	2023 年 5 月 24 日
Amazon RDS 在性能详情控制面板中提供组合的性能详情和 CloudWatch 指标	Amazon RDS 现在在性能详情控制面板中提供性能详情和 CloudWatch 指标的合并视图。有关更多信息，请参阅使用性能详情控制面板查看组合指标。	2023 年 5 月 24 日
Amazon RDS 优化型读取功能在中国区域推出	Amazon RDS 优化型读取功能现已在 Amazon Web Services 区域中国（北京）和中国（宁夏）推出。有关更多信息，请参阅使用 Amazon RDS 优化型读取功能提高 RDS for MariaDB 的查询性能和使用 Amazon RDS 优化型读取功能提高 RDS for MySQL 的查询性能。	2023 年 4 月 24 日
Amazon RDS 在中国区域支持 Amazon Secrets Manager	Amazon RDS 在中国（北京）和中国（宁夏）区域支持 Secrets Manager。有关更多信息，请参阅使用 Amazon RDS 和 Amazon Secrets Manager 管理密码。	2023 年 4 月 20 日

RDS Custom for Oracle 支持对新的 CEV 重用 AMI ID	在创建自定义引擎版本（CEV）时，RDS Custom for Oracle 默认为最新可用的亚马逊机器映像（AMI）。现在，您可以指定以前的 CEV 中使用的 AMI ID。有关更多信息，请参阅创建 CEV。	2023 年 4 月 19 日
Amazon RDS 支持向主题订阅用户发布带有标签的事件	发送到 Amazon Simple Notification Service（Amazon SNS）或 Amazon EventBridge 的 Amazon RDS 事件通知现在在消息正文中包含事件标签。这些标签提供受服务事件影响的资源数据。有关更多信息，请参阅 Amazon RDS 事件通知标签和属性。	2023 年 4 月 17 日
为多可用区数据库集群购买预留实例	您现在可以为多可用区数据库集群购买预留数据库实例。有关更多信息，请参阅多可用区数据库集群的预留数据库实例。	2023 年 4 月 12 日
Amazon RDS 支持 db.m7g 和 db.r7g 实例类	现在，您可以为 RDS for MySQL、RDS for MariaDB 和 RDS for PostgreSQL 数据库实例使用 db.m7g 和 db.r7g 实例类。有关更多信息，请参阅数据库实例类支持的数据库引擎。	2023 年 4 月 12 日

更新 Amazon RDS Custom 服务相关角色权限	`AmazonRDSCustomServiceRolePolicy` 现在授予额外的权限，以允许 RDS Custom for SQL Server 使用 Amazon SQS 和创建快照。有关更多信息，请参阅对 Amazon 托管策略进行的更新。	2023 年 4 月 6 日
使用只读副本迁移到 RDS for MySQL 多可用区数据库集群	您可以使用只读副本，将 RDS for MySQL 单可用区部署或多可用区数据库实例部署迁移到停机时间更少的 RDS for MySQL 多可用区数据库集群部署。有关更多信息，请参阅使用只读副本迁移到多可用区数据库集群。	2023 年 4 月 6 日
从多可用区数据库集群创建数据库实例只读副本	现在，您可以从多可用区数据库集群创建数据库实例只读副本，以便扩展到源集群的计算容量之外。有关更多信息，请参阅从多可用区数据库集群创建数据库实例只读副本。	2023 年 4 月 6 日
Amazon RDS Custom for SQL Server 支持多可用区	您可以使用 RDS Custom for SQL Server 创建多可用区部署。有关更多信息，请参阅管理 RDS Custom for SQL Server 的多可用区部署。	2023 年 4 月 6 日

更新 Amazon 托管式策略权限	`AmazonRDSFullAccess` 和 `AmazonRDSReadOnlyAccess` 策略现在授予额外的权限，以允许在 RDS 控制台中显示 Amazon DevOps Guru 调查发现。有关更多信息，请参阅 Amazon RDS 对 Amazon 托管策略的更新。	2023 年 3 月 30 日
Amazon RDS 支持 Oracle APEX 版本 22.2.v1	您可以将 Oracle APEX 22.2.v1 与所有受支持的 Oracle Database 版本一起使用。有关更多信息，请参阅 Oracle Application Express。	2023 年 3 月 30 日
Amazon DevOps Guru 可用于 RDS for PostgreSQL	RDS for PostgreSQL 提醒您注意 Amazon DevOps Guru 检测到的最近异常情况。控制台的数据库详细信息页面提醒您注意当前和过去 24 小时内发生的异常情况。DevOps Guru 发布主动见解及建议，以帮助您在预测 RDS for PostgreSQL 数据库中的问题发生之前解决它们。有关更多信息，请参阅 DevOps Guru for RDS 工作原理。	2023 年 3 月 30 日
RDS Custom 支持 Amazon EBS gp3 存储卷	RDS Custom for Oracle 和 RDS Custom for SQL Server 都支持 io1、gp2 和 gp3 基于 SSD 的 EBS 卷。有关更多信息，请参阅 RDS Custom for Oracle 的一般要求和 RDS Custom for SQL Server 的一般要求。	2023 年 3 月 29 日

更新 Amazon 托管式策略权限	`AmazonRDSFullAccess` 和 `AmazonRDSReadOnlyAccess` 策略现在会向 Amazon CloudWatch 授予额外的权限。有关更多信息，请参阅 Amazon RDS 对 Amazon 托管策略的更新。	2023 年 3 月 16 日
RDS 代理在中国区域推出	RDS 代理现已在中国（北京）和中国（宁夏）区域推出。有关 RDS 代理的更多信息，请参阅使用 Amazon RDS 代理。	2023 年 3 月 15 日
RDS 代理已在亚太地区（雅加达）区域推出	RDS 代理现已在亚太地区（雅加达）区域推出。有关 RDS 代理的更多信息，请参阅使用 Amazon RDS 代理。	2023 年 3 月 8 日
Amazon RDS 优化型写入功能提高了 RDS for MariaDB 的写入事务性能	您可以使用 Amazon RDS 优化型写入功能来提高 RDS for MariaDB 数据库实例的写入事务性能。有关更多信息，请参阅使用适用于 MariaDB 的 Amazon RDS 优化型写入功能提高写入性能。	2023 年 3 月 7 日
Amazon RDS for PostgreSQL 版本 15.2	Amazon RDS for PostgreSQL 15.2 中的新特征包括适用于条件 SQL 查询的 SQL 标准“MERGE”命令、适用于内存中和基于磁盘的排序的性能改进，以及对逻辑复制的两阶段提交和行/列筛选的支持。	2023 年 2 月 27 日

RDS Custom for Oracle 已在加拿大（中部）区域和南美洲（圣保罗）区域推出	要查看显示所有受支持 Amazon Web Services 区域的表，请参阅支持 RDS Custom for Oracle 的区域和数据库引擎。	2023 年 2 月 22 日
Amazon RDS 支持 RDS for MariaDB 和 RDS for MySQL 的跨区域自动备份	对于 RDS for MariaDB 和 RDS for MySQL 数据库实例，您现在可以在 Amazon Web Services 区域之间复制数据库快照和事务日志。有关更多信息，请参阅将自动备份复制到另一个 Amazon Web Services 区域。	2023 年 2 月 22 日
Amazon RDS for Oracle 支持自动次要版本升级提前通知	RDS 会在 RDS for Oracle 引擎新的次要版本推出日期之前通知您。RDS 在可用日期开始为您的 RDS for Oracle 数据库实例计划自动次要版本升级。有关更多信息，请参阅在计划自动次要版本升级之前。	2023 年 2 月 21 日
Amazon RDS for SQL Server 支持数据库活动流	您现在可以使用数据库活动流监控 SQL Server 数据库实例。SQL Server 数据库实例具有由 Amazon RDS 管理的服务器审计。您可以定义策略以在服务器审计规范中记录服务器事件。您可以创建数据库审计规范并定义策略以记录数据库事件。收集活动流并将其传输到 Amazon Kinesis。从 Kinesis 中，您可以监控活动流以进行进一步分析。有关更多信息，请参阅使用数据库活动流监控 Amazon RDS。	2023 年 2 月 15 日

RDS 支持 MySQL 8.0.32 和 5.7.41	您现在可以创建运行 MySQL 版本 8.0.32 和 5.7.41 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2023 年 2 月 7 日
Amazon RDS for Oracle 对于 SSL 支持新的密码套件	如果您运行 Oracle Database 19c 或 21c，则可以在 RDS for Oracle 的 SSL 选项中指定六个新的密码套件。这些套件支持 FIPS 且符合 FedRAMP 要求。有关更多信息，请参阅 Oracle 安全套接字层。	2023 年 2 月 3 日
Amazon RDS for Oracle 对于 Oracle Enterprise Manager 支持新的密码套件	对于 OEM 选项，您可以使用四个新的符合 FedRAMP 要求的密码套件。有关更多信息，请参阅适用于 Enterprise Manager Cloud Control 的 Oracle Management Agent。	2023 年 2 月 3 日
RDS for Oracle 在亚太地区（海得拉巴）、欧洲（西班牙）和中东（阿联酋）区域中支持数据库活动流	有关更多信息，请参阅支持 Amazon RDS 中数据库活动流的区域和数据库引擎。	2023 年 1 月 27 日
使用只读副本迁移到 RDS for PostgreSQL 多可用区数据库集群	通过使用只读副本，您可以将 RDS for PostgreSQL 单可用区部署或多可用区数据库实例部署迁移到停机时间更少的 RDS for PostgresSQL 多可用区数据库集群部署。有关更多信息，请参阅使用只读副本迁移到多可用区数据库集群。	2023 年 1 月 23 日

Amazon RDS 在亚太地区（墨尔本）区域中推出	Amazon RDS 现已在亚太地区（墨尔本）区域中推出。有关更多信息，请参阅区域和可用区。	2023 年 1 月 23 日
RDS for MariaDB 支持实施 SSL/TLS 连接	RDS for MariaDB 现在支持通过将 `require_secure_transport` 参数设置为 ON 来实施 SSL/TLS 连接。有关更多信息，请参阅与 MariaDB 数据库实例的所有连接都需要 SSL/TLS。	2023 年 1 月 19 日
Amazon RDS 优化读取提高了 RDS for MariaDB 的查询性能	使用 Amazon RDS 优化读取，您可以更快地处理 RDS for MariaDB 数据库实例的查询。有关更多信息，请参阅使用 Amazon RDS 优化读取提高 RDS for MariaDB 的查询性能。	2023 年 1 月 11 日
将多可用区数据库集群快照还原到数据库实例	您现在可以将多可用区数据库集群快照还原到单可用区部署或多可用区数据库实例部署。有关更多信息，请参阅从多可用区数据库集群快照还原到数据库实例。	2023 年 1 月 10 日
在创建数据库实例期间指定证书颁发机构（CA）	现在，您可以在创建数据库实例期间指定将哪个 CA 用于数据库实例的服务器证书。有关更多信息，请参阅证书颁发机构。	2023 年 1 月 5 日

RDS Custom for SQL Server 支持自定义引擎版本	RDS Custom for SQL Server 的自定义引擎版本（CEV）是预安装了 Microsoft SQL Server 的亚马逊机器映像（AMI）。您可以选择 Amazon EC2 Windows AMI 用作基础映像，并可以在操作系统（OS）上安装其他软件。您可以自定义操作系统和 SQL Server 的配置以满足您的企业需求。有关更多信息，请参阅使用 RDS Custom for SQL Server 的自定义引擎版本。	2022 年 12 月 28 日
在其他 Amazon Web Services 区域中使用可用的 Amazon RDS 蓝绿部署	蓝绿部署特征现已在中国（北京）和中国（宁夏）区域推出。有关更多信息，请参阅使用 Amazon RDS 蓝绿部署进行数据库更新。	2022 年 12 月 22 日
IAM 服务相关角色权限更新	AmazonRDSServiceRolePolicy 策略现在会向 Amazon Secrets Manager 授予额外的权限。有关更多信息，请参阅 Amazon RDS 对 Amazon 托管式策略的更新。	2022 年 12 月 22 日
Amazon RDS 支持重命名多可用区数据库集群	您现在可以重命名多可用区数据库集群了。有关更多信息，请参阅重命名多可用区数据库集群。	2022 年 12 月 22 日

Amazon RDS 与 Amazon Secrets Manager 集成以进行密码管理	Amazon RDS 可以在 Secrets Manager 中管理数据库实例或多可用区数据库集群的主用户密码。有关更多信息，请参阅使用 Amazon RDS 和 Amazon Secrets Manager 管理密码。	2022 年 12 月 22 日
Amazon RDS 优化写入支持 db.r6g 和 db.r6gd 数据库实例类	Amazon RDS 优化写入现在支持 db.r6g 和 db.r6gd 数据库实例类。有关更多信息，请参阅使用 Amazon RDS 优化写入提高写入性能。	2022 年 12 月 22 日
Amazon RDS Custom for Oracle 支持新的 Amazon Web Services 区域	您可以在亚太地区（首尔）和亚太地区（大阪）区域中创建 RRDS Custom for Oracle 数据库实例。有关更多信息，请参阅支持 RDS Custom for Oracle 的区域和数据库引擎。	2022 年 12 月 21 日
Amazon RDS on Amazon Outposts 支持只读副本	您现在可以从 RDS on Outposts MySQL 或 PostgreSQL 数据库实例上创建只读副本。有关更多信息，请参阅为 Amazon RDS on Amazon Outposts 创建只读副本。	2022 年 12 月 19 日
RDS Custom for Oracle 支持修改数据库实例类	您现在可以更改 RDS Custom for Oracle 数据库实例的实例类。有关更多信息，请参阅修改 RDS Custom for Oracle 数据库实例。	2022 年 12 月 16 日

RDS for MySQL 和 RDS for PostgreSQL 支持 db.x2iedn 数据库实例类	现在，您可以为 RDS for MySQL 和 RDS for PostgreSQL 数据库实例使用 db.x2iedn 数据库实例类。有关更多信息，请参阅数据库实例类支持的数据库引擎。	2022 年 12 月 14 日
Amazon RDS 优化写入支持 db.x2iedn 数据库实例类	Amazon RDS 优化写入现在支持 db.x2iedn 数据库实例类。有关更多信息，请参阅使用 Amazon RDS 优化写入提高写入性能。	2022 年 12 月 14 日
Amazon RDS 支持在复制数据库快照时复制数据库选项组	现在，您可以跨 Amazon Web Services 账户复制选项组，作为 RDS for Oracle 数据库上快照复制请求的一部分。有关更多信息，请参阅选项组注意事项。	2022 年 12 月 13 日
Amazon RDS 支持使用 RDS for PostgreSQL 版本 14 的 RDS 代理	现在，您可以使用 RDS for PostgreSQL 版本 14 数据库来创建 RDS 代理。有关 RDS 代理的更多信息，请参阅使用 Amazon RDS 代理。	2022 年 12 月 13 日
Amazon RDS for Oracle 支持 db.x2idn、db.x2iedn 和 db.x2iezn 实例类	您现在可以将 db.x2idn、db.x2iedn 和 db.x2iezn 实例类用于 Amazon RDS for Oracle 数据库实例。有关更多信息，请参阅数据库实例类支持的数据库引擎和支持的 RDS for Oracle 实例类。	2022 年 12 月 12 日

RDS for PostgreSQL 数据库实例支持适用于 PostgreSQL 的可信语言扩展	适用于 PostgreSQL 的可信语言扩展是一个开源开发套件，它允许您构建高性能 PostgreSQL 扩展并在您的 RDS for PostgreSQL 数据库实例上安全地运行它们。有关更多信息，请参阅使用适用于 PostgreSQL 的可信语言扩展。	2022 年 11 月 30 日
使用 Amazon RDS 蓝绿部署进行数据库更新	您可以在暂存环境中对数据库实例进行更改，并在不影响生产数据库实例的情况下测试更改。准备就绪后，可以将暂存环境提升为新的生产环境，最大限度地减少停机时间。有关更多信息，请参阅使用 Amazon RDS 蓝绿部署进行数据库更新。	2022 年 11 月 27 日
Amazon RDS 优化写入提高了 RDS for MySQL 的写入事务性能	您可以使用 Amazon RDS 优化写入来提高 RDS for MySQL 数据库实例的写入事务性能。有关更多信息，请参阅使用适用于 MySQL 的 Amazon RDS 优化型写入功能提高写入性能。	2022 年 11 月 27 日
Amazon RDS 优化读取提高了 RDS for MySQL 的查询性能	使用 Amazon RDS 优化读取，您可以更快地处理 RDS for MySQL 数据库实例的查询。有关更多信息，请参阅使用 Amazon RDS 优化读取提高查询性能。	2022 年 11 月 27 日
Amazon RDS 在亚太地区（海得拉巴）区域中推出	Amazon RDS 现已在亚太地区（海得拉巴）区域中推出。有关更多信息，请参阅区域和可用区。	2022 年 11 月 22 日

RDS 支持 MariaDB 10.6.11、10.5.18、10.4.27 和 10.3.37	您现在可以创建运行 MariaDB 版本 10.6.11、10.5.18、10.4.27 和 10.3.37 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MariaDB 版本。	2022 年 11 月 18 日
RDS Custom for Oracle 支持在自定义引擎版本（CEV）中设置非默认安装参数	创建 CEV 时，可以为 Oracle 根目录、Oracle 主目录、UNIX 用户名和 ID 以及 UNIX 组名和 ID 设置非默认值。通过这种方式，您可以更好地控制 RDS Custom for Oracle 数据库实例上的数据库安装。有关更多信息，请参阅准备 CEV 清单。	2022 年 11 月 18 日
Amazon RDS 支持 Oracle APEX 版本 22.1.v1	您可以将 Oracle APEX 22.1.v1 与所有受支持的 Oracle Database 版本一起使用。有关更多信息，请参阅 Oracle Application Express。	2022 年 11 月 18 日
RDS for SQL Server 支持跨区域只读副本	现在，您可以创建跨区域只读副本，以增强灾难恢复能力、减少应用程序读取延迟，并从主数据库实例分载读取工作负载。有关更多信息，请参阅在不同的 Amazon Web Services 区域中创建只读副本。	2022 年 11 月 16 日
Amazon RDS 在欧洲（西班牙）区域中推出	Amazon RDS 现已在欧洲（西班牙）区域中推出。有关更多信息，请参阅区域和可用区。	2022 年 11 月 16 日

RDS for SQL Server 对 Oracle 数据库支持链接服务器	现在，您可以创建链接服务器来访问外部 Oracle 数据库，以读取数据和执行 SQL 命令。有关更多信息，请参阅具有 Oracle OLEDB 及 RDS for SQL Server 的链接服务器。	2022 年 11 月 15 日
RDS Custom for Oracle 支持多租户	您可以将 RDS Custom for Oracle 数据库实例创建为容器数据库（CDB）。创建后，CDB 包含 CDB 根目录、PDB 种子和一个 PDB。您可以使用 Oracle SQL 手动添加更多 PDB。有关更多信息，请参阅 Amazon RDS Custom for Oracle 架构概述。	2022 年 11 月 15 日
Amazon RDS for Oracle 支持 Amazon EFS 集成	如果您将 `EFS_INTEGRATION` 选项添加到您的选项组，您可以在 RDS for Oracle 数据库实例和 Amazon EFS 文件系统之间传输文件。有关更多信息，请参阅 Amazon EFS。	2022 年 11 月 15 日
RDS 支持 MySQL 8.0.31 和 5.7.40	您现在可以创建运行 MySQL 版本 8.0.31 和 5.7.40 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2022 年 11 月 10 日
Amazon RDS 在欧洲（苏黎世）区域中推出	Amazon RDS 现已在欧洲（苏黎世）区域中推出。有关更多信息，请参阅区域和可用区。	2022 年 11 月 9 日

RDS for SQL Server 现在可以访问事务日志备份	您现在可以查看数据库事务日志备份并将其复制到 Amazon S3 桶。有关更多信息，请参阅访问事务日志备份。	2022 年 11 月 7 日
更多 Amazon Web Services 区域支持多可用区数据库集群	现在，更多 Amazon Web Services 区域支持多可用区数据库集群。有关更多信息，请参阅支持 Amazon RDS 中多可用区数据库集群的区域和数据库引擎。	2022 年 11 月 4 日
Amazon RDS 支持 gp3 存储	现在，您可以创建使用 Amazon EBS 通用型 SSD（gp3）存储卷的 Amazon RDS 数据库实例，这使您可以独立于存储容量自定义存储性能。有关更多信息，请参阅通用型 SSD 存储。	2022 年 11 月 4 日
Amazon RDS 支持针对操作系统更新的新事件	Amazon RDS 现在支持新的数据库实例事件 RDS-EVENT-0230，属于安全修补事件类别。当数据库实例有可用的操作系统更新时，此新事件将提醒您。有关更多信息，请参阅监控 Amazon RDS 事件和处理操作系统更新。	2022 年 10 月 28 日

Amazon RDS for Oracle 支持预配置的 r5b 内存优化型实例	db.r5b Oracle 数据库实例类已针对每个 vCPU 需要更多内存、存储和输入/输出的工作负载进行了优化。例如，db.r5b.4xlarge.tpc2.mem2x 开启了多线程，提供的内存是 db.r5b.4xlarge 的两倍。有关更多信息，请参阅 RDS for Oracle 实例类。	2022 年 10 月 27 日
Amazon RDS 对于 RDS for MariaDB、MySQL 和 PostgreSQL 数据库实例支持 15 个只读副本	您现在可以为 RDS for MariaDB、MySQL 和 PostgreSQL 数据库实例创建多达 15 个只读副本。有关只读副本的更多信息，请参阅使用只读副本。	2022 年 10 月 20 日
在数据库预览环境中，Amazon RDS for PostgreSQL 现在支持 PostgreSQL 版本 15 RC 3	PostgreSQL 版本 15 Beta 3 现已在美国东部（俄亥俄州）Amazon Web Services 区域 的数据库预览环境中提供。有关更多信息，请参阅使用数据库预览环境。	2022 年 10 月 18 日
Amazon RDS 支持自动设置 RDS 数据库和 EC2 实例之间的连接	您可以使用 Amazon Web Services Management Console 在现有 RDS 数据库实例或多可用区数据库集群与 EC2 实例之间设置连接。有关更多信息，请参阅自动连接 EC2 实例和 RDS 数据库。	2022 年 10 月 14 日

适用于 PostgreSQL 的 Amazon JDBC 驱动程序已正式发布	适用于 PostgreSQL 的 Amazon JDBC 驱动程序是专为 RDS for PostgreSQL 而设计的客户端驱动程序。适用于 PostgreSQL 的 Amazon JDBC 驱动程序现已正式发布。有关更多信息，请参阅连接适用于 PostgreSQL 的 Amazon JDBC 驱动程序。	2022 年 10 月 6 日
Amazon RDS for Oracle 支持 Oracle APEX 版本 21.2.v1	Oracle APEX 21.2 包括补丁 33420059。有关更多信息，请参阅 Oracle APEX 版本要求。	2022 年 10 月 3 日
RDS 支持 MySQL 5.7.39	您现在可以创建运行 MySQL 版本 5.7.39 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2022 年 9 月 29 日
RDS 支持 MariaDB 10.6.10	您现在可以创建运行 MariaDB 版本 10.6.10 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MariaDB 版本。	2022 年 9 月 29 日
RDS 代理支持 RDS for SQL Server	现在，您可以为运行 Microsoft SQL Server 版本 2014 或更高版本的 RDS 数据库实例创建 RDS 代理。有关 RDS 代理的更多信息，请参阅使用 Amazon RDS 代理。	2022 年 9 月 19 日

RDS 支持 MariaDB 10.5.17、10.4.26 和 10.3.36	您现在可以创建运行 MariaDB 版本 10.5.17、10.4.26 和 10.3.36 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MariaDB 版本。	2022 年 9 月 15 日
Amazon RDS for Oracle 对临时数据支持本地实例存储	现在，您可以在 Amazon EC2 db.r5d 和 db.m5d 实例类型上启动 Amazon RDS for Oracle，并将临时表空间和数据库智能闪存缓存（闪存缓存）配置为使用实例存储。与基于 Amazon EBS 的标准存储相比，在本地存储临时数据可实现更低的读写延迟。有关更多信息，请参阅在实例存储中存储临时 Oracle 数据。	2022 年 9 月 14 日
性能详情显示前 25 个 SQL 查询	在性能详情控制面板中，Top SQL（热门 SQL）选项卡将显示对数据库负载影响最大的 25 个 SQL 查询。有关详细信息，请参阅 Top SQL（主要 SQL）选项卡概览。	2022 年 9 月 13 日
RDS 支持 MySQL 8.0.30	您现在可以创建运行 MySQL 版本 8.0.30 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2022 年 9 月 9 日
Amazon RDS 在中东（阿联酋）区域中推出	Amazon RDS 现已在中东（阿联酋）区域中推出。有关更多信息，请参阅区域和可用区。	2022 年 8 月 30 日

Amazon RDS for SQL Server 支持 SSRS 电子邮件订阅	现在，您可以使用 SQL Server Reporting Services（SSRS）电子邮件扩展向用户发送报告并在报告服务器上订阅报告。有关更多信息，请参阅 RDS for SQL Server 中的 SQL Server Reporting Services 支持。	2022 年 8 月 26 日
RDS for Oracle 支持只读副本备份	您可以启用自动备份并为 RDS for Oracle 副本创建手动快照。有关更多信息，请参阅使用 RDS for Oracle 副本备份。	2022 年 8 月 23 日
RDS for Oracle 支持 Oracle Data Guard 切换	切换是主数据库与已挂载或打开的 Oracle 副本之间的角色反转。在切换期间，原始主数据库转换为备用角色，而原始备用数据库转换为主角色。有关更多信息，请参阅执行 Oracle Data Guard 切换。	2022 年 8 月 23 日
Amazon RDS 支持自动设置与 EC2 实例的连接	当您创建数据库实例或多可用区数据库集群时，可以使用 Amazon Web Services Management Console在 Amazon Elastic Compute Cloud 实例和新的数据库实例或数据库集群之间设置连接。有关更多信息，请参阅配置 EC2 实例的自动网络连接（对于新的数据库实例）和配置 EC2 实例的自动网络连接（对于新的数据库集群）。	2022 年 8 月 22 日

RDS Custom for Oracle 支持提升 Oracle 副本	如果您使用 RDS Custom for Oracle，则可以使用 `promote-read-replica` CLI 命令提升托管式 Oracle 副本。此外，您可以删除主数据库实例，这会导致 RDS Custom for Oracle 将您的托管式 Oracle 副本提升为独立实例。有关更多信息，请参阅使用 RDS Coustom for Oracle 的 Oracle 副本。	2022 年 8 月 5 日
RDS for MySQL 支持实施 SSL/TLS 连接	RDS for MySQL 现在支持通过将 `require_secure_transport` 参数设置为 ON 来实施 SSL/TLS 连接。有关更多信息，请参阅需要至 MySQL 数据库实例的 SSL/TLS 连接。	2022 年 8 月 1 日
Amazon RDS 已停止对 Oracle Database 12c 第 1 版（12.1.0.2）的支持	对于 BYOL 和 LI 许可模型，已停止对版本 12.1.0.2 的支持。2022 年 8 月 1 日，RDS for Oracle 开始自动升级 12c 第 1 版本（12.1.0.2）数据库实例，并将 12.1.0.2 快照还原到 Oracle Database 19c。有关更多信息，请参阅 Amazon re:Post 上的支持结束时间表。	2022 年 8 月 1 日
RDS 代理支持 RDS for MariaDB	现在，您可以为运行 MariaDB 版本 10.2、10.3、10.4 或 10.5 的 RDS 数据库实例创建 RDS 代理。MariaDB 支持包含在 MySQL 引擎系列中。有关 RDS 代理的更多信息，请参阅使用 Amazon RDS 代理。	2022 年 7 月 26 日

RDS for MariaDB 支持 db.r5b 数据库实例类	您现在可以创建使用 db.r5b 数据库实例类的 RDS for MariaDB 数据库实例类。有关更多信息，请参阅数据库实例类支持的数据库引擎。	2022 年 7 月 25 日
RDS for Oracle 支持修改数据库活动流	如果您使用 RDS for Oracle，则可以将数据库活动流的审核策略状态更改为锁定（默认）或已解锁。您可以解锁活动流的策略状态，自定义审核策略，然后重新锁定策略状态，而不是停止活动流。有关更多信息，请参阅修改数据库活动流。	2022 年 7 月 22 日
性能详情支持亚太地区（雅加达）区域	以前，您无法在亚太地区（雅加达）区域使用性能详情。这一限制已删除。有关更多信息，请参阅支持 Amazon RDS 中 Performance Insights 的区域和数据库引擎。	2022 年 7 月 21 日
Amazon RDS 上的 Microsoft SQL Server 2012 已经终止支持	Microsoft SQL Server 2012 已经终止支持，与 Microsoft 计划于 2022 年 7 月 12 日结束对此版本的延长支持一致。从 2022 年 6 月 1 日起，任何现有 Microsoft SQL Server 2012 实例都将自动升级到 Microsoft SQL Server 2014 的最新次要版本。有关更多信息，请参阅 Amazon RDS 上的 Microsoft SQL Server 2012 支持。	2022 年 7 月 12 日

RDS 支持 MariaDB 10.6.8、10.5.16、10.4.25、10.3.35 和 10.2.44	您现在可以创建运行 MariaDB 版本 10.6.8、10.5.16、10.4.25、10.3.35 和 10.2.44 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上支持的 MariaDB 版本。	2022 年 7 月 8 日
RDS 性能详情支持额外的保留期	以前，性能详情仅提供两个保留期：7 天（默认）或 2 年（731 天）。现在，如果您需要将性能数据保留超过 7 天，则可以指定 1-24 个月。有关更多信息，请参阅性能详情的定价和数据留存。	2022 年 7 月 1 日
RDS Custom 支持亚太地区（孟买）和欧洲（伦敦）区域	您可以在两个新的 Amazon Web Services 区域中创建 RDS Custom for Oracle 和 RDS Custom for SQL Server 数据库实例：亚太地区（孟买）和欧洲（伦敦）。有关更多信息，请参阅 RDS Custom for Oracle 的 Amazon Web Services 区域支持和 RDS Custom for SQL Server 的 Amazon Web Services 区域支持。	2022 年 6 月 21 日

RDS Custom for Oracle 支持 Oracle Database 18c 和 12c 第 2 版 (12.2)	您现在可以使用 Oracle Database 18c 和 12c 第 2 版 (12.2) 的安装文件创建用于 RDS Custom for Oracle 的 CEV。您可以使用这些 CEV 创建 RDS Custom for Oracle 数据库实例。有关更多信息，请参阅使用 Amazon RDS Custom for Oracle 的自定义引擎版本。	2022 年 6 月 21 日
多可用区数据库集群支持 db.m5d 和 db.r5d 数据库实例类	您现在可以创建使用 db.m5d 和 db.r5d 数据库实例类的多可用区数据库集群。有关更多信息，请参阅多可用区数据库集群部署和数据库实例类类型。	2022 年 6 月 21 日
多可用区数据库集群在更多 Amazon Web Services 区域提供	您现在可以在以下区域中创建多可用区数据库集群：欧洲（法兰克福）和欧洲（斯德哥尔摩）。有关更多信息，请参阅多可用区数据库集群部署。	2022 年 6 月 21 日
RDS for Microsoft SQL Server 支持迁移使用透明数据加密 (TDE) 的数据库	RDS for SQL Server 现在支持使用本机备份和还原在启用 TDE 的情况下迁移 Microsoft SQL Server 数据库。有关更多信息，请参阅 SQL Server 中的透明数据加密支持。	2022 年 6 月 14 日

Amazon RDS 支持将事件发布到加密的 Amazon SNS 主题	Amazon RDS 现在可以将事件发布到启用了服务器端加密 (SSE) 的 Amazon Simple Notification Service (Amazon SNS) 主题，以便为携带敏感数据的事件提供额外保护。有关更多信息，请参阅订阅 Amazon RDS 事件通知。	2022 年 6 月 1 日
RDS 支持 MySQL 5.7.38	您现在可以创建运行 MySQL 版本 5.7.38 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2022 年 5 月 31 日
RDS for PostgreSQL 支持级联只读副本	现在，您可以将级联只读副本与 RDS for PostgreSQL 版本 14.1 及更高版本结合使用。有关更多信息，请参阅在 Amazon RDS 中使用 PostgreSQL 只读副本。	2022 年 5 月 4 日
Amazon RDS on Amazon Outposts 支持扩展存储和自动扩展操作	现在，您可以在 Outpost 上更改数据库实例的存储大小并使用存储自动扩展。有关更多信息，请参阅 Amazon RDS on Amazon Outposts 对 Amazon RDS 特征的支持。	2022 年 5 月 2 日
多可用区数据库集群在更多 Amazon Web Services 区域提供	您现在可在以下区域创建多可用区数据库集群：亚太地区（新加坡）和亚太地区（悉尼）。有关更多信息，请参阅多可用区数据库集群部署。	2022 年 4 月 29 日

Amazon RDS 支持双堆栈模式	数据库实例现在可以在双堆栈模式下运行。在双堆栈模式下，资源可通过 IPv4 和/或 IPv6 与数据库实例进行通信。有关更多信息，请参阅 Amazon RDS IP 寻址。	2022 年 4 月 29 日
Amazon RDS 会将使用情况指标发布到 Amazon CloudWatch	Amazon CloudWatch 中的 `AWS/Usage` 命名空间中包含 Amazon RDS 服务配额的账户级使用情况指标。有关更多信息，请参阅 Amazon RDS 的 Amazon CloudWatch 使用情况指标。	2022 年 4 月 28 日
Amazon RDS for MySQL 支持 db.m6i 和 db.r6i 数据库实例类	您现在可以将 db.m6i 和 db.r6i 数据库实例类用于运行 MySQL 的 Amazon RDS 数据库实例。有关更多信息，请参阅数据库实例类支持的数据库引擎。	2022 年 4 月 28 日
Amazon RDS for PostgreSQL 支持 db.m6i 和 db.r6i 数据库实例类	您现在可以将 db.m6i 和 db.r6i 数据库实例类用于运行 PostgreSQL 的 Amazon RDS 数据库实例。有关更多信息，请参阅数据库实例类支持的数据库引擎。	2022 年 4 月 27 日
Amazon RDS for MariaDB 支持 db.m6i 和 db.r6i 数据库实例类	您现在可以将 db.m6i 和 db.r6i 数据库实例类用于运行 MariaDB 的 Amazon RDS 数据库实例。有关更多信息，请参阅数据库实例类支持的数据库引擎。	2022 年 4 月 26 日

Amazon RDS on Amazon Outposts 支持多可用区部署	您现在可在其他 Outpost 上创建备用数据库实例。有关更多信息，请参阅 Amazon RDS on Amazon Outposts 对 Amazon RDS 特征的支持。	2022 年 4 月 19 日
Amazon RDS for Oracle 支持 db.m6i 和 db.r6i 数据库实例类	如果您运行 Oracle Database 19c，可以使用 db.m6i 和 db.r6i 实例类。db.m6i 类是通用实例类，非常适合广泛的工作负载。有关更多信息，请参阅 RDS for Oracle 实例类。	2022 年 4 月 8 日
Amazon RDS for SQL Server 支持 SQL Server Agent 作业复制	启用此特征后，在主要主机上创建、修改或删除的 SQL Server Agent 作业将自动同步到多可用区配置中的辅助主机。有关更多信息，请参阅使用 SQL Server Agent。	2022 年 4 月 7 日
Amazon RDS 支持使用 RDS for PostgreSQL 版本 13 的 RDS Proxy	现在，您可以使用 RDS for PostgreSQL 版本 13 数据库来创建 RDS Proxy。有关 RDS 代理的更多信息，请参阅使用 Amazon RDS 代理。	2022 年 4 月 4 日
Amazon RDS 计划弃用 Oracle Database 12c	将弃用 Oracle Database 12c。在 Oracle Database 12c 版本支持结束日期之后，Oracle Corporation 将不再为其提供补丁。Amazon RDS 计划开始自动将 Oracle Database 12c 数据库实例升级到 Oracle Database 19c。	2022 年 3 月 22 日

Amazon RDS for PostgreSQL 发布说明	现在提供有 Amazon RDS for PostgreSQL 发布说明的单独指南。有关更多信息，请参阅 Amazon RDS for PostgreSQL 发布说明。	2022 年 3 月 22 日
Amazon RDS for Oracle 发布说明	现在提供有 Amazon RDS for Oracle 发布说明的单独指南。有关更多信息，请参阅 Amazon RDS for Oracle 发布说明。	2022 年 3 月 22 日
多可用区数据库集群在更多 Amazon Web Services 区域提供	您现在可以在以下区域创建多可用区数据库集群：美国东部（俄亥俄州）和亚太地区（东京）。有关更多信息，请参阅多可用区数据库集群部署。	2022 年 3 月 15 日
Amazon RDS for PostgreSQL 版本 14.2、13.6、12.10、11.15 和 10.20	RDS for PostgreSQL 现在支持版本 14.2、13.6、12.10、11.15 和 10.20。版本 14.2 和 13.6 增加了对两个新的外部数据包装器的支持。`mysql_fdw` 扩展允许 PostgreSQL 处理存储在 MySQL、MariaDB 和 Aurora MySQL 数据库中的数据。`tds_fdw` 扩展允许 PostgreSQL 处理存储在 SQL Server 数据库中的数据。有关更多信息，请参阅受支持的 PostgreSQL 数据库版本。	2022 年 3 月 12 日
RDS 支持 MySQL 5.7.37	您现在可以创建运行 MySQL 版本 5.7.37 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2022 年 3 月 11 日

Amazon RDS for SQL Server 支持新的数据库实例类	您现在可以创建运行 Microsoft SQL Server 并使用 db.m6i 和 db.r6i 数据库实例类的 Amazon RDS 数据库实例。有关更多信息，请参阅 Microsoft SQL Server 的数据库实例类支持。	2022 年 3 月 9 日
Amazon RDS for Oracle 支持 Oracle Database 21c	您现在可以创建运行 Oracle Database 21c (21.0.0.0) 的 Amazon RDS 数据库实例。这是首个仅支持多租户 (CDB) 体系结构的 Oracle 数据库版本。有关更多信息，请参阅适用于 Amazon RDS 的 Oracle Database 21c。	2022 年 3 月 7 日
RDS 支持 MariaDB 10.6.7、10.5.15、10.4.24、10.3.34 和 10.2.43	您现在可以创建运行 MariaDB 版本 10.6.7、10.5.15、10.4.24、10.3.34 和 10.2.43 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MariaDB 版本。	2022 年 3 月 3 日
适用于 MySQL 的 Amazon JDBC 驱动程序已正式发布	适用于 MySQL 的 Amazon JDBC 驱动程序是专为 RDS for MySQL 而设计的客户端驱动程序。适用于 MySQL 的 Amazon JDBC 驱动程序现已正式发布。有关更多信息，请参阅连接适用于 MySQL 的 Amazon Web Services JDBC 驱动程序。	2022 年 3 月 2 日

多可用区数据库集群已正式发布	一个多可用区数据库集群部署是 Amazon RDS 的高可用性部署模式，具有两个可读的备用数据库实例。多可用区数据库集群现已正式发布。有关更多信息，请参阅多可用区数据库集群部署。	2022 年 3 月 1 日
RDS 支持 MySQL 8.0.28	您现在可以创建运行 MySQL 版本 8.0.28 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2022 年 2 月 28 日
Amazon RDS for Oracle 支持本机网络加密 (NNE) 的新设置	要控制客户端是否可以使用非安全加密和校验和方法进行连接，请在 NNE 选项中设置 `SQLNET.ALLOW_WEAK_CRYPTO_CLIENTS` 和 `SQLNET.ALLOW_WEAK_CRYPTO`。不安全方法的例子包括 DES、3DES、RC4 和 MD5。有关更多信息，请参阅 NNE 选项设置。	2022 年 2 月 25 日
Amazon RDS for SQL Server 支持 Microsoft SQL Server 2017 标准版的 Always On 可用性组	当您使用 SQL Server 2017 标准版 14.00.3401.7 及更高版本上的多可用区配置创建数据库实例时，RDS 会自动使用可用性组。有关更多信息，请参阅 Microsoft SQL Server 的多可用区部署。	2022 年 2 月 18 日
RDS for Oracle 支持亚太地区（雅加达）区域的数据库活动流	有关更多信息，请参阅数据库活动流的 Amazon Web Services 区域 支持。	2022 年 2 月 16 日

Amazon RDS Custom for Oracle 支持 Oracle Database 12.1	现在，您可以为使用 Oracle Database 12.1 企业版的 RDS Custom for Oracle 创建自定义引擎版本。有关更多信息，请参阅使用 Amazon RDS Custom for Oracle 的自定义引擎版本。	2022 年 2 月 4 日
Amazon RDS for MariaDB 支持新的主要版本	您现在可以创建运行 MariaDB 版本 10.6 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MariaDB 10.6 支持。	2022 年 2 月 3 日
性能详情支持 Oracle 查询的计划捕获	性能详情控制台支持主要 SQL 的新计划维度。按计划切片时，您可以看到主要 Oracle 查询正在使用哪些计划。如果查询使用多个计划，则可在控制台中并排比较计划，确定哪个计划最有效。您还可以深入分析，了解计划中哪些步骤的成本最高。有关更多信息，请参阅使用 Performance Insights 控制面板分析 Oracle 执行计划。	2022 年 1 月 27 日

性能详情支持新的 API	性能详情支持以下 API：`GetResourceMetadata` 、`ListAvailableResourceDimensions` 和 `ListAvailableResourceMetrics` 。有关更多信息，请参阅本说明中的使用 Performance Insights API 检索指标以及《Amazon RDS Performance Insights API 参考》。	2022 年 1 月 12 日
RDS Proxy 支持事件	RDS Proxy 现在会生成可供您在 CloudWatch Events 中订阅和查看的事件，或可供您配置后发送到 Amazon EventBridge 的事件。有关更多信息，请参阅使用 RDS Proxy 事件。	2022 年 1 月 11 日
Amazon RDS for SQL Server 支持 SSAS 多维模式	RDS for SQL Server 支持在表格模式或多维模式下运行 SQL Server Analysis Services (SSAS)。有关更多信息，请参阅支持 RDS for SQL Server 中的 SQL Server Analysis Services。	2022 年 1 月 7 日
在其他 Amazon Web Services 区域提供的 RDS 代理	以下区域现在可提供 RDS 代理：非洲（开普敦）、亚太地区（香港）、亚太地区（大阪）、欧洲（米兰）、欧洲（巴黎）、欧洲（斯德哥尔摩）、中东（巴林）和南美洲（圣保罗）。有关 RDS 代理的更多信息，请参阅使用 Amazon RDS 代理。	2022 年 1 月 5 日

RDS 支持 MySQL 8.0.27	您现在可以创建运行 MySQL 版本 8.0.27 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2021 年 12 月 21 日
Amazon RDS 已在亚太地区（雅加达）区域推出	Amazon RDS 现已在亚太地区（雅加达）区域推出。有关更多信息，请参阅区域和可用区。	2021 年 12 月 13 日
Amazon RDS 支持 MariaDB 10.5.13、10.4.22、10.3.32 和 10.2.41	您现在可以创建运行 MariaDB 版本 10.5.13、10.4.22、10.3.32 和 10.2.41 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MariaDB 版本。	2021 年 12 月 8 日
Amazon RDS Custom for SQL Server	Amazon RDS Custom 是一种托管式数据库服务，适用于需要访问底层操作系统和数据库环境的旧版、自定义和打包应用程序。借助 Amazon RDS Custom，您可以获得 Amazon RDS 的自动化操作和 Amazon EC2 的灵活度。有关更多信息，请参阅使用 Amazon RDS Custom。	2021 年 12 月 1 日

多可用区数据库集群（预览）	现在，您可以为 RDS for MySQL 和 RDS for PostgreSQL 创建多可用区数据库集群了。多可用区数据库集群部署是 Amazon RDS 的高可用性部署模式，具有两个可读的备用数据库实例。多可用区数据库集群目前为预览版功能。有关更多信息，请参阅多可用区数据库集群部署（预览版）。	2021 年 11 月 23 日
Amazon RDS 借助 RDS for PostgreSQL 版本 12 支持 RDS Proxy	现在，您可以使用 RDS for PostgreSQL 版本 12 的数据库来创建 RDS Proxy 了。有关 RDS Proxy 的更多信息，请参阅使用 Amazon RDS Proxy。	2021 年 11 月 22 日
Amazon RDS on Amazon Outposts 支持本地备份	您可以将自动备份和手动快照存储在您的 Amazon Web Services 区域中或您本地的 Outpost 中。有关更多信息，请参阅 Amazon RDS on Amazon Outposts 对 Amazon RDS 特征的支持。	2021 年 11 月 22 日
Amazon RDS 支持跨账户 Amazon KMS keys	在将数据库快照导出到 Amazon S3 时，您可以使用来自不同 Amazon 账户的 KMS 密钥进行加密。有关更多信息，请参阅将数据库快照数据导出到 Amazon S3。	2021 年 11 月 3 日

Amazon RDS on Amazon Outposts 支持将数据库引擎日志发布到 CloudWatch Logs	RDS on Outposts 现在支持将数据库引擎日志发布到 CloudWatch Logs。有关更多信息，请参阅 Amazon RDS on Amazon Outposts 对 Amazon RDS 特征的支持。	2021 年 11 月 2 日
Amazon RDS Custom for Oracle	Amazon RDS Custom 是一种托管式数据库服务，适用于需要访问底层操作系统和数据库环境的旧版、自定义和打包应用程序。借助 Amazon RDS Custom，您可以获得 Amazon RDS 的自动化操作和 Amazon EC2 的灵活度。有关更多信息，请参阅使用 Amazon RDS Custom。	2021 年 10 月 26 日
支持 RDS for MySQL 版本 8.0 的延迟复制	从 RDS for MySQL 版本 8.0.26 开始，您可以为 RDS for MySQL 版本 8.0 数据库实例配置复制了。有关更多信息，请参阅使用 MySQL 配置延迟复制。	2021 年 10 月 25 日
支持 MySQL 8.0.26	您现在可以创建运行 MySQL 版本 8.0.26 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2021 年 10 月 25 日

RDS for MySQL 版本 8.0 支持基于 GTID 的复制	从 RDS for MySQL 版本 8.0.26 开始，您可以为 RDS for MySQL 版本 8.0 数据库实例配置基于 GTID 的复制了。有关更多信息，请参阅将基于 GTID 的复制用于 RDS for MySQL。	2021 年 10 月 25 日
Amazon RDS 借助 RDS for MySQL 8.0 支持 RDS Proxy	现在，您可以为 RDS for MySQL 8.0 数据库实例创建 RDS Proxy 了。有关更多信息，请参阅使用 Amazon RDS Proxy。	2021 年 10 月 21 日
Amazon RDS on Amazon Outposts 支持其他 RDS for MySQL 版本	RDS on Outposts 现支持 RDS for MySQL 版本 8.0.23 和 8.0.25。有关更多信息，请参阅 Amazon RDS on Amazon Outposts 对 Amazon RDS 特征的支持。	2021 年 10 月 20 日
在数据库预览环境中，Amazon RDS for PostgreSQL 现在支持 PostgreSQL 版本 14 RC 1	PostgreSQL 版本 14 RC 1 现已在美国东部（俄亥俄州）Amazon Web Services 区域 的数据库预览环境中提供。有关更多信息，请参阅使用数据库预览环境。	2021 年 10 月 19 日
Amazon RDS 还在其他 Amazon Web Services 区域支持性能详情	中东（巴林）、非洲（开普敦）、欧洲（米兰）和亚太地区（大阪）区域现已提供性能详情。有关更多信息，请参阅支持 Amazon RDS 中 Performance Insights 的区域和数据库引擎。	2021 年 10 月 5 日

性能详情支持 Oracle 的摘要级统计数据	使用性能详情时，您可以查看 Amazon RDS for Oracle 的语句和摘要级别的 SQL 统计信息。有关更多信息，请参阅分析正在 Oracle 中运行的查询。	2021 年 10 月 4 日
Amazon RDS on Amazon Outposts 支持其他 RDS for PostgreSQL 版本	RDS on Outposts 现支持 RDS for PostgreSQL 版本 12.8 和 13.4。有关更多信息，请参阅 Amazon RDS on Amazon Outposts 对 Amazon RDS 特征的支持。	2021 年 10 月 1 日
Amazon RDS for Oracle 支持 Oracle APEX 版本 21.1.v1	您可以将 Oracle APEX 21.1.v1 与所有受支持的 Oracle Database 版本一起使用。有关更多信息，请参阅 Oracle Application Express。	2021 年 9 月 24 日
Amazon RDS for Oracle 支持 NNE 的客户端加密	配置 NNE 时，您可能希望避免在服务器端强制加密。例如，您可能不希望强制所有客户端通信使用加密，因为服务器需要加密。在这种情况下，您可以使用 `SQLNET.*CLIENT` 选项强制在客户端加密。有关更多信息，请参阅 Oracle 本机网络加密。	2021 年 9 月 24 日
Amazon RDS for MySQL 和 RDS for PostgreSQL 支持新的数据库实例类	您现在可以使用 db.r5b、db.t4g 和 db.x2g 数据库实例类创建运行 MySQL 或 PostgreSQL 的 Amazon RDS 数据库实例。有关更多信息，请参阅数据库实例类支持的数据库引擎。	2021 年 9 月 15 日

Amazon RDS for Microsoft SQL Server 通过 Microsoft 分布式事务处理协调器 (MSDTC) 支持 Java Database Connectivity (JDBC)	SQL Server 2017 14.00.3223.3 及更高版本以及 SQL Server 2019 现在支持 JDBC XA 事务。有关更多信息，请参阅 RDS for SQL Server 中的 Microsoft 分布式事务处理协调器支持。	2021 年 9 月 7 日
Amazon RDS 支持 MariaDB 10.5.12、10.4.21、10.3.31 和 10.2.40	您现在可以创建运行 MariaDB 版本 10.5.12、10.4.21、10.3.31 和 10.2.40 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MariaDB 版本。	2021 年 9 月 2 日
Amazon RDS 已结束对 Oracle Database 18c 的支持	您只能为 Oracle Database 12c 和 Oracle Database 19c 创建数据库实例。如果您有 Oracle 数据库 18c 快照，请将其升级到更高版本。有关详细信息，请参阅升级 Oracle 数据库快照。	2021 年 8 月17 日
在数据库预览环境中，Amazon RDS for PostgreSQL 现在支持 PostgreSQL 版本 14 测试版 2	有关 PostgreSQL 版本 14 测试版 1 的更多信息，请参阅 PostgreSQL 14 测试版 1 发布说明。有关 PostgreSQL 版本 14 测试版 2 的更多信息，请参阅 PostgreSQL 14 测试版 2 发布说明。有关数据库预览环境的信息，请参阅使用数据库预览环境。	2021 年 8 月 9 日

Amazon RDS 在共享 VPC 中支持 RDS Proxy	您现在可在共享 VPC 中创建 RDS 代理。有关 RDS 代理的更多信息，请参阅 Amazon RDS 用户指南或 Aurora 用户指南中的“使用 Amazon RDS 代理管理连接”。	2021 年 8 月 6 日
Amazon RDS 支持 MariaDB 10.2.39	您现在可以创建运行 MariaDB 版本 10.2.39 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MariaDB 版本。	2021 年 8 月 4 日
Amazon RDS for Oracle 添加了 `TIMEZONE_FILE_AUTO_UPGRADE` 选项	使用此选项，您可以将当前时区文件升级到 Oracle 数据库实例上的最新版本。有关更多信息，请参阅 Oracle 时区文件自动升级。	2021 年 7 月 30 日
Amazon RDS 扩展了对跨区域自动备份的支持	您现在可以在更多 Amazon Web Services 区域 之间复制数据库快照和事务日志。有关更多信息，请参阅将自动备份复制到另一个 Amazon 区域。	2021 年 7 月 19 日
支持 MySQL 5.7.34	您现在可以创建运行 MySQL 版本 5.7.34 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2021 年 7 月 8 日
Amazon RDS on Amazon Outposts 支持其他 RDS for PostgreSQL 版本	RDS on Outposts 现支持 RDS for PostgreSQL 版本 12.7 和 13.3。有关更多信息，请参阅 Amazon RDS on Amazon Outposts 对 Amazon RDS 特征的支持。	2021 年 7 月 8 日

Amazon RDS for PostgreSQL 支持 oracle_fdw	现在，您可以使用 oracle_fdw 扩展为访问 Oracle 数据库提供外部数据包装器。有关更多信息，请参阅使用 oracle_fdw 扩展访问外部数据。	2021 年 7 月 8 日
Amazon RDS 支持 Oracle Management Agent (OMA) 版本 13.5	您可以将 Oracle Management Agent（OMA）版本 13.5 与 Oracle Enterprise Manager（OEM）Cloud Control 13c Release 5 及更高版本结合使用。Amazon RDS for Oracle 会安装 OMA，后者随后可与您的 Oracle Management Service (OMS) 通信以提供监控信息。如果运行 OMS 13.5，可以通过安装 OMA 13.5 来管理数据库。有关更多信息，请参阅适用于 Enterprise Manager Cloud Control 的 Oracle Management Agent。	2021 年 7 月 7 日
Amazon RDS for Oracle 支持从 Amazon S3 下载日志	如果存档的重做日志不在您的实例上，但受到备份保留期的保护，则可以使用 `rdsadmin.rdsadmin_archive_log_download` 从 Amazon S3 下载。RDS for Oracle 会将日志保存到数据库实例的 `/rdsdbdata/log/arch` 目录。有关更多信息，请参阅从 Amazon S3 下载归档重做日志。	2021 年 7 月 2 日

Amazon RDS 支持 MariaDB 10.4.18 和 10.5.9	您现在可以创建运行 MariaDB 版本 10.4.18 和 10.5.9 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MariaDB 版本。	2021 年 6 月 30 日
Amazon RDS for Oracle 支持数据库活动流	您现在可以使用数据库活动流监控 Oracle 数据库实例。Oracle 数据库会将审计记录写入统一审计跟踪。在 Oracle 数据库实例上启动数据库活动流时，Amazon Kinesis 会流式处理与 Oracle 数据库审核策略匹配的所有活动。有关更多信息，请参阅使用数据库活动流监控 Amazon RDS。	2021 年 6 月 23 日
Amazon RDS for Oracle 推出内存优化型实例类	新的 Oracle 数据库实例类已针对每个 vCPU 需要更多内存、存储和输入/输出的工作负载进行了优化。有关更多信息，请参阅 RDS for Oracle 实例类。	2021 年 6 月 23 日
支持 MySQL 8.0.25	您现在可以创建运行 MySQL 版本 8.0.25 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2021 年 6 月 18 日
Amazon RDS on Amazon Outposts 支持其他 RDS for PostgreSQL 版本	RDS on Outposts 现支持 RDS for PostgreSQL 版本 12.5、12.6、13.1 和 13.2。有关更多信息，请参阅 Amazon RDS on Amazon Outposts 对 Amazon RDS 特征的支持。	2021 年 5 月 28 日

Amazon RDS 支持 MariaDB 10.2.37 和 10.3.28	您现在可以创建运行 MariaDB 版本 10.2.37 和 10.3.28 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MariaDB 版本。	2021 年 5 月 27 日
Amazon RDS for Oracle 支持多租户容器数据库 (CDB)	多租户架构使 Oracle 数据库成为 CDB。在 Oracle 数据库 19c 中，您的 CDB 可以包含单个 PDB。PDB 的用户体验与非 CDB 的用户体验基本相同。有关更多信息，请参阅 RDS for Oracl 架构。	2021 年 5 月 25 日
Amazon RDS on Amazon Outposts 支持 Amazon RDS for SQL Server	RDS on Outposts 现支持 Amazon RDS for SQL Server。有关更多信息，请参阅 Amazon RDS on Amazon Outposts 对 Amazon RDS 特征的支持。	2021 年 5 月 11 日
Amazon RDS 扩展了对跨区域自动备份的支持	您现在可以将运行 Microsoft SQL Server 的 Amazon RDS 数据库实例配置为将数据库快照和事务日志复制到其他Amazon区域。有关更多信息，请参阅将自动备份复制到另一个 Amazon 区域。	2021 年 5 月 7 日
Amazon RDS 支持加密数据库实例的跨区域自动备份	对于运行 Oracle 或 PostgreSQL 的加密 Amazon RDS 数据库实例，您现可将数据库快照和事务日志复制到其他Amazon区域。有关更多信息，请参阅将自动备份复制到另一个 Amazon 区域。	2021 年 5 月 3 日

Amazon RDS on Amazon Outposts 支持 Amazon CloudWatch 监控	RDS on Outposts 现在支持 Amazon CloudWatch 监控。有关更多信息，请参阅 Amazon RDS on Amazon Outposts 对 Amazon RDS 特征的支持。	2021 年 4 月 21 日
RDS for PostgreSQL 支持 Amazon Lambda 函数	您现在可以为 RDS for PostgreSQL 数据库实例调用 Amazon Lambda 函数。有关更多信息，请参阅从 RDS for PostgreSQL 数据库实例中调用 Amazon Lambda 函数。	2021 年 4 月 13 日
RDS for SQL Server 支持扩展事件	您可以使用 SQL Server 扩展事件来捕获调试和故障排除信息。有关更多信息，请参阅通过 Amazon RDS for Microsoft SQL Server 使用扩展事件。	2021 年 4 月 8 日
支持 MySQL 8.0.23、5.7.33 和 5.6.51	您现在可以创建运行 MySQL 版本 8.0.23、5.7.33 和 5.6.51 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2021 年 3 月 31 日
自动回滚失败的 Amazon RDS for MySQL 升级	如果数据库实例从 MySQL 5.7 版升级到 MySQL 8.0 版失败，Amazon RDS 会自动回滚为升级执行的更改。回滚后，MySQL 数据库实例将运行 MySQL 版本 5.7。有关详细信息，请参阅从 MySQL 5.7 升级到 8.0 失败后回滚。	2021 年 3 月 18 日

Amazon RDS 支持选择加入区域中的跨区域只读副本	现在，您可以将数据库实例复制到选择加入区域。有关更多信息，请参阅在不同的 Amazon 区域中创建只读副本。	2021 年 3 月 18 日
Amazon RDS 计划弃用 Oracle Database 18c	Oracle Database 18c (18.0.0.0) 将被弃用。在支持结束日期之后，Oracle Corporation 将不再为 Oracle Database 18c 提供补丁。Amazon RDS 计划于 2021 年 7 月 1 日开始自动将 Oracle Database 18c 实例升级到 Oracle Database 19c。在自动升级开始之前，我们强烈建议您将现有 Oracle Database 18c 实例手动升级到 Oracle Database 19c。有关更多信息，请参阅准备 Oracle Database 18c 的自动升级。	2021 年 3 月 11 日
Amazon RDS 已结束对 Oracle Database 11g 的支持	您只能为 Oracle Database 12c 第 1 版 (12.1.0.2) 和更高版本创建数据库实例。如果您有 Oracle 数据库 11g 快照，请将其升级到更高版本。有关详细信息，请参阅升级 Oracle 数据库快照。	2021 年 3 月 11 日
Amazon RDS 支持在 Amazon Backup 中持续备份数据库实例	现在，您可以在 Amazon Backup 中创建自动备份，并将数据库实例从这些备份还原到指定的时间。有关更多信息，请参阅使用 Amazon Backup 管理自动备份。	2021 年 3 月 10 日

Amazon RDS 支持 Oracle Management Agent (OMA) 版本 13.4	您可以将 Oracle Management Agent (OMA) 版本 13.4 与 Oracle Enterprise Manager (OEM) Cloud Control 13c Release 4 Update 9 结合使用。Amazon RDS for Oracle 会安装 OMA，后者随后可与您的 Oracle Management Service (OMS) 通信以提供监控信息。如果运行 OMS 13.4，可以通过安装 OMA 13.4 来管理数据库。有关更多信息，请参阅适用于 Enterprise Manager Cloud Control 的 Oracle Management Agent。	2021 年 3 月 10 日
RDS Proxy 端点增强	您可以创建与每个 RDS 代理关联的其他端点。在不同 VPC 中创建端点可实现代理的跨 VPC 访问。Aurora MySQL 集群的代理还可以有只读端点。这些读取器端点连接到集群中的读取器数据库实例，可提高查询密集型应用程序的读取可扩展性和可用性。有关 RDS 代理的更多信息，请参阅 Amazon RDS 用户指南或 Aurora 用户指南中的“使用 Amazon RDS 代理管理连接”。	2021 年 3 月 8 日
Amazon RDS 扩展了对跨区域自动备份的支持	您现在可以将运行 PostgreSQL 的 Amazon RDS 数据库实例配置为将数据库快照和事务日志复制到其他Amazon区域。有关更多信息，请参阅将自动备份复制到另一个 Amazon 区域。	2021 年 3 月 8 日

中国（北京）区域和中国（宁夏）区域支持 Amazon RDS for MariaDB 和 MySQL 复制筛选条件	中国（北京）地区和中国（宁夏）地区现支持复制筛选条件。有关更多信息，请参阅使用 MariaDB 配置复制筛选条件和使用 MySQL 配置复制筛选条件。	2021 年 3 月 5 日
Amazon RDS 支持选择加入区域中的跨区域数据库快照副本	现在，您可以向选择加入的 Amazon 区域或从其中复制数据库快照。有关更多信息，请参阅跨 Amazon 区域复制快照。	2021 年 3 月 4 日
Amazon RDS for SQL Server 支持标准版的 Always On 可用性组	当您使用 SQL Server 2019 标准版数据库引擎上的多可用区配置创建数据库实例时，RDS 会自动使用可用性组。有关更多信息，请参阅 Microsoft SQL Server 的多可用区部署。	2021 年 2 月 23 日
Amazon RDS for Oracle 引入了与顾问相关的过程	`rdsadmin_util` 程序包中包含过程 `advisor_task_set_parameter` 、`advisor_task_drop` 和 `dbms_stats_init` 。您可以使用这些过程修改、停止和重新启用顾问任务，例如 `AUTO_STATS_ADVISOR_TASK` 。有关更多信息，请参阅设置顾问任务的参数。	2021 年 2 月 23 日
Amazon RDS 为多可用区数据库实例提供失效转移原因	现在，当多可用区数据库实例故障转移到备用副本时，可以看到更详细的说明。有关更多信息，请参阅 Amazon RDS的故障转移流程。	2021 年 2 月 18 日

Amazon RDS 扩展了对将快照导出到 Amazon S3 的支持	现在，您可以将数据库快照数据导出到 中国 中的 Amazon S3。有关更多信息，请参阅将数据库快照数据导出到 Amazon S3。	2021 年 2 月 17 日
Amazon RDS for MariaDB 和 MySQL 的复制筛选条件	您可以为 MySQL 和 MariaDB 实例配置复制筛选条件。复制筛选条件指定在只读副本中复制的数据库和数据表。您可以为每个只读副本创建包含或排除的数据库和表的列表。有关更多信息，请参阅使用 MariaDB 配置复制筛选条件和使用 MySQL 配置复制筛选条件。	2021 年 2 月 12 日
RDS for Oracle 支持 Oracle APEX 20.2v1	您可以将 Oracle APEX 20.2.v1 与所有受支持的 Oracle Database 版本一起使用。有关更多信息，请参阅 Oracle Application Express。	2021 年 2 月 2 日
Amazon RDS for SQL Server 支持 tempdb 数据库的本地实例存储	现在，您可以在 Amazon EC2 db.r5d 和 db.m5d 实例类型上启动 Amazon RDS for SQL Server，并将 tempdb 数据库配置为使用实例存储。与基于 Amazon EBS 的标准存储相比，将 tempdb 数据文件和日志文件放在本地可实现更低的读写延迟。有关更多信息，请参阅 Amazon RDS for SQL Server 上 的 tempdb 数据库的实例存储支持。	2021 年 1 月 27 日

Amazon RDS for PostgreSQL 支持 pg_partman and pg_cron	Amazon RDS for PostgreSQL 现在支持 pg_partman 和 pg_cron 扩展。有关 pg_partman 扩展的更多信息，请参阅使用 pg_partman 扩展管理 PostgreSQL 分区。有关 pg_cron 扩展的更多信息，请参阅使用 PostgreSQL pg_cron 扩展计划维护。	2021 年 1 月 12 日
Amazon RDS 支持将 Oracle Management Agent 日志发布到 Amazon CloudWatch Logs	Oracle Management Agent 日志包括 emctl.log、emdctlj.log、gcagent.log、gcagent_errors.log、emagent.nohup 和 secure.log。Amazon RDS 将这些日志中的每个日志作为单独的 CloudWatch 日志流发布。有关更多信息，请参阅将 Oracle 日志发布到 Amazon CloudWatch Logs。	2020 年 12 月 28 日
Amazon RDS on Amazon Outposts 支持其他数据库版本	RDS on Outposts 现支持其他 MySQL 和 PostgreSQL 版本。有关更多信息，请参阅 Amazon RDS on Amazon Outposts 对 Amazon RDS 特征的支持。	2020 年 12 月 23 日
Amazon RDS on Amazon Outposts 支持 CoIP	Outposts 上的 RDS 现在支持客户拥有的 IP 地址 (COIP)。COIP 通过您的本地网络提供连接到 Outpost 子网中资源的本地或外部连接。有关更多信息，请参阅适用于 Outposts 上 RDS 的客户拥有的 IP 地址。	2020 年 12 月 22 日

Amazon RDS for Oracle 计划将 11g BYOL 实例升级到 19c	我们计划在 2021 年 1 月 4 日开始自动将自带许可证 (BYOL) 模式下的所有 Oracle Database 11g 实例版本升级到 Oracle Database 19c。所有 Oracle Database 11g 实例（包括预留实例）将移到最新的可用版本更新 (RU)。有关更多信息，请参阅为 Oracle Database 11g BYOL 的自动升级做准备。	2020 年 12 月 11 日
Amazon RDS 支持将自动备份复制到另一个 Amazon 区域	现在，您现在可以将 Amazon RDS 数据库实例配置为将快照和事务日志复制到您选择的目标Amazon区域。有关更多信息，请参阅将自动备份复制到另一个 Amazon 区域。	2020 年 12 月 4 日
Amazon RDS for Oracle 和 Microsoft SQL Server 支持新的数据库实例类	现在，您可以使用 db.r5b 实例类来创建运行 Oracle 或 SQL Server 的 Amazon RDS 数据库实例。有关更多信息，请参阅数据库实例类支持的数据库引擎。	2020 年 12 月 4 日
支持 MariaDB 10.2.32	您现在可以创建运行 MariaDB 版本 10.2.32 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MariaDB 版本。	2020 年 11 月 25 日

Amazon RDS for SQL Server 支持 SQL Server 2019 上的 Microsoft 商业智能套件	现在，您可以在数据库实例上运行最新版本的 SQL Server Analysis Services、SQL Server Integration Services 和 SQL Server Reporting Services。有关详细信息，请参阅适用于 Microsoft SQL Server 数据库引擎的选项。	2020 年 11 月 24 日
数据库预览环境中的 Amazon RDS for PostgreSQL 版本 13	在数据库预览环境中，Amazon RDS for PostgreSQL 现在支持 PostgreSQL 版本 13。有关更多信息，请参阅 PostgreSQL 版本 13。	2020 年 11 月 24 日
Amazon RDS 性能详情引入了新的维度	您可根据数据库（PostgreSQL、MySQL 和 MariaDB）、应用程序（PostgreSQL）和会话类型（PostgreSQL）的维度组对数据库负载进行分组。此外，Amazon RDS 还支持维度 db.name（PostgreSQL、MySQL 和 MariaDB）、db.application.name（PostgreSQL）和 db.session_type.name（PostgreSQL）。有关详细信息，请参阅首要负载表。	2020 年 11 月 24 日
Amazon RDS for MariaDB 支持新的主要版本	您现在可以创建运行 MariaDB 版本 10.5 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MariaDB 版本。	2020 年 11 月 23 日

支持 MySQL 5.6.49	您现在可以创建运行 MySQL 版本 5.6.49 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2020 年 11 月 20 日
支持 MySQL 5.5.62	您现在可以创建运行 MySQL 版本 5.5.62 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2020 年 11 月 20 日
性能详情支持分析正在运行的 PostgreSQL 查询的统计数据	现在可以使用 PostgreSQL 数据库实例的性能详情分析正在运行的查询的统计数据。有关详细信息，请参阅 PostgreSQL 的统计数据。	2020 年 11 月 18 日
Amazon RDS 扩展了对存储自动扩展功能的支持	现在，您可以在创建只读副本、将数据库实例恢复到指定时间或从 Amazon S3 备份还原 MySQL 数据库实例时启用存储自动扩展功能。有关详细信息，请参阅使用 Amazon RDS 存储自动扩展功能自动管理容量。	2020 年 11 月 18 日
Amazon RDS for SQL Server 支持数据库邮件	借助 Database Mail，您可以从您的 Amazon RDS for SQL Server 数据库实例发送电子邮消息。指定电子邮件收件人之后，您可以向所发送的消息添加文件或查询结果。有关详细信息，请参阅在 Amazon RDS for SQL Server 上使用 Database Mail。	2020 年 11 月 4 日

支持 MySQL 8.0.21	您现在可以创建运行 MySQL 版本 8.0.21 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2020 年 10 月 22 日
Amazon RDS 扩展了对将快照导出到 Amazon S3 的支持	现在，您可以在所有商业 Amazon区域中将数据库快照数据导出到 Amazon S3。有关更多信息，请参阅将数据库快照数据导出到 Amazon S3。	2020 年 10 月 22 日
Amazon RDS for PostgreSQL 支持只读副本升级	对于 Amazon RDS for PostgreSQL，当您执行主数据库实例的主要版本升级时，只读副本也会自动升级。有关更多信息，请参阅升级 PostgreSQL 数据库引擎。	2020 年 10 月 15 日
Amazon RDS for MariaDB、MySQL 和 PostgreSQL 支持 Graviton2 数据库实例类	现在，您可以使用 Graviton2 数据库实例类 db.m6g.x 和 db.r6g.x 来创建运行 MariaDB、MySQL 或 PostgreSQL 的 Amazon RDS 数据库实例。有关更多信息，请参阅所有可用数据库实例类支持的数据库引擎。	2020 年 10 月 15 日
Amazon RDS for SQL Server 支持升级到 SQL Server 2019	您可以将 SQL Server 数据库实例升级到 SQL Server 2019。有关更多信息，请参阅升级 Microsoft SQL Server 数据库引擎。	2020 年 10 月 6 日

Amazon RDS for Oracle 支持指定国家/地区字符集	国家/地区字符集也称为 NCHAR 字符集，在 NCHAR、NVARCHAR2 和 NCLOB 数据类型中使用。创建数据库时，可以指定 AL16UTF16（默认）或 UTF8 作为 NCHAR 字符集。有关更多信息，请参阅 Amazon RDS 中支持的 Oracle 字符集。	2020 年 10 月 2 日
支持 MySQL 5.7.31	您现在可以创建运行 MySQL 版本 5.7.31 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2020 年 10 月 1 日
Amazon RDS for PostgreSQL 支持将数据导出到 Amazon S3	您可以从 PostgreSQL 数据库实例中查询数据，并将数据直接导出到存储在 Amazon S3 存储桶中的文件中。有关更多信息，请参阅将数据从 RDS for PostgreSQL 数据库实例导出到 Amazon S3。	2020 年 9 月 24 日
Amazon RDS for MySQL 8.0 支持 Percona XtraBackup	现在，您可以使用 Percona XtraBackup 将备份还原到 Amazon RDS for MySQL 8.0 数据库实例中。有关更多信息，请参阅将备份还原到 MySQL 数据库实例。	2020 年 9 月 17 日
Amazon RDS for SQL Server 支持使用只读副本在数据库实例上进行本机备份和还原	您可以将 SQL Server 本机备份还原到已配置只读副本的数据库实例上。有关更多信息，请参阅导入和导出 SQL Server 数据库。	2020 年 9 月 16 日

Amazon RDS for SQL Server 支持更多时区	您可以将数据库实例时区与您选择的时区相匹配。有关更多信息，请参阅 Microsoft SQL Server 数据库实例的本地时区。	2020 年 9 月 11 日
数据库预览环境中的 Amazon RDS for PostgreSQL 版本 13 测试版 3	在数据库预览环境中，Amazon RDS for PostgreSQL 现在支持 PostgreSQL 版本 13 测试版 3。有关更多信息，请参阅 PostgreSQL 版本 13。	2020 年 9 月 9 日
Amazon RDS for SQL Server 支持跟踪标志 692	您现在可以使用数据库参数组启用跟踪标志 692 作为启动参数。启用此跟踪标志会在将数据批量加载到堆或集群化索引时禁用快速插入。有关更多信息，请参阅在批量加载期间禁用快速插入。	2020 年 8 月 27 日
Amazon RDS for SQL Server 支持 Microsoft SQL Server 2019	您现在可以创建使用 SQL Server 2019 的 RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 Microsoft SQL Server 版本。	2020 年 8 月 26 日
RDS for Oracle 支持装载的副本数据库	创建或修改 Oracle 副本时，可以将其置于装载模式。由于副本数据库不接受用户连接，因此它无法支持只读工作负载。装载副本会在应用存档重做日志文件后删除这些文件。装载副本的主要用途是跨区域灾难恢复。有关更多信息，请参阅 Oracle 副本概述。	2020 年 8 月 13 日

RDS for Oracle 计划升级 11g SE1 LI 实例	2020 年 11 月 1 日，我们计划开始自动将包含 Oracle Database 11g SE1 许可证 (LI) 的实例升级到适用于 Amazon RDS for Oracle 的 Oracle Database 19c。所有 11g 实例（包括预留实例）将移到最新的可用 Oracle 版本更新 (RU)。有关更多信息，请参阅准备 Oracle Database 11g SE1 的自动升级。	2020 年 7 月 31 日
Amazon RDS 在 PostgreSQL 和 MySQL 的预览版本中支持新的 Graviton2 数据库实例类	您现在可以创建运行 PostgreSQL 或 MySQL 的 Amazon RDS 数据库实例，这些数据库实例使用 db.m6g.x 和 db.r6g.x 数据库实例类。有关更多信息，请参阅所有可用数据库实例类支持的数据库引擎。	2020 年 7 月 30 日
RDS for Oracle 支持 Oracle APEX 20.1v1	您可以将 Oracle APEX 20.1v1 与所有受支持的 Oracle Database 版本一起使用。有关更多信息，请参阅 Oracle Application Express。	2020 年 7 月 28 日
支持 MySQL 8.0.20	您现在可以创建运行 MySQL 版本 8.0.20 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2020 年 7 月 23 日

Amazon RDS for MariaDB 和 Amazon RDS for MySQL 支持新的数据库实例类	您现在可以创建运行 MariaDB 和 MySQL 的 Amazon RDS 数据库实例，这些数据库实例使用 db.m5.16xlarge、db.m5.8xlarge、db.r5.16xlarge 和 db.r5.8xlarge 数据库实例类。有关更多信息，请参阅所有可用数据库实例类支持的数据库引擎。	2020 年 7 月 23 日
RDS for SQL Server 支持禁用旧版本 TLS 和密码	您可以打开或关闭某些安全协议和密码。有关更多信息，请参阅配置安全协议和密码。	2020 年 7 月 21 日
RDS 支持 SE2 上的 Oracle Spatial	您可以将标准版 2 (SE2) 中的 Oracle Spatial 用于 12.2、18c 和 19c 的所有版本。有关更多信息，请参阅 Oracle Spatial。	2020 年 7 月 9 日
Amazon RDS 支持 Amazon PrivateLink	Amazon RDS 现支持为 Amazon RDS API 调用创建 Amazon VPC 端点，以在 Amazon网络中保持应用程序和 Amazon RDS 之间的流量。有关更多信息，请参阅 Amazon RDS 和接口 VPC 端点（Amazon PrivateLink）。	2020 年 7 月 9 日
Amazon RDS for PostgreSQL 版本 9.4.x 已经终止支持。	Amazon RDS for PostgreSQL 不再支持版本 9.4.x。有关支持的版本，请参阅支持的 PostgreSQL 数据库版本。	2020 年 7 月 8 日

支持 MariaDB 10.3.23 和 10.4.13	您现在可以创建运行 MariaDB 版本 10.3.23 和 10.4.13 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MariaDB 版本。	2020 年 7 月 6 日
Amazon RDS on Amazon Outposts	您可以在 Amazon Outposts 上创建 Amazon RDS 数据库实例。有关更多信息，请参阅使用 Amazon RDS on Amazon Outposts。	2020 年 7 月 6 日
Amazon RDS for Oracle 自动创建清单文件	为了向 BYOL 客户提出服务请求，Oracle 支持部门会请求由 Opatch 生成的清单文件。Amazon RDS for Oracle 每小时自动在 BDUMP 目录中创建清单文件。有关更多信息，请参阅访问 Opatch 文件。	2020 年 7 月 6 日
支持 MySQL 5.7.30 和 5.6.48	您现在可以创建运行 MySQL 版本 5.7.30 和 5.6.48 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2020 年 6 月 25 日
Amazon RDS for Oracle 支持 ADRCI	自动诊断存储库命令解释器 (ADRCI) 实用程序是用于管理诊断数据的 Oracle 命令行工具。通过使用 Amazon RDS 程序包 `rdsadmin_adrci_util` 中的功能，您可以列出和打包问题和事件，还可以显示跟踪文件。有关更多信息，请参阅 Oracle 数据库实例的常见 DBA 诊断任务。	2020 年 6 月 17 日

支持 MySQL 8.0.19	您现在可以创建运行 MySQL 版本 8.0.19 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2020 年 6 月 2 日
MySQL 8.0 支持小写表名称	您现在可以为运行 MySQL 8.0.19 版及更高的 8.0 版本的 Amazon RDS 数据库实例将 `lower_case_table_names` 参数设置为 `1`。有关更多信息，请参阅 Amazon RDS 数据库实例的 MySQL 参数异常。	2020 年 6 月 2 日
Amazon RDS for Microsoft SQL Server 支持 SQL Server Integration Services (SSIS)	SSIS 是一个用于数据集成和工作流应用程序的平台。您可以在现有的或新的数据库实例上启用 SSIS。它安装在与您的数据库引擎相同的数据库实例上。有关更多信息，请参阅 SQL Server 中的 SQL Server Integration Services 支持。	2020 年 5 月 19 日
Amazon RDS for Microsoft SQL Server 支持 SQL Server Reporting Services (SSRS)	SSRS 是一个基于服务器的应用程序，用于生成和分发报告。您可以在现有或新数据库实例上启用 SSRS。它安装在与您的数据库引擎相同的数据库实例上。有关更多信息，请参阅 SQL Server 中的 SQL Server Reporting Services 支持。	2020 年 5 月 15 日

Amazon RDS for Microsoft SQL Server 支持多可用区实例上的 S3 集成	现在，您可以将 Amazon S3 与 SQL Server 特征一起使用，例如在多可用区数据库实例上的批量插入。有关更多信息，请参阅将 Amazon RDS for SQL Server 数据库实例与 Amazon S3 集成。	2020 年 5 月 15 日
Amazon RDS for Oracle 支持清除回收站	`rdsadmin.rdsadmin_util.purge_dba_recyclebin` 过程清除回收站。有关更多信息，请参阅清除回收站。	2020 年 5 月 13 日
Amazon RDS for Oracle 改进了 Automatic Workload Repository (AWR) 的可管理性	`rdsadmin.rdsadmin_diagnostic_util` 过程生成 AWR 报告并将 AWR 数据提取到转储文件中。有关更多信息，请参阅使用 Automatic Workload Repository (AWR) 生成性能报告。	2020 年 5 月 13 日
Amazon RDS for Microsoft SQL Server 支持 Microsoft 分布式事务处理协调器 (MSDTC)	Amazon RDS for SQL Server 支持主机之间的分布式事务处理。有关更多信息，请参阅 SQL Server 中的 Microsoft 分布式事务处理协调器支持。	2020 年 5 月 4 日

Amazon RDS for Microsoft SQL Server 支持新版本	您现在可以创建 Amazon RDS 数据库实例，运行 SQL Server 版本 2017 CU19 14.00.3281.6、2016 SP2 CU11 13.00.5598.27、2014 SP3 CU4 12.00.6329.1 和 2012 SP4 GDR 11.0.7493.4 的所有修订版。有关更多信息，请参阅 Amazon RDS 上的 Microsoft SQL Server 版本。	2020 年 4 月 28 日
Amazon RDS 在欧洲（米兰）区域中可用	Amazon RDS 现已在欧洲（米兰）区域中可用。有关更多信息，请参阅区域和可用区。	2020 年 4 月 28 日
Amazon RDS 支持 Local Zones	您现在可以在本地扩展区子网中启动数据库实例。有关更多信息，请参阅区域、可用区和 Local Zones。	2020 年 4 月 23 日
Amazon RDS 在非洲（开普敦）区域中可用	Amazon RDS 现已在非洲（开普敦）区域中可用。有关更多信息，请参阅区域和可用区。	2020 年 4 月 22 日
Amazon RDS for Microsoft SQL Server 支持 SQL Server Analysis Services (SSAS)	SSAS 是一个在线分析处理 (OLAP) 和数据挖掘工具，安装在 SQL Server 中。您可以在现有或新数据库实例上启用 SSAS。它安装在与您的数据库引擎相同的数据库实例上。有关更多信息，请参阅支持 SQL Server 中的 SQL Server Analysis Services。	2020 年 4 月 17 日

适用于 PostgreSQL 的 Amazon RDS Proxy	Amazon RDS 代理现在可用于 PostgreSQL。您可以使用 RDS 代理减少数据库实例上连接管理的开销，并降低“连接过多”错误的可能性。RDS 代理目前提供针对 PostgreSQL 的公共预览版。有关更多信息，请参阅使用 Amazon RDS 代理管理连接（预览版）。	2020 年 4 月 8 日
Amazon RDS for Oracle 支持 Oracle APEX 版本 19.2.v1	Amazon RDS for Oracle 现在支持 Oracle Application Express (APEX) 19.2.v1 版。有关更多信息，请参阅 Oracle Application Express。	2020 年 4 月 8 日
Amazon RDS for MariaDB 支持新的主要版本	您现在可以创建运行 MariaDB 版本 10.4 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MariaDB 版本。	2020 年 4 月 6 日
Amazon RDS 性能详情可用于 Amazon RDS for MariaDB 10.4	Amazon RDS 性能详情现在可供 Amazon RDS for MariaDB 版本 10.4 使用。有关更多信息，请参阅使用 Amazon RDS 性能详情。	2020 年 4 月 6 日
Amazon RDS for PostgreSQL 版本 9.3.x 已经终止支持	Amazon RDS for PostgreSQL 不再支持版本 9.3.x。有关支持的版本，请参阅支持的 PostgreSQL 数据库版本。	2020 年 4 月 3 日
Amazon RDS for Microsoft SQL Server 支持只读副本	您现在可以为 SQL Server 数据库实例创建只读副本。有关更多信息，请参阅使用只读副本。	2020 年 4 月 3 日

Amazon RDS for Microsoft SQL Server 支持多文件备份	您现在可以使用 SQL Server 本机备份和还原将数据库备份到多个文件。有关更多信息，请参阅备份数据库。	2020 年 4 月 2 日
Amazon RDS for Oracle 与 Amazon License Manager 集成	Amazon RDS for Oracle 现已与 Amazon License Manager 集成。如果您使用“自带许可”模型，则 Amazon License Manager 集成可以更轻松地监控您组织内的 Oracle 许可证使用情况。有关更多信息，请参阅与 Amazon License Manager 集成。	2020 年 3 月 23 日
在 Amazon RDS for MariaDB 和 Amazon RDS for MySQL 中，在 db.r5 实例上支持 64 TiB	现在，您可以为 MariaDB 和 MySQL 创建 Amazon RDS 数据库实例，以使用最多带 64 TiB 存储的 db.r5 数据库实例类。有关更多信息，请参阅影响存储性能的因素。	2020 年 3 月 18 日
支持 MySQL 8.0.17	您现在可以创建运行 MySQL 版本 8.0.17 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2020 年 3 月 10 日
Amazon RDS 性能详情可用于 Amazon RDS for MySQL 8.0	Amazon RDS 性能详情现在可用于 Amazon RDS for MySQL 8.0.17 及更高的 8.0 版本。有关更多信息，请参阅使用 Amazon RDS 性能详情。	2020 年 3 月 10 日

支持 MySQL 5.6.46	您现在可以创建运行 MySQL 版本 5.6.46 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2020 年 2 月 28 日
Amazon RDS 性能详情可用于 Amazon RDS for MariaDB 10.3	Amazon RDS 性能详情现在可用于 Amazon RDS for MariaDB 10.3.13 及更高的 10.3 版本。有关更多信息，请参阅使用 Amazon RDS 性能详情。	2020 年 2 月 26 日
支持 MySQL 5.7.28	您现在可以创建运行 MySQL 版本 5.7.28 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2020 年 2 月 20 日
支持 MariaDB 10.3.20	您现在可以创建运行 MariaDB 版本 10.3.20 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MariaDB 版本。	2020 年 2 月 20 日
Amazon RDS for Microsoft SQL Server 支持新的数据库实例类	您现在可以创建运行 SQL Server 并使用 db.z1d 数据库实例类的 Amazon RDS 数据库实例。有关更多信息，请参阅 Microsoft SQL Server 的数据库实例类支持。	2020 年 2 月 19 日

在 Amazon RDS for SQL Server 中支持跨账户、跨 VPC 的 Active Directory 域	Amazon RDS for Microsoft SQL Server 现在支持将数据库实例与不同账户和 VPC 所拥有的 Active Directory 域相关联。有关更多信息，请参阅将 Windows 身份验证与 Microsoft SQL Server 数据库实例结合使用。	2020 年 2 月 13 日
Oracle OLAP 选项	Amazon RDS for Oracle 现在支持 Oracle 数据库实例的在线分析处理 (OLAP) 选项。您可以使用 Oracle OLAP，根据 OLAP 标准创建维度对象和多维数据集来分析大量数据。有关更多信息，请参阅 Oracle OLAP。	2020 年 2 月 13 日
Oracle 支持 FIPS 140-2	Amazon RDS for Oracle 支持针对 SSL/TLS 连接的美国联邦信息处理标准出版物 140-2 (FIPS 140-2)。有关更多信息，请参阅 FIPS 支持。	2020 年 2 月 11 日
Amazon RDS for PostgreSQL 支持新的数据库实例类	您现在可以创建运行 PostgreSQL 的 Amazon RDS 数据库实例，这些数据库实例使用 db.m5.16xlarge、db.m5.8xlarge、db.r5.16xlarge 和 db.r5.8xlarge 数据库实例类。有关更多信息，请参阅所有可用数据库实例类支持的数据库引擎。	2020 年 2 月 11 日

性能详情支持分析正在运行的 MariaDB 和 MySQL 查询的统计数据	现在可以使用适用于 MariaDB 和 MySQL 数据库实例的性能详情分析正在运行的查询的统计数据。有关更多信息，请参阅分析正在运行的查询的统计信息。	2020 年 2 月 4 日
支持将数据库快照数据导出到 Amazon S3 for MariaDB、MySQL 和 PostgreSQL	Amazon RDS 支持将数据库快照数据导出到 Amazon S3 for MariaDB、MySQL 和 PostgreSQL。有关更多信息，请参阅将数据库快照数据导出到 Amazon S3。	2020 年 1 月 23 日
Amazon RDS for MySQL 支持 Kerberos 身份验证	现在，当用户连接到 Amazon RDS for MySQL 数据库实例时，您可以使用 Kerberos 身份验证来验证用户的身份。有关更多信息，请参阅对 MySQL 使用 Kerberos 身份验证。	2020 年 1 月 21 日
Amazon RDS 性能详情支持查看 Amazon RDS for Microsoft SQL Server 的更多 SQL 文本	Amazon RDS 性能详情现在支持在 Amazon RDS for Microsoft SQL Server DB 数据库实例的性能详情控制面板中查看更多 SQL 文本。有关更多信息，请参阅在性能详情控制面板中查看更多 SQL 文本。	2019 年 12 月 17 日

Amazon RDS 代理	使用 Amazon RDS 代理可以减少集群上连接管理的开销，并降低“连接过多”错误的可能性。请将每个代理与一个 RDS 数据库实例或 Aurora 数据库集群进行关联。然后，请在应用程序的连接字符串中使用代理端点。Amazon RDS 代理当前是公开预览版。它支持 RDS for MySQL 数据库引擎。有关更多信息，请参阅使用 Amazon RDS 代理管理连接（预览版）。	2019 年 12 月 3 日
Amazon Outposts 上的 Amazon RDS（预览版）	使用 Amazon RDS on Amazon Outposts，您可在本地数据中心中创建Amazon托管式关系数据库。使用 RDS on Outposts，用户可在 Amazon Outposts 上运行 RDS 数据库。有关更多信息，请参阅 Amazon RDS on Amazon Outposts（预览版）。	2019 年 12 月 3 日
Amazon RDS for Oracle 支持跨区域只读副本	Amazon RDS for Oracle 现在支持使用 Active Data Guard 的跨区域只读副本。有关更多信息，请参阅使用只读副本和使用 Oracle 只读副本。	2019 年 11 月 26 日
性能详情支持分析正在运行的 Oracle 查询的统计数据	现在，您可以使用性能详情分析正在对 Oracle 数据库实例运行的查询的统计信息。有关更多信息，请参阅分析正在运行的查询的统计信息。	2019 年 11 月 25 日

Amazon RDS for Microsoft SQL Server 支持将日志发布到 CloudWatch Logs	您可以将 Amazon RDS for SQL Server 数据库实例配置为将日志事件直接发布到 Amazon CloudWatch Logs。有关更多信息，请参阅将 SQL Server 日志发布到 Amazon CloudWatch Logs。	2019 年 11 月 25 日
Amazon RDS for Microsoft SQL Server 支持新的数据库实例类	您现在可以创建运行 SQL Server 并使用 db.x1e 和 db.x1 数据库实例类的 Amazon RDS 数据库实例。有关更多信息，请参阅 Microsoft SQL Server 的数据库实例类支持。	2019 年 11 月 25 日
Amazon RDS for Microsoft SQL Server 支持差异和日志还原	您可以使用 SQL Server 本机备份和还原来还原差异备份和日志。有关更多信息，请参阅使用本机备份和还原。	2019 年 11 月 25 日
Amazon RDS for Microsoft SQL Server 在新区域中支持多可用区	SQL Server 上的多可用区现在可在中国、中东（巴林）和欧洲（斯德哥尔摩）中使用。有关更多信息，请参阅 Microsoft SQL Server 的多可用区部署。	2019 年 11 月 22 日
Amazon RDS for Microsoft SQL Server 现在支持批量插入和 S3 集成	您可以在 SQL Server 数据库实例和 Amazon S3 存储桶之间传输文件。然后，您可以结合使用 Amazon S3 和 SQL Server 特征，例如批量插入。有关更多信息，请参阅将 Amazon RDS for SQL Server 数据库实例与 Amazon S3 集成。	2019 年 11 月 21 日

适用于 Amazon RDS for Microsoft SQL Server 的性能详情计数器	现在，您可以将性能计数器添加到 Microsoft SQL Server 数据库实例的性能详情图表。有关更多信息，请参阅 Amazon RDS for Microsoft SQL Server 的性能详情计数器。	2019 年 11 月 12 日
Amazon RDS for Microsoft SQL Server 支持新的数据库实例类大小	您现在可以创建运行 SQL Server 并使用 db.m5 和 db.r5 数据库实例类的 8xlarge 和 16xlarge 实例大小的 Amazon RDS 数据库实例。包括小至 2xlarge 的实例大小现在可用于 db.t3 实例类。有关更多信息，请参阅 Microsoft SQL Server 的数据库实例类支持。	2019 年 11 月 11 日
支持 PostgreSQL 快照升级	如果您有 Amazon RDS PostgreSQL 数据库实例的现有手动数据库快照，现在则可以将其升级到更高版本的 PostgreSQL 数据库引擎。有关更多信息，请参阅升级 PostgreSQL 数据库快照。	2019 年 11 月 7 日
Amazon RDS for Oracle 支持新的主要版本	您现在可以创建运行 Oracle Database 19c (19.0) 的 Amazon RDS 数据库实例。有关更多信息，请参阅适用于 Amazon RDS 的 Oracle Database 19c。	2019 年 11 月 7 日

数据库预览环境中的 Amazon RDS for PostgreSQL 版本 12.0	在数据库预览环境中，Amazon RDS for PostgreSQL 现在支持 PostgreSQL 版本 12.0。有关更多信息，请参阅数据库预览环境中的 PostgreSQL 版本 12.0。	2019 年 11 月 1 日
Amazon RDS for PostgreSQL 支持 Kerberos 身份验证	在用户连接到运行 PostgreSQL 的 Amazon RDS 数据库实例时，您现在可以使用 Kerberos 身份验证对用户进行身份验证。有关更多信息，请参阅在 Amazon RDS for PostgreSQL 中使用 Kerberos 身份验证。	2019 年 10 月 28 日
Oracle 数据库实例的 OEM Management Agent 数据库任务	Amazon RDS for Oracle 数据库实例现在支持使用一些过程，以便在 Management Agent 上调用某些 EMCTL 命令。有关更多信息，请参阅 OEM Agent 数据库任务。	2019 年 10 月 24 日
Amazon RDS for PostgreSQL 支持 PostgreSQL 可传输数据库	PostgreSQL 可传输数据库提供了一种非常快的方法，以便在两个数据库实例之间迁移 RDS PostgreSQL 数据库。有关更多信息，请参阅在数据库实例之间传输 PostgreSQL 数据库。	2019 年 10 月 8 日
Amazon RDS for Oracle 支持 Kerberos 身份验证	当用户连接到运行 Oracle 的 Amazon RDS 数据库实例时，现在您可以使用 Kerberos 身份验证来验证用户的身份。有关更多信息，请参阅对 Amazon RDS for Oracle 使用 Kerberos 身份验证。	2019 年 9 月 30 日

数据库预览环境中的 Amazon RDS for PostgreSQL 版本 12 测试版 3	在数据库预览环境中，Amazon RDS for PostgreSQL 现在支持 PostgreSQL 版本 12 测试版 3。有关更多信息，请参阅数据库预览环境中的 Amazon RDS 上的 PostgreSQL 版本 12 测试版 3。	2019 年 8 月 28 日
支持 MySQL 8.0.16	您现在可以创建运行 MySQL 版本 8.0.16 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2019 年 8 月 19 日
Amazon RDS for Oracle 支持新的主要版本	您现在可以创建运行 Oracle Database 18c (18.0) 的 Amazon RDS 数据库实例。有关更多信息，请参阅适用于 Amazon RDS 的 Oracle Database 18c。	2019 年 8 月 15 日
适用于 OEM 13c 版本 3 的 Management Agent	Amazon RDS for Oracle 数据库实例现在支持适用于 Oracle Enterprise Manager (OEM) Cloud Control 13c 版本 3 的 Management Agent。有关更多信息，请参阅适用于 Enterprise Manager Cloud Control 的 Oracle Management Agent。	2019 年 8 月 7 日
数据库预览环境中的 Amazon RDS for PostgreSQL 版本 12 测试版 2	在数据库预览环境中，Amazon RDS for PostgreSQL 现在支持 PostgreSQL 版本 12 测试版 2。有关更多信息，请参阅数据库预览环境中的 Amazon RDS 上的 PostgreSQL 版本 12 测试版 2。	2019 年 8 月 6 日

Amazon RDS 支持 SQL Server 的服务器排序规则	Amazon RDS for SQL Server 支持新数据库实例的一系列排序规则。有关更多信息，请参阅 Microsoft SQL Server 的排序规则和字符集。	2019 年 7 月 29 日
Amazon RDS for Oracle 支持 Oracle APEX 版本 19.1.v1	Amazon RDS for Oracle 现在支持 Oracle Application Express (APEX) 19.1.v1 版。有关更多信息，请参阅 Oracle Application Express。	2019 年 6 月 28 日
数据库预览环境中的 Amazon RDS for PostgreSQL 版本 13 测试版 1	在数据库预览环境中，Amazon RDS for PostgreSQL 现在支持 PostgreSQL 版本 13 测试版 1。有关更多信息，请参阅 PostgreSQL 版本 13。	2019 年 6 月 22 日
Amazon RDS 存储自动扩展	利用 Amazon RDS 数据库实例的存储扩展，Amazon RDS 可自动扩展与数据库实例关联的存储，以减小出现空间不足情况的可能性。有关存储自动扩展的信息，请参阅使用 Amazon RDS 数据库实例的存储。	2019 年 6 月 20 日
Amazon RDS for Oracle 支持 db.z1d 数据库实例类	您现在可以创建运行 Oracle 并使用 db.z1d 数据库实例类的 Amazon RDS 数据库实例。有关更多信息，请参阅数据库实例类。	2019 年 6 月 13 日

Amazon RDS 性能详情支持查看 Amazon RDS for Oracle 的更多 SQL 文本	Amazon RDS 性能详情现在支持在 Amazon RDS for Oracle 数据库实例的性能详情控制面板中查看更多 SQL 文本。有关更多信息，请参阅在性能详情控制面板中查看更多 SQL 文本。	2019 年 6 月 10 日
Amazon RDS 将对本机还原 SQL Server 数据库的支持增加到最大 16 TB	您现在可以执行从 SQL Server 到 Amazon RDS 的最多 16 TB 的本机还原。有关更多信息，请参阅 Amazon RDS for SQL Server：限制和建议。	2019 年 6 月 4 日
Amazon RDS 增加对 Microsoft SQL Server 审计的支持	凭借 Amazon RDS for Microsoft SQL Server，您可以使用 SQL Server Audit 来审核服务器和数据库级事件，以及查看您的数据库实例的结果或将审核日志文件直接发送至 Amazon S3。有关更多信息，请参阅 SQL Server Audit。	2019 年 5 月 23 日
改进 Amazon RDS 建议	Amazon RDS 已为数据库资源改进其自动建议。例如，Amazon RDS 现在为数据库参数据供建议。有关更多信息，请参阅使用 Amazon RDS 建议。	2019 年 5 月 22 日
Amazon RDS for SQL Server 的每个数据库实例支持更多数据库	对于运行在 Microsoft SQL Server 上的每个数据库实例，您最多可以创建 30 个数据库。有关更多信息，请参阅 Microsoft SQL Server 数据库实例的限制。	2019 年 5 月 21 日

为 Amazon RDS for MariaDB、MySQL 和 PostgreSQL 支持 64 TiB 和 80k IOPS 的存储	您现在可为 MariaDB、MySQL 和 PostgreSQL 创建最多具有 64 TiB 存储和 80000 个预调配 IOPS 的 Amazon RDS 数据库实例。有关更多信息，请参阅数据库实例存储。	2019 年 5 月 20 日
Amazon RDS for MySQL 支持升级预检查	当您将数据库实例从 MySQL 5.7 升级到 MySQL 8.0 时，Amazon RDS 执行预检查以确定是否存在不兼容性。有关更多信息，请参阅从 MySQL 5.7 升级到 8.0 的预检查。	2019 年 5 月 17 日
支持 MySQL 密码验证插件	现在您可以使用 MySQL `validate_password` 插件来增强 Amazon RDS for MySQL 数据库实例的安全性。有关更多信息，请参阅使用密码验证插件。	2019 年 5 月 16 日
Amazon RDS for Oracle 的性能详情计数器	现在，您可以将性能计数器添加到 Oracle 数据库实例的性能详情图表。有关更多信息，请参阅 Amazon RDS for Oracle 的性能详情计数器。	2019 年 5 月 8 日
支持每秒计费	现在，Amazon RDS 在除 Amazon GovCloud（美国）以外的所有Amazon区域对按需实例按 1 秒增量计费。有关更多信息，请参阅 Amazon RDS 的数据库实例计费。	2019 年 4 月 25 日

支持从 Amazon S3 为 Amazon RDS for PostgreSQL 导入数据	现在，您可以将数据从 Amazon S3 文件导入至 RDS PostgreSQL 数据库实例中的表。有关更多信息，请参阅将 Amazon S3 数据导入至 RDS PostgreSQL 数据库实例。	2019 年 4 月 24 日
支持从 Amazon S3 还原 5.7 备份	您现在可以创建 MySQL 5.7 数据库的备份，将其存储在 Amazon S3 上，然后将该备份文件还原到运行 MySQL 的新 Amazon RDS 数据库实例上。有关更多信息，请参阅将备份还原到 MySQL 数据库实例。	2019 年 4 月 17 日
支持 Amazon RDS for PostgreSQL 的多个主要版本升级	利用 Amazon RDS for PostgreSQL，现在您升级数据库引擎时可以从多个主要版本中选择。此特征让您可以在升级所选 PostgreSQL 引擎版本时跳至更新的主要版本。有关更多信息，请参阅升级 PostgreSQL 数据库引擎。	2019 年 4 月 16 日
为 Amazon RDS for Oracle 支持 64 TiB 存储	您现在可为 Oracle 创建最多具有 64 TiB 存储和 80000 个预调配 IOPS 的 Amazon RDS 数据库实例。有关更多信息，请参阅数据库实例存储。	2019 年 4 月 4 日
支持 MySQL 8.0.15	您现在可以创建运行 MySQL 版本 8.0.15 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2019 年 4 月 3 日

支持 MariaDB 10.3.13	您现在可以创建运行 MariaDB 版本 10.3.13 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MariaDB 版本。	2019 年 4 月 3 日
Amazon RDS 上的 Microsoft SQL Server 2008 R2 已经终止支持	Microsoft SQL Server 2008 R2 已终止支持，与 Microsoft 计划于 2019 年 7 月 9 日结束对此版本的延长支持一致。从 2019 年 6 月 1 日开始，任何现有 Microsoft SQL Server 2008 R2 快照会自动升级到 Microsoft SQL Server 2012 的最新次要版本。有关更多信息，请参阅 Amazon RDS 上的 Microsoft SQL Server 2008 R2 支持。	2019 年 4 月 2 日
Microsoft SQL Server 2017 中支持 Always On 可用性组	现在您可以在 SQL Server 2017 企业版 14.00.3049.1 或更高版本中使用 Always On 可用性组。有关更多信息，请参阅 Microsoft SQL Server 的多可用区部署。	2019 年 3 月 29 日
查看卷指标	现在您可以查看 Amazon Elastic Block Store (Amazon EBS) 卷的指标，这些卷是用于存储数据库和日志的物理设备。有关更多信息，请参阅查看增强监控。	2019 年 3 月 20 日
支持 MySQL 5.7.25	您现在可以创建运行 MySQL 版本 5.7.25 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2019 年 3 月 19 日

Amazon RDS for Oracle 支持 RMAN DBA 任务	Amazon RDS for Oracle 现在支持 Oracle Recovery Manager (RMAN) DBA 任务，包括 RMAN 备份。有关更多信息，请参阅 Oracle 数据库实例的常见 DBA Recovery Manager (RMAN) 任务。	2019 年 3 月 14 日
Amazon RDS for PostgreSQL 支持版本 11.1	您现在可以创建运行 PostgreSQL 11.1 版的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 PostgreSQL 版本 11.1。	2019 年 3 月 12 日
多文件还原在 Amazon RDS for SQL Server 中可用	现在您可以使用 Amazon RDS for SQL Server 从多个文件还原。有关更多信息，请参阅还原数据库。	2019 年 3 月 11 日
MariaDB 10.2.21	您现在可以创建运行 MariaDB 版本 10.2.21 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MariaDB 版本。	2019 年 3 月 11 日
Amazon RDS for Oracle 支持只读副本	Amazon RDS for Oracle 现在支持使用 Active Data Guard 的只读副本。有关更多信息，请参阅使用只读副本和使用 Oracle 只读副本。	2019 年 3 月 11 日
Amazon RDS 性能详情可供 Amazon RDS for MariaDB 使用	Amazon RDS 性能详情现在可供 Amazon RDS for MariaDB 使用。有关更多信息，请参阅使用 Amazon RDS 性能详情。	2019 年 3 月 11 日

MySQL 8.0.13 和 5.7.24	您现在可以创建运行 MySQL 8.0.13 和 5.7.24 版的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2019 年 3 月 8 日
Amazon RDS 性能详情可供 Amazon RDS for SQL Server 使用	Amazon RDS 性能详情现在可供 Amazon RDS for SQL Server 使用。有关更多信息，请参阅使用 Amazon RDS 性能详情。	2019 年 3 月 4 日
Amazon RDS for Oracle 支持 Amazon S3 集成	现在您可以在 Amazon RDS for Oracle DB 实例和 Amazon S3 存储桶之间传输文件。有关更多信息，请参阅将 Amazon RDS for Oracle 与 Amazon S3 集成。	2019 年 2 月 26 日
Amazon RDS for MySQL 和 Amazon RDS for MariaDB 支持 db.t3 数据库实例类	您现在可以创建运行 MySQL 或 MariaDB 并使用 db.t3 数据库实例类的 Amazon RDS 数据库实例。有关更多信息，请参阅数据库实例类。	2019 年 2 月 20 日
Amazon RDS for MySQL 和 Amazon RDS for MariaDB 支持 db.r5 数据库实例类	您现在可以创建运行 MySQL 或 MariaDB 并使用 db.r5 数据库实例类的 Amazon RDS 数据库实例。有关更多信息，请参阅数据库实例类。	2019 年 2 月 20 日
RDS for MySQL 和 PostgreSQL 的性能详情计数器	现在，您可以将性能计数器添加到 MySQL 和 PostgreSQL 数据库实例的性能详情图表中。有关更多信息，请参阅性能详情控制面板组件。	2019 年 2 月 19 日

Amazon RDS for PostgreSQL 现在支持自适应 Autovacuum 参数调整	使用 Amazon RDS for PostgreSQL 进行自适应 Autovacuum 参数调整，通过自动调整 Autovacuum 参数，可帮助防止事务 ID 重叠。有关更多信息，请参阅降低事务 ID 重叠的可能性。	2019 年 2 月 12 日
Amazon RDS for Oracle 支持 Oracle APEX 版本 18.1.v1 和 18.2.v1	Amazon RDS for Oracle 现在支持 Oracle Application Express (APEX) 18.1.v1 和 18.2.v1 版。有关更多信息，请参阅 Oracle Application Express。	2019 年 2 月 11 日
Amazon RDS 性能详情支持查看 RDS for MySQL 的更多 SQL 文本	Amazon RDS 性能详情现在支持在 MySQL 数据库实例的性能详情控制面板中查看更多 SQL 文本。有关更多信息，请参阅在性能详情控制面板中查看更多 SQL 文本。	2019 年 2 月 6 日
Amazon RDS for PostgreSQL 支持 db.t3 数据库实例类	您现在可以创建运行 PostgreSQL 并使用 db.t3 数据库实例类的 Amazon RDS 数据库实例。有关更多信息，请参阅数据库实例类。	2019 年 1 月 25 日
Amazon RDS for Oracle 支持 db.t3 数据库实例类	您现在可以创建运行 Oracle 并使用 db.t3 数据库实例类的 Amazon RDS 数据库实例。有关更多信息，请参阅数据库实例类。	2019 年 1 月 25 日

Amazon RDS 性能详情支持查看 Amazon RDS PostgreSQL 的更多 SQL 文本	Amazon RDS 性能详情现在支持在 Amazon RDS PostgreSQL 数据库实例的性能详情控制面板中查看更多 SQL 文本。有关更多信息，请参阅在性能详情控制面板中查看更多 SQL 文本。	2019 年 1 月 24 日
Amazon RDS for Oracle 支持新版本的 SQLT	Amazon RDS for Oracle 现在支持 SQLT 12.2.180725 版。有关更多信息，请参阅 Oracle SQLT。	2019 年 1 月 22 日
Amazon RDS for PostgreSQL 支持 db.r5 数据库实例类	您现在可以创建运行 PostgreSQL 并使用 db.r5 数据库实例类的 Amazon RDS 数据库实例。有关更多信息，请参阅数据库实例类。	2018 年 12 月 19 日
Amazon RDS for PostgreSQL 现在支持受限制的密码管理	Amazon RDS for PostgreSQL 允许您使用参数 `rds.restrict_password_commands` 和角色 `rds_password` 限制哪些用户可以管理用户密码和密码失效时间更改。有关更多信息，请参阅限制密码管理。	2018 年 12 月 19 日
Amazon RDS for PostgreSQL 支持将数据库日志上传到 Amazon CloudWatch Logs	Amazon RDS for PostgreSQL 支持将数据库日志上传到 CloudWatch Logs。有关更多信息，请参阅将 PostgreSQL 日志发布到 CloudWatch Logs。	2018 年 12 月 10 日

Amazon RDS for Oracle 支持 db.r5 数据库实例类	您现在可以创建运行 Oracle 并使用 db.r5 数据库实例类的 Amazon RDS 数据库实例。有关更多信息，请参阅数据库实例类。	2018 年 11 月 20 日
在删除数据库实例时保留备份	Amazon RDS 支持在您删除数据库实例时保留自动备份。有关更多信息，请参阅使用备份。	2018 年 11 月 15 日
Amazon RDS for PostgreSQL 支持 db.m5 数据库实例类	您现在可以创建运行 PostgreSQL 并使用 db.m5 数据库实例类的 Amazon RDS 数据库实例。有关更多信息，请参阅数据库实例类。	2018 年 11 月 15 日
Amazon RDS for Oracle 支持新的主要版本	您现在可以创建运行 Oracle 版本 12.2 的 Amazon RDS 数据库实例。	2018 年 11 月 13 日
Amazon RDS for SQL Server 支持 Always On	Amazon RDS for SQL Server 支持 Always On 可用性组。有关更多信息，请参阅 Microsoft SQL Server 的多可用区部署。	2018 年 11 月 8 日
Amazon RDS for PostgreSQL 支持使用自定义 DNS 服务器的出站网络访问	Amazon RDS for PostgreSQL 支持使用自定义 DNS 服务器的出站网络访问。有关更多信息，请参阅将自定义 DNS 服务器用于出站网络访问。	2018 年 11 月 8 日
Amazon RDS for MariaDB、MySQL 和 PostgreSQL 支持 32 TiB 的存储	您现在可以为 MySQL、MariaDB 和 PostgreSQL 创建最多具有 32 TiB 存储的 Amazon RDS 数据库实例。有关更多信息，请参阅数据库实例存储。	2018 年 11 月 7 日

Amazon RDS for Oracle 支持扩展数据类型	您现在可以在运行 Oracle 的 Amazon RDS 数据库实例上启用扩展数据类型。对于扩展数据类型，VARCHAR2、NVARCHAR2 和 RAW 数据类型的最大大小为 32,767 字节。有关更多信息，请参阅使用扩展数据类型。	2018 年 11 月 6 日
Amazon RDS for Oracle 支持 db.m5 数据库实例类	您现在可以创建运行 Oracle 并使用 db.m5 数据库实例类的 Amazon RDS 数据库实例。有关更多信息，请参阅数据库实例类。	2018 年 11 月 2 日
将 Amazon RDS for Oracle 从 SE、SE1 或 SE2 迁移到 EE	您现在可以从任何 Oracle Database 标准版（SE、SE 1 或 SE2）迁移到 Oracle Database 企业版 (EE)。有关更多信息，请参阅在 Oracle 版本之间迁移。	2018 年 10 月 31 日
Amazon RDS 现在可以停止多可用区实例	Amazon RDS 现在可以停止属于多可用区部署的数据库实例。以前，停止实例特征在多可用区实例中具有限制。有关更多信息，请参阅暂时停止 Amazon RDS 数据库实例。	2018 年 10 月 29 日
Amazon RDS 性能详情可用于 Amazon RDS for Oracle	Amazon RDS 性能详情现在可用于 Amazon RDS for Oracle。有关更多信息，请参阅使用 Amazon RDS 性能详情。	2018 年 10 月 29 日

在数据库预览环境中，Amazon RDS for PostgreSQL 支持 PostgreSQL 版本 11	在数据库预览环境中，Amazon RDS for PostgreSQL 现在支持 PostgreSQL 11 版。有关更多信息，请参阅数据库预览环境中的 Amazon RDS 上的 PostgreSQL 11 版。	2018 年 10 月 25 日
MySQL 支持新的主要版本	您现在可以创建运行 MySQL 8.0 版的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2018 年 10 月 23 日
MariaDB 支持新的主要版本	您现在可以创建运行 MariaDB 版本 10.3 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MariaDB 版本。	2018 年 10 月 23 日
Amazon RDS for Oracle 支持 Oracle JVM	Amazon RDS for Oracle 现在支持 Oracle Java 虚拟机 (JVM) 选项。有关更多信息，请参阅 Oracle Java 虚拟机。	2018 年 10 月 16 日
用于还原和时间点恢复的自定义参数组	您现在可以在还原快照或执行时间点恢复操作时指定自定义参数组。有关更多信息，请参阅从数据库快照还原和将数据库实例还原到指定的时间。	2018 年 10 月 15 日
Amazon RDS for Oracle 支持 32 TiB 存储	您现在可以创建最多具有 32 TiB 存储的 Oracle RDS 数据库实例。有关更多信息，请参阅数据库实例存储。	2018 年 10 月 15 日

Amazon RDS for MySQL 支持 GTID	Amazon RDS for MySQL 现在支持全局事务标识符 (GTID)，它们在所有数据库实例和复制配置中是唯一的。有关更多信息，请参阅将基于 GTID 的复制用于 RDS for MySQL。	2018 年 10 月 10 日
MySQL 5.7.23、5.6.41 和 5.5.61	您现在可以创建运行 MySQL 5.7.23、5.6.41 和 5.5.61 版的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2018 年 10 月 8 日
Amazon RDS for Oracle 支持新版本的 SQLT	Amazon RDS for Oracle 现在支持 SQLT 12.2.180331 版。有关更多信息，请参阅 Oracle SQLT。	2018 年 10 月 4 日
Amazon RDS for PostgreSQL 现在支持 IAM 身份验证	Amazon RDS for PostgreSQL 现在支持 IAM 身份验证。有关更多信息，请参阅在 MySQL 和 PostgreSQL 中使用 IAM 数据库身份验证。	2018 年 9 月 27 日
您可以为 Amazon RDS 数据库实例启用删除保护	如果为数据库实例启用删除保护，则任何用户都无法删除数据库。有关更多信息，请参阅删除数据库实例。	2018 年 9 月 26 日
Amazon RDS for MySQL 和 Amazon RDS for MariaDB 支持 db.m5 数据库实例类	您现在可以创建运行 MySQL 或 MariaDB 并使用 db.m5 数据库实例类的 Amazon RDS 数据库实例。有关更多信息，请参阅数据库实例类。	2018 年 9 月 18 日

Amazon RDS 现在支持升级到 SQL Server 2017	您可以将现有数据库实例从 SQL Server 2008 之外的任何版本升级到 SQL Server 2017。要从 SQL Server 2008 升级，请先升级到其他版本之一。有关信息，请参阅升级 Microsoft SQL Server 数据库引擎。	2018 年 9 月 11 日
在数据库预览环境中，Amazon RDS for PostgreSQL 现在支持 PostgreSQL 版本 11 测试版 3	在此版本中，预写日志 (WAL) 段大小 (wal_segment_size) 现在设置为 64MB。有关 PostgreSQL 版本 11 Beta 3 的更多信息，请参阅 PostgreSQL 11 Beta 3 已发布。有关数据库预览环境的信息，请参阅使用数据库预览环境。	2018 年 9 月 7 日
Amazon Aurora 用户指南	Amazon Aurora 用户指南介绍了所有 Amazon Aurora 概念并提供了关于通过控制台和命令行界面使用各种特征的说明。Amazon RDS 用户指南 现在涵盖非 Aurora 数据库引擎。	2018 年 8 月 31 日
Amazon RDS 性能详情可用于 RDS for MySQL	Amazon RDS 性能详情现在可用于 RDS for MySQL。有关更多信息，请参阅使用 Amazon RDS 性能详情。	2018 年 8 月 28 日
Aurora PostgreSQL 兼容版现在支持 Aurora Auto Scaling	Aurora 副本的 Auto Scaling 现在可用于Aurora PostgreSQL 兼容版。有关更多信息，请参阅 Amazon Aurora Auto Scaling 与 Aurora 副本结合使用。	2018 年 8 月 16 日

Aurora Serverless for Aurora MySQL	Aurora Serverless 是一种适用于 Amazon Aurora 的按需自动扩展配置。有关更多信息，请参阅使用 Amazon Aurora Serverless。	2018 年 8 月 9 日
MySQL 5.7.22 和 5.6.40	您现在可以创建运行 MySQL 版本 5.7.22 和 5.6.40 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MySQL 版本。	2018 年 8 月 6 日
Aurora 现已在中国（宁夏）区域可用	Aurora MySQL 和 Aurora PostgreSQL 现已在中国（宁夏）区域推出。有关更多信息，请参阅 Amazon Aurora MySQL 的可用性和 Amazon Aurora PostgreSQL 的可用性。	2018 年 8 月 6 日
Amazon RDS for MySQL 支持延迟复制	Amazon RDS for MySQL 现在支持延迟复制作为灾难恢复策略。有关更多信息，请参阅使用 MySQL 配置延迟复制。	2018 年 8 月 6 日
Amazon RDS 性能详情可用于 Aurora MySQL	Amazon RDS 性能详情现在可用于 Aurora MySQL。有关更多信息，请参阅使用 Amazon RDS 性能详情。	2018 年 8 月 6 日
Amazon RDS 性能详情与 Amazon CloudWatch 的集成	Amazon RDS 性能详情会自动将指标发布到 Amazon CloudWatch。有关更多信息，请参阅发布到 CloudWatch 的性能详情指标。	2018 年 8 月 6 日

Amazon RDS 建议	Amazon RDS 现在可为数据库资源提供自动建议。有关更多信息，请参阅使用 Amazon RDS 建议。	2018 年 7 月 25 日
跨 Amazon 区域的增量快照复制	Amazon RDS 对未加密和已加密实例均支持跨Amazon区域的增量快照复制。有关更多信息，请参阅跨Amazon区域复制快照。	2018 年 7 月 24 日
Amazon RDS 性能详情可用于 Amazon RDS for PostgreSQL	Amazon RDS 性能详情现在可供 Amazon RDS for PostgreSQL 使用。有关更多信息，请参阅使用 Amazon RDS 性能详情。	2018 年 7 月 18 日
Amazon RDS for Oracle 支持 Oracle APEX 版本 5.1.4.v1	Amazon RDS for Oracle 现在支持 Oracle Application Express (APEX) 5.1.4.v1 版。有关更多信息，请参阅 Oracle Application Express。	2018 年 7 月 10 日
Amazon RDS for Oracle 支持将日志发布到 Amazon CloudWatch Logs	Amazon RDS for Oracle 现在支持将警报、审核、跟踪和侦听器日志数据发布到 CloudWatch Logs 中的日志组。有关更多信息，请参阅将 Oracle 日志发布到 Amazon CloudWatch Logs。	2018 年 7 月 9 日
MariaDB 10.2.15、10.1.34 和 10.0.35	您现在可以创建运行 MariaDB 版本 10.2.15、10.1.34 和 10.0.35 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MariaDB 版本。	2018 年 5 月 7 日

Aurora PostgreSQL 1.2 已推出并与 PostgreSQL 9.6.8 兼容	Aurora PostgreSQL 1.2 已推出并与 PostgreSQL 9.6.8 兼容。有关更多信息，请参阅版本 1.2。	2018 年 27 月 6 日
Amazon RDS PostgreSQL 的只读副本支持多可用区部署	现在，Amazon RDS PostgreSQL 的 RDS 只读副本支持多可用区。有关更多信息，请参阅使用 PostgreSQL 只读副本。	2018 年 25 月 6 日
性能详情可用于 Aurora PostgreSQL	性能详情通常可用于 Aurora PostgreSQL，它支持长时间保留性能数据。有关更多信息，请参阅使用 Amazon RDS 性能详情。	2018 年 6 月 21 日
Aurora PostgreSQL 已在美国西部（加利福尼亚北部）区域推出	Aurora PostgreSQL 现已在美国西部（加利福尼亚北部）区域推出。有关更多信息，请参阅 Amazon Aurora PostgreSQL 的可用性。	2018 年 6 月 11 日
Amazon RDS for Oracle 现在支持 CPU 配置	Amazon RDS for Oracle 支持为数据库实例类的处理器配置 CPU 内核数和每个内核的线程数。有关更多信息，请参阅配置数据库实例类的处理器。	2018 年 5 月 6 日

早期更新

下表描述了 2018 年 6 月之前每次发布 Amazon RDS 用户指南 时进行的重要更改。

更改	描述	更改日期
在数据库预览环境中，Amazon RDS	PostgreSQL 版本 11 测试版 1 包含一些改进，如 PostgreSQL 11 测试版 1 已发布！中所述。	2018 年 5 月 31 日

更改	描述	更改日期
for PostgreSQL 现在支持 PostgreSQL 版本 11 测试版 1	有关数据库预览环境的信息，请参阅使用数据库预览环境。	
Amazon RDS for Oracle 现在支持 TLS 版本 1.0 和 1.2	Amazon RDS for Oracle 支持传输层安全性 (TLS) 版本 1.0 和 1.2。有关更多信息，请参阅“Oracle SSL 选项的 TLS 版本”。	2018 年 5 月 30 日
Aurora MySQL 支持将日志发布到 Amazon CloudWatch Logs	Aurora MySQL 现在支持将一般、慢速、审核和错误日志数据发布到 CloudWatch Logs 中的日志组。有关更多信息，请参阅将 Aurora MySQL 发布到 CloudWatch Logs。	2018 年 5 月 23 日
Amazon RDS PostgreSQL 的数据库预览环境	在预览模式下，您可以立即启动 Amazon RDS PostgreSQL 的新实例。有关数据库预览环境的更多信息，请参阅使用数据库预览环境。	2018 年 5 月 22 日
Amazon RDS for Oracle 数据库实例支持新的数据库实例类	Oracle 数据库实例现在支持 db.x1e 和 db.x1 数据库实例类。有关更多信息，请参阅“数据库实例类”和“RDS for Oracle 数据库实例类”。	2018 年 5 月 22 日
Amazon RDS PostgreSQL 现在支持对只读副本执行 postgres_fdw。	您现在可以使用 postgres_fdw 从只读副本连接到远程服务器。有关更多信息，请参阅“使用 postgres_fdw 扩展访问外部数据”。	2018 年 5 月 17 日
Amazon RDS for Oracle 现在支持设置 sqlnet.ora 参数	您现在可以使用 Amazon RDS for Oracle 设置 sqlnet.ora 参数。有关更多信息，请参阅“使用 sqlnet.ora 参数修改连接属性”。	2018 年 5 月 10 日
Aurora PostgreSQL 现已在亚太 (首尔) 区域提供。	Aurora PostgreSQL 现已在亚太 (首尔) 区域提供。有关更多信息，请参阅 Amazon Aurora PostgreSQL 的可用性。	2018 年 5 月 9 日

更改	描述	更改日期
Aurora MySQL 支持回溯	Aurora MySQL 现在支持将数据库集群"倒回"到特定时间，而无需从备份还原数据。有关更多信息，请参阅回溯 Aurora 数据库集群。	2018 年 5 月 9 日
Aurora MySQL 支持从外部 MySQL 进行加密迁移和复制	目前，Aurora MySQL 支持从外部 MySQL 数据库进行加密迁移和复制。有关更多信息，请参阅从外部 MySQL 数据库将数据迁移到 Amazon Aurora MySQL 数据库集群和Aurora 与 MySQL 之间或 Aurora 与其他 Aurora 数据库集群之间的复制。	2018 年 4 月 25 日
Aurora PostgreSQL 兼容版 支持写入时复制协议。	现在，您可以克隆 Aurora PostgreSQL 数据库集群中的数据库。有关更多信息，请参阅克隆 Aurora 数据库集群中的数据库。	2018 年 4 月 10 日
MariaDB 10.2.12、10.1.31 和 10.0.34	您现在可以创建运行 MariaDB 版本 10.2.12、10.1.31 和 10.0.34 的 Amazon RDS 数据库实例。有关更多信息，请参阅"Amazon RDS 上的 MariaDB 版本"。	2018 年 3 月 21 日
新区域的 Aurora PostgreSQL 支持	Aurora PostgreSQL 现已在欧洲 (伦敦) 和亚太 (新加坡) 区域可用。有关更多信息，请参阅 Amazon Aurora PostgreSQL 的可用性。	2018 年 3 月 13 日
MySQL 5.7.21、5.6.39 和 5.5.59	您现在可以创建运行 MySQL 版本 5.7.21、5.6.39 和 5.5.59 的 Amazon RDS 数据库实例。有关更多信息，请参阅"Amazon RDS 上的 MySQL 版本"。	2018 年 3 月 9 日
Amazon RDS for Oracle 现在支持 Oracle REST Data Services	Amazon RDS for Oracle 支持将 Oracle REST Data Services 作为 APEX 选项的一部分。有关更多信息，请参阅"Oracle Application Express (APEX)"。	2018 年 3 月 9 日
Amazon Aurora MySQL 兼容版在新的Amazon区域中可用。	Aurora MySQL 现已在亚太地区 (新加坡) 区域提供。有关 Aurora MySQL 的Amazon区域的完整列表，请参阅 Amazon Aurora MySQL 的可用性。	2018 年 3 月 6 日

更改	描述	更改日期
运行 Microsoft SQL Server 的 Amazon RDS 数据库实例支持更改数据捕获 (CDC)	运行 Amazon RDS for Microsoft SQL Server 的数据库实例现在支持更改数据捕获 (CDC)。有关更多信息，请参阅“适用于 Microsoft SQL Server 数据库实例的更改数据捕获支持”。	2018 年 2 月 6 日
Aurora MySQL 支持新主要版本	您现在可以创建运行 MySQL 版本 5.7 的 Aurora MySQL 数据库集群。有关更多信息，请参阅 Amazon Aurora MySQL 数据库引擎更新 2018-02-06。	2018 年 2 月 6 日
将 MySQL 和 MariaDB 日志发布到 Amazon CloudWatch Logs	您现在可以将 MySQL 和 MariaDB 日志数据发布到 CloudWatch Logs。有关更多信息，请参阅“将 MySQL 日志发布到 Amazon CloudWatch Logs”和“将 MariaDB 日志发布到 Amazon CloudWatch Logs”。	2018 年 1 月 17 日
只读副本的多可用区支持	您现可将只读副本创建为多可用区数据库实例。Amazon RDS 会在另一个可用区中创建您的副本的备用，用于支持副本的失效转移。创建您的只读副本作为多可用区数据库实例与源数据库是否为多可用区数据库实例无关。有关更多信息，请参阅“使用数据库实例只读副本”。	2018 年 1 月 11 日
Amazon RDS for MariaDB 支持新主要版本	您现在可以创建运行 MariaDB 版本 10.2 的 Amazon RDS 数据库实例。有关更多信息，请参阅 Amazon RDS 上的 MariaDB 10.2 支持。	2018 年 1 月 3 日
Amazon Aurora PostgreSQL 兼容版在新的Amazon 区域中可用。	Aurora PostgreSQL 现已在欧洲（巴黎）区域中提供。有关 Aurora PostgreSQL Amazon区域的完整列表，请参阅 Amazon Aurora PostgreSQL 可用性。	2017 年 12 月 22 日
Aurora PostgreSQL 支持新实例类型	Aurora PostgreSQL 现在支持新实例类型。有关实例类型的完整列表，请参阅选择数据库实例类。	2017 年 12 月 20 日

更改	描述	更改日期
Amazon Aurora MySQL 兼容版在新的Amazon区域中可用。	Aurora MySQL 现已在欧洲（巴黎）区域中提供。有关 Aurora MySQL 的Amazon区域的完整列表，请参阅 Amazon Aurora MySQL 的可用性。	2017 年 12 月 18 日
Aurora MySQL 支持哈希联接	在需要使用 equijoin 联接大量数据时，该特征可以提高查询性能。有关更多信息，请参阅在 Aurora MySQL 中使用哈希联接。	2017 年 12 月 11 日
Aurora MySQL 支持本机函数以调用 Amazon Lambda 函数	在使用 Aurora MySQL 时，您可以调用本机函数 `lambda_sync` 和 `lambda_async` 。有关更多信息，请参阅从 Amazon Aurora MySQL 数据库集群调用 Lambda 函数。	2017 年 12 月 11 日
已添加 Aurora PostgreSQL HIPAA 资格	Aurora PostgreSQL 现在支持构建符合 HIPAA 标准的应用程序。有关更多信息，请参阅使用 Amazon Aurora PostgreSQL。	2017 年 12 月 6 日
可在其他Amazon区域中使用 Amazon Aurora 与 PostgreSQL 的兼容性功能	现可在四个新的Amazon区域中使用 Amazon Aurora 与 PostgreSQL 的兼容性功能。有关更多信息，请参阅 Amazon Aurora PostgreSQL 的可用性。	2017 年 11 月 22 日
修改运行 Microsoft SQL Server 的 Amazon RDS 数据库实例的存储	您现在可以修改运行 SQL Server 的 Amazon RDS 数据库实例的存储。有关更多信息，请参阅"修改 Amazon RDS 数据库实例"。	2017 年 11 月 21 日
对于基于 Linux 的引擎，Amazon RDS 支持 16 TiB 存储	您现在可以创建最多具有 16 TiB 存储的 MySQL、MariaDB、PostgreSQL 和 Oracle RDS 数据库实例。有关更多信息，请参阅"Amazon RDS 数据库实例存储"。	2017 年 11 月 21 日

更改	描述	更改日期
Amazon RDS 支持快速扩展存储	您现在可以在几分钟内在 MySQL、MariaDB、PostgreSQL 和 Oracle RDS 数据库实例中添加存储。有关更多信息，请参阅“Amazon RDS 数据库实例存储”。	2017 年 11 月 21 日
Amazon RDS 支持 MariaDB 版本 10.1.26 和 10.0.32	您现在可以创建运行 MariaDB 版本 10.1.26 和 10.0.32 的 Amazon RDS 数据库实例。有关更多信息，请参阅“Amazon RDS 上的 MariaDB 版本”。	2017 年 11 月 20 日
Amazon RDS for Microsoft SQL Server 现在支持新的数据库实例类	您现在可以创建运行 SQL Server 并使用 db.r4 和 db.m4.16xlarge 数据库实例类的 Amazon RDS 数据库实例。有关更多信息，请参阅“Microsoft SQL Server 的数据库实例类支持”。	2017 年 11 月 20 日
Amazon RDS for MySQL 和 Amazon RDS for MariaDB 现在支持新的数据库实例类	您现在可以创建运行 MySQL 和 MariaDB 并使用 db.r4、db.m4.16xlarge、db.t2.xlarge 和 db.t2.2xlarge 数据库实例类的 Amazon RDS 数据库实例。有关更多信息，请参阅“数据库实例类”。	2017 年 11 月 20 日
SQL Server 2017	您现在可以创建运行 Microsoft SQL Server 2017 的 Amazon RDS 数据库实例。您还可以创建运行 SQL Server 2016 SP1 CU5 的数据库实例。有关更多信息，请参阅“Amazon RDS for Microsoft SQL Server”。	2017 年 11 月 17 日
从 Amazon S3 中还原 MySQL 备份	您现在可以创建本地数据库的备份，将其存储在 Amazon S3 上，然后将该备份文件还原到运行 MySQL 的新 Amazon RDS 数据库实例上。有关更多信息，请参阅“将备份还原到 Amazon RDS for MySQL 数据库实例”。	2017 年 11 月 17 日
具有 Aurora 副本的 Auto Scaling	Amazon Aurora MySQL 现在支持 Aurora Auto Scaling。Aurora Auto Scaling 根据连接或工作负载增加或减少来动态调整 Aurora 副本数。有关更多信息，请参阅 Amazon Aurora Auto Scaling 与 Aurora 副本结合使用。	2017 年 11 月 17 日

更改	描述	更改日期
Oracle 默认版本支持	Amazon RDS for Oracle 数据库实例现在支持为数据库实例设置默认版本。有关更多信息，请参阅"设置数据库实例的默认版本"。	2017 年 11 月 3 日
Oracle 数据库实例文件验证	Amazon RDS for Oracle 数据库实例现已支持通过 Oracle Recovery Manager (RMAN) 逻辑验证实用程序验证数据库实例文件。有关更多信息，请参阅"在 RDS for Oracle 中验证数据库文件 "。	2017 年 11 月 3 日
适用于 OEM 13c 的 Management Agent	Amazon RDS for Oracle 数据库实例现在支持适用于 Oracle Enterprise Manager (OEM) Cloud Control 13c 的管理代理。有关更多信息，请参阅"适用于 Enterprise Manager Cloud Control 的 Oracle Management Agent"。	2017 年 11 月 1 日
Microsoft SQL Server 快照的存储重新配置	现在，将快照还原到运行 Microsoft SQL Server 的 Amazon RDS 数据库实例时，您可以重新配置存储。有关更多信息，请参阅"还原到数据库实例"。	2017 年 10 月 26 日
Aurora MySQL 兼容版 的异步键预取	异步键预取 (AKP) 在需要时提前从内存中预取键，从而提高未缓存索引联接的性能。有关更多信息，请参阅在 Amazon Aurora 中使用异步键预取。	2017 年 10 月 26 日
MySQL 5.7.19、5.6.37 和 5.5.57	您现在可以创建运行 MySQL 版本 5.7.19、5.6.37 和 5.5.57 的 Amazon RDS 数据库实例。有关更多信息，请参阅"Amazon RDS 上的 MySQL 版本"。	2017 年 10 月 25 日
公开发布兼容 PostgreSQL 的 Amazon Aurora	借助于 Amazon Aurora 与 PostgreSQL 的兼容性功能，可以通过简单且经济高效的方式设置、运行和扩展新的和现有的 PostgreSQL 部署，以使您腾出时间专注于业务和应用程序。有关更多信息，请参阅使用 Amazon Aurora PostgreSQL。	2017 年 10 月 24 日

更改	描述	更改日期
Amazon RDS for Oracle 数据库实例支持新的数据库实例类	Amazon RDS for Oracle 数据库实例现已支持内存优化的新一代（db.r4）实例类。Amazon RDS for Oracle 数据库实例现在还支持以下新的当前一代的实例类：db.m4.16xlarge、db.t2.xlarge 和 db.t2.2xlarge。有关更多信息，请参阅"数据库实例类"和"RDS for Oracle 数据库实例类"。	2017 年 10 月 23 日
新特征	现在，在同一个数据库实例类中，您的新的和现有预留实例可以涵盖多种大小。为数据库实例提供了灵活大小的预留实例，这些实例具有相同的 Amazon 区域、数据库引擎和实例系列以及跨可用区配置。为以下数据库引擎提供了灵活大小的预留实例：Amazon Aurora、MariaDB、MySQL、Oracle (自带许可) 和 PostgreSQL。有关更多信息，请参阅"大小灵活的预留数据库实例"。	2017 年 10 月 11 日
新特征	现在您可以使用 Oracle SQLT 选项调整 SQL 语句以获得最优性能。有关更多信息，请参阅"Oracle SQLT"。	2017 年 9 月 22 日
新特征	如果您有 Amazon RDS for Oracle 数据库实例的现有手动数据库快照，现在可以将其升级到更高版本的 Oracle 数据库引擎。有关更多信息，请参阅"升级 Oracle 数据库快照"。	2017 年 9 月 20 日
新特征	您现在可以使用 Oracle Spatial 在运行 Oracle 的 Amazon RDS 数据库实例中存储、检索、更新和查询空间数据。有关更多信息，请参阅"Oracle Spatial"。	2017 年 9 月 15 日
新特征	您现在可以使用 Oracle Locator 支持将基于 Internet 和无线服务的应用程序与基于合作伙伴的 GIS 解决方案与运行 Oracle 的 Amazon RDS 数据库实例结合使用。有关更多信息，请参阅"Oracle Locator"。	2017 年 9 月 15 日
新特征	您现在可以使用 Oracle Multimedia 在运行 Oracle 的 Amazon RDS 数据库实例中存储、管理和检索图像、音频、视频和其他异构媒体数据。	2017 年 9 月 15 日

更改	描述	更改日期
新特征	您现在可以将审核日志从 Amazon Aurora MySQL 数据库集群导出到 Amazon CloudWatch Logs。有关更多信息，请参阅将 Aurora MySQL 日志发布到 Amazon CloudWatch Logs。	2017 年 9 月 14 日
新特征	Amazon RDS 现在支持针对运行 Oracle 的数据库实例的多个版本的 Oracle Application Express (APEX)。有关更多信息，请参阅“Oracle Application Express (APEX)”。	2017 年 9 月 13 日
新特征	您现在可以使用 Amazon Aurora 将未加密或已加密的数据库快照或者 MySQL 数据库实例迁移到已加密的 Aurora MySQL 数据库集群。有关更多信息，请参阅将 RDS for MySQL 快照迁移到 Aurora 和使用 Aurora 只读副本将数据从 MySQL 数据库实例迁移到 Amazon Aurora MySQL 数据库集群。	2017 年 9 月 5 日
新特征	您可以使用 Amazon RDS for Microsoft SQL Server 数据库来构建符合 HIPAA 标准的应用程序。有关更多信息，请参阅“Microsoft SQL Server 数据库实例的合规性计划支持”。	2017 年 8 月 31 日
新特征	您现在可以使用 Amazon RDS for MariaDB 数据库来构建符合 HIPAA 标准的应用程序。有关更多信息，请参阅“Amazon RDS for MariaDB”。	2017 年 8 月 31 日
新特征	您现在可以创建运行 Microsoft SQL Server 且具有多达 16 TiB 的已分配存储空间的 Amazon RDS 数据库实例，并可以创建预置 IOPS 来存储范围 1:1–:50:1。有关更多信息，请参阅“Amazon RDS 数据库实例存储”。	2017 年 8 月 22 日
新特征	您现在可以对在欧洲 (法兰克福) 区域中运行 Microsoft SQL Server 的数据库实例使用多可用区部署。有关更多信息，请参阅“Amazon RDS for Microsoft SQL Server 多可用区部署”。	2017 年 8 月 3 日

更改	描述	更改日期
新特征	您现在可以创建运行 MariaDB 版本 10.1.23 和 10.0.31 的 Amazon RDS 数据库实例。有关更多信息，请参阅“Amazon RDS 上的 MariaDB 版本”。	2017 年 7 月 17 日
新特征	Amazon RDS 现在在所有Amazon区域支持包含许可模式的 Microsoft SQL Server Enterprise Edition。有关更多信息，请参阅“授予 Amazon RDS 上 Microsoft SQL Server 的许可”。	2017 年 7 月 13 日
新特征	Amazon RDS for Oracle 现在支持 Linux 内核大页，提高了数据库扩展能力。使用大页的结果是页表变小，花在内存管理上的 CPU 时间也减少，从而提高大数据库实例的性能。您可以对运行所有版本的 Oracle 12.1.0.2 和 11.2.0.4 版的 Amazon RDS 数据库实例使用大页。有关更多信息，请参阅“为 RDS for Oracle 实例开启大页”。	2017 年 7 月 7 日
新特征	更新为对任何非 Aurora 数据库引擎支持 db.t2.small 和 db.t2.medium 数据库实例类的静态加密 (EAR)。有关更多信息，请参阅“Amazon RDS 加密的可用性”。	2017 年 6 月 27 日
新特征	更新为在欧洲（法兰克福）区域中支持 Amazon Aurora。有关更多信息，请参阅 Amazon Aurora MySQL 的可用性。	2017 年 6 月 16 日
新特征	跨 Amazon 区域复制数据库快照时，现在您可以指定选项组。有关更多信息，请参阅“选项组的注意事项”。	2017 年 6 月 12 日
新特征	现在，您可以跨 Amazon 区域从专用数据库实例复制数据库快照。您可以从使用 Oracle TDE、Microsoft SQL Server TDE 和带镜像的 Microsoft SQL Server 多可用区的数据库实例复制快照。有关更多信息，请参阅“复制数据库快照”。	2017 年 6 月 12 日

更改	描述	更改日期
新特征	Amazon Aurora 现在让您可以快速且经济高效地复制 Amazon Aurora 数据库集群中的所有数据库。有关更多信息，请参阅克隆 Aurora 数据库集群中的数据库。	2017 年 6 月 12 日
新特征	Amazon RDS 现在支持 Microsoft SQL Server 2016 SP1 CU2。有关更多信息，请参阅“Amazon RDS for Microsoft SQL Server”。	2017 年 6 月 7 日
预览版	兼容 PostgreSQL 的 Amazon Aurora 的公开预览版。有关更多信息，请参阅使用 Amazon Aurora PostgreSQL。	2017 年 4 月 19 日
新特征	现在，利用 Amazon Aurora，您几乎可以立即运行 tbl_name ADD COLUMN col_name column_definition 操作。完成该操作无需复制表，对其他 DML 语句也没有重大影响。有关更多信息，请参阅使用快速 DDL 在 Amazon Aurora 中修改表。	2017 年 4 月 5 日
新特征	我们新增了一条监控命令，SHOW VOLUME STATUS，用于显示卷中的节点和磁盘数量。有关更多信息，请参阅显示 Aurora 数据库集群的卷状态。	2017 年 4 月 5 日
新特征	现在，您可以通过 Amazon RDS 在适用于 Oracle 的自定义密码验证函数中使用您的自定义逻辑。有关更多信息，请参阅“创建自定义函数来验证密码”。	2017 年 3 月 21 日
新特征	现在，可以通过 Amazon RDS 访问您的 Oracle 数据库实例的在线和存档重做日志文件。有关更多信息，请参阅“访问在线和存档的重做日志”。	2017 年 3 月 21 日
新特征	您现在可以在相同区域中，在账户之间复制加密和未加密的数据库集群快照。有关更多信息，请参阅跨账户复制数据库集群快照。	2017 年 3 月 7 日
新特征	您现在可以在相同区域中，在账户之间共享加密的数据库集群快照。有关更多信息，请参阅共享数据库集群快照。	2017 年 3 月 7 日

更改	描述	更改日期
新特征	您现在可以复制加密的 Amazon Aurora MySQL 数据库集群来创建跨区域 Aurora 副本。有关更多信息，请参阅跨Amazon区域复制 Aurora MySQL 数据库集群。	2017 年 3 月 7 日
新特征	现在，您可以要求与您的运行 Microsoft SQL Server 的数据库实例的所有连接都使用安全套接字层 (SSL)。有关更多信息，请参阅“将 SSL 与 Microsoft SQL Server 数据库实例结合使用”。	2017 年 2 月 27 日
新特征	现在，您可以将本地时区设置为额外的 15 个时区之一。有关更多信息，请参阅“支持的时区”。	2017 年 2 月 27 日
新特征	现在，您可以使用 Amazon RDS 过程 `msdb.dbo.rds_shrink_tempdbfile` 缩减运行 Microsoft SQL Server 的数据库实例上的临时数据库。有关更多信息，请参阅“缩减临时数据库”。	2017 年 2 月 17 日
新特征	现在，在将企业版和标准版的 Microsoft SQL Server 数据库从 Amazon RDS 数据库实例导出到 Amazon S3 时，您可以压缩您的备份文件。有关更多信息，请参阅“压缩备份文件”。	2017 年 2 月 17 日
新特征	Amazon RDS 现在支持使用自定义 DNS 服务器解析 DNS 名称，这些 DNS 名称用于运行 Oracle 的数据库实例的出站网络访问。有关更多信息，请参阅“设置自定义 DNS 服务器”。	2017 年 1 月 26 日
新特征	Amazon RDS 现在支持在另一区域创建加密只读副本。有关更多信息，请参阅在不同的 Amazon Web Services 区域中创建只读副本和 CreateDBInstanceReadReplica。	2017 年 1 月 23 日
新特征	Amazon RDS 现在支持将 MySQL 数据库快照从 MySQL 5.1 升级到 MySQL 5.5。	2017 年 1 月 20 日

更改	描述	更改日期
新特征	Amazon RDS 现在支持为 MariaDB、MySQL、Oracle、PostgreSQL 和 Microsoft SQL Server 数据库引擎将加密数据库快照复制到另一个区域。有关更多信息，请参阅复制数据库快照和 CopyDBSnapshot。	2016 年 12 月 20 日
新特征	Amazon Aurora MySQL 现在支持空间索引。 空间索引将提高使用空间数据的查询在大型数据集上的查询性能。有关更多信息，请参阅 Amazon Aurora MySQL 和空间数据。	2016 年 12 月 14 日
新特征	Amazon RDS 现在支持对运行 Oracle 的数据库实例进行出站网络访问。您可以使用 utl_http、utl_tcp 和 utl_smtp 从数据库实例连接到网络。有关更多信息，请参阅"使用证书和 Oracle Wallet 配置 UTL_HTTP 访问"。	2016 年 12 月 5 日
新特征	Amazon RDS 停止了对 MySQL 版本 5.1 的支持。但您可以将现有的 MySQL 5.1 快照还原到 MySQL 5.5 实例。有关更多信息，请参阅"支持的 RDS for MySQL 存储引擎"。	2016 年 11 月 15 日
新特征	Amazon RDS 现在支持 Microsoft SQL Server 2016 RTM CU2。有关更多信息，请参阅"Amazon RDS for Microsoft SQL Server"。	2016 年 11 月 4 日
新特征	Amazon RDS 现在支持运行 Oracle 的数据库实例的主要版本升级。您现在可以将 Oracle 数据库实例从 11g 升级到 12c。有关更多信息，请参阅"升级 RDS for Oracle 数据库引擎"。	2016 年 11 月 2 日
新特征	您现在可以创建运行 Microsoft SQL Server 2014 Enterprise Edition 的数据库实例。Amazon RDS 现在支持所有版本和所有区域的 SQL Server 2014 SP2。有关更多信息，请参阅"Amazon RDS for Microsoft SQL Server"。	2016 年 10 月 25 日

更改	描述	更改日期
新特征	Amazon Aurora MySQL 现已集成其他Amazon服务：您可将文本或 XML 数据从 Amazon S3 存储桶加载到表中，或者从数据库代码调用 Amazon Lambda 函数。有关更多信息，请参阅将 Aurora MySQL 与其他Amazon 服务集成。	2016 年 10 月 18 日
新特征	您现在可以在运行 Microsoft SQL Server 的 Amazon RDS 数据库实例上访问临时数据库。您可以通过 Microsoft SQL Server Management Studio (SSMS) 或任何其他标准 SQL 客户端应用程序使用 Transact-SQL 访问临时数据库。有关更多信息，请参阅“访问 Amazon RDS 上 Microsoft SQL Server 数据库实例的临时数据库”。	2016 年 9 月 29 日
新特征	您现在可将 UTL_MAIL 包与运行 Oracle 的 Amazon RDS 数据库实例结合使用。有关更多信息，请参阅“Oracle UTL_MAIL”。	2016 年 9 月 20 日
新特征	您现在可将新 Microsoft SQL Server 数据库实例的时区设置为本地时区以匹配应用程序的时区。有关更多信息，请参阅“Microsoft SQL Server 数据库实例的本地时区”。	2016 年 9 月 19 日
新特征	现在，您可以使用“Oracle Label Security”选项控制对运行 Oracle Database 12c 的 Amazon RDS 数据库实例中的各个表行的访问。借助 Oracle Label Security，您可以通过基于策略的管理模型实施法规合规性，并确保仅具有相应清除级别的用户能访问敏感数据。有关更多信息，请参阅“Oracle Label Security”。	2016 年 9 月 8 日

更改	描述	更改日期
新特征	您现在可使用读取器端点 (将实现跨数据库集群中可用的 Aurora 副本的连接的负载均衡) 连接到 Amazon Aurora 数据库集群。当客户端请求与读取器端点的新连接时，Aurora 将在数据库集群中的 Aurora 副本之间分配连接请求。该功能可帮助平衡跨数据库集群中的多个 Aurora 副本的读取工作负载。有关更多信息，请参阅 Amazon Aurora 端点。	2016 年 9 月 8 日
新特征	现在您可以在运行 Oracle 的 Amazon RDS 数据库实例上支持 Oracle Enterprise Manager Cloud Control。您可以在数据库实例上启用 Management Agent，并与 Oracle Management Service (OMS) 共享数据。有关更多信息，请参阅“适用于 Enterprise Manager Cloud Control 的 Oracle Management Agent”。	2016 年 9 月 1 日
新特征	此版本增加了为资源获取 ARN 的支持。有关更多信息，请参阅“获取 Amazon RDS 的现有 ARN”。	2016 年 8 月 23 日
新特征	您现在可以为每个 Amazon RDS 资源分配最多 50 个标签，以便管理资源和跟踪成本。有关更多信息，请参阅“为 Amazon RDS 资源添加标签”。	2016 年 8 月 19 日
新特征	Amazon RDS 现在支持 Oracle 标准版 2 的“附带许可”模式。有关更多信息，请参阅“创建 Amazon RDS 数据库实例”。 您现在可以更改运行 Microsoft SQL Server 和 Oracle 的 Amazon RDS 数据库实例的许可模式。有关更多信息，请参阅“授予 Amazon RDS 上 Microsoft SQL Server 的许可”和“RDS for Oracle 许可选项”。	2016 年 8 月 5 日

更改	描述	更改日期
新特征	Amazon RDS 现在支持使用完整备份文件 (.bak 文件) 对 Microsoft SQL Server 数据库进行本机备份和还原。您现在可以使用用于存储的 Amazon S3 和用于加密的 Amazon KMS，将 SQL Server 数据库轻松迁移到 Amazon RDS，并在单个便携文件中导入和导出数据库。有关更多信息，请参阅"使用本机备份和还原导入和导出 SQL Server 数据库"。	2016 年 7 月 27 日
新特征	现在可以从 MySQL 数据库中将源文件复制到 Amazon Simple Storage Service (Amazon S3) 存储桶，然后从这些文件还原 Amazon Aurora 数据库集群。该选项可能比使用 `mysqldump` 迁移数据要快得多。有关更多信息，请参阅从外部 MySQL 数据库将数据迁移到 Aurora MySQL 数据库集群。	2016 年 7 月 20 日
新特征	现在可以通过在还原操作期间包含一个 Amazon Key Management Service (Amazon KMS) 加密密钥来还原未加密的 Amazon Aurora 数据库集群快照，从而创建加密的 Amazon Aurora 数据库集群。有关更多信息，请参阅加密 Amazon RDS 资源。	2016 年 6 月 30 日
新特征	您可以使用 Oracle Repository Creation Utility (RCU) 在 Amazon RDS for Oracle 上创建存储库。有关更多信息，请参阅"在 RDS for Oracle 上使用 Oracle Repository Creation Utility "。	2016 年 6 月 17 日
新特征	增加了对 PostgreSQL 跨区域只读副本的支持。有关更多信息，请参阅"在不同的 Amazon Web Services 区域中创建只读副本"。	2016 年 6 月 16 日
新特征	您现在可以使用 Amazon Web Services Managemen t Console轻松地将带镜像的多可用区添加到 Microsoft SQL Server 数据库实例。有关更多信息，请参阅"向 Microsoft SQL Server 数据库实例添加多可用区"。	2016 年 6 月 9 日

更改	描述	更改日期
新特征	您现在可以在以下区域实施使用 SQL Server 镜像的多可用区部署：亚太地区（悉尼）、亚太地区（东京）和南美洲（圣保罗）。有关更多信息，请参阅Amazon RDS for Microsoft SQL Server 多可用区部署。	2016 年 6 月 9 日
新特征	进行了更新，以便支持 MariaDB 版本 10.1。有关更多信息，请参阅“Amazon RDS for MariaDB”。	2016 年 6 月 1 日
新特征	进行了更新，以便支持作为只读副本的 Amazon Aurora 跨区域数据库集群。有关更多信息，请参阅跨Amazon 区域复制 Aurora MySQL 数据库集群。	2016 年 6 月 1 日
新特征	增强监控现在可用于 Oracle 数据库实例。有关更多信息，请参阅“使用增强监控来监控操作系统指标”和“修改 Amazon RDS 数据库实例”。	2016 年 5 月 27 日
新特征	进行了更新，以便支持 Amazon Aurora 数据库集群快照的手动快照共享。有关更多信息，请参阅共享数据库集群快照。	2016 年 5 月 18 日
新特征	您现在可以使用 MariaDB 审核插件来记录 MariaDB 和 MySQL 数据库实例上的数据库活动。有关更多信息，请参阅“适用于 MariaDB 数据库引擎的选项”和“适用于 MySQL 数据库实例的选项”。	2016 年 4 月 27 日
新特征	当前支持从 MySQL 5.6 版到 5.7 版的主要版本就地升级。有关更多信息，请参阅“升级 RDS for MySQL 数据库引擎”。	2016 年 4 月 26 日
新特征	增强监控现在可用于 Microsoft SQL Server 数据库实例。有关更多信息，请参阅“使用增强监控来监控操作系统指标”。	2016 年 4 月 22 日
新特征	更新为在 Amazon RDS 控制台中提供 Amazon Aurora 集群视图。有关更多信息，请参阅查看 Aurora 数据库集群。	2016 年 4 月 1 日

更改	描述	更改日期
新特征	更新为在亚太区域 (首尔)区域中支持带镜像的 SQL Server 多可用区。有关更多信息，请参阅“Amazon RDS for Microsoft SQL Server 多可用区部署”。	2016 年 3 月 31 日
新特征	更新为在亚太区域 (首尔)区域中支持带镜像的 Amazon Aurora 多可用区。有关更多信息，请参阅 Amazon Aurora MySQL 的可用性。	2016 年 3 月 31 日
新特征	PostgreSQL 数据库实例具有需要连接才能使用 SSL 的功能。有关更多信息，请参阅“将 SSL 与 PostgreSQL 数据库实例结合使用”。	2016 年 3 月 25 日
新特征	增强监控现在可用于 PostgreSQL 数据库实例。有关更多信息，请参阅“使用增强监控来监控操作系统指标”。	2016 年 3 月 25 日
新特征	Microsoft SQL Server 数据库实例现在可使用 Windows 身份验证进行用户身份验证。有关更多信息，请参阅“将 Amazon Managed Active Directory 用于 RDS for SQL Server”。	2016 年 3 月 23 日
新特征	增强监控目前已在亚太地区 (首尔) 区域提供。有关更多信息，请参阅“使用增强监控来监控操作系统指标”。	2016 年 3 月 16 日
新特征	您现在可以自定义失效转移期间将 Aurora 副本提升为主实例的顺序。有关更多信息，请参阅 Aurora 数据库集群的容错能力。	2016 年 3 月 14 日
新特征	更新为在迁移到 Aurora 数据库集群时支持加密。有关更多信息，请参阅将数据迁移到 Aurora 数据库集群。	2016 年 3 月 2 日
新特征	更新为支持 Aurora 数据库集群的本地时区。有关更多信息，请参阅 Aurora 数据库集群的本地时区。	2016 年 3 月 1 日
新特征	更新后添加了对当前一代 Amazon RDS 数据库实例类的 MySQL 版本 5.7 的支持。	2016 年 2 月 22 日

更改	描述	更改日期
新特征	更新为在 Amazon GovCloud（美国西部）区域中支持 db.r3 and db.t2 数据库实例。	2016 年 2 月 11 日
新特征	更新为支持加密数据库快照的副本和共享加密的数据库快照。有关更多信息，请参阅“复制 Amazon RDS 的数据库快照”和“共享 Amazon RDS 的数据库快照”。	2016 年 2 月 11 日
新特征	更新为在亚太区域（悉尼）区域中支持 Amazon Aurora。有关更多信息，请参阅 Amazon Aurora MySQL 的可用性。	2016 年 2 月 11 日
新特征	更新为支持面向 Oracle 数据库实例的 SSL。有关更多信息，请参阅“对 RDS for Oracle 数据库实例使用 SSL”。	2016 年 2 月 9 日
新特征	更新为针对 MySQL 和 MariaDB 数据库实例支持本地时区。有关更多信息，请参阅“MySQL 数据库实例的本地时区”和“MariaDB 数据库实例的本地时区”。	2015 年 12 月 21 日
新特征	更新为针对 MySQL 和 MariaDB 实例和 Aurora 数据库集群支持操作系统指标增强监控。有关更多信息，请参阅“在 Amazon RDS 控制台中查看指标”。	2015 年 12 月 18 日
新特征	更新为针对 MySQL 版本 5.5 支持 db.t2、db.r3 和 db.m4 数据库实例类。有关更多信息，请参阅“数据库实例类”。	2015 年 12 月 4 日
新特征	更新为支持修改现有数据库实例的数据库端口。	2015 年 12 月 3 日
新特征	更新为针对 PostgreSQL 实例支持数据库引擎的主要版本升级。有关更多信息，请参阅“升级 RDS for PostgreSQL 数据库引擎”。	2015 年 11 月 19 日
新特征	更新为支持修改现有数据库实例的公开可用性。更新为支持 db.m4 标准数据库实例类。	2015 年 11 月 11 日

更改	描述	更改日期
新特征	更新为支持手动数据库快照共享。有关更多信息，请参阅“共享 Amazon RDS 的数据库快照”。	2015 年 10 月 28 日
新特征	更新为支持 Microsoft SQL Server 2014 的 Web、Express 和 Standard 版本。	2015 年 10 月 26 日
新特征	更新为支持基于 MySQL 的 MariaDB 数据库引擎。有关更多信息，请参阅“Amazon RDS for MariaDB”。	2015 年 10 月 7 日
新特征	更新为在亚太区域（东京）区域中支持 Amazon Aurora。有关更多信息，请参阅 Amazon Aurora MySQL 的可用性。	2015 年 10 月 7 日
新特征	更新为对所有数据库引擎支持具有突增能力的 db.t2 数据库实例，并添加了 db.t2.large 数据库实例类。有关更多信息，请参阅“数据库实例类”。	2015 年 9 月 25 日
新特征	更新为支持 R3 和 T2 数据库实例类上的 Oracle 数据库实例。有关更多信息，请参阅“数据库实例类”。	2015 年 8 月 5 日
新特征	Microsoft SQL Server 企业版现在提供有“附带许可”服务模型。有关更多信息，请参阅“授予 Amazon RDS 上 Microsoft SQL Server 的许可”。	2015 年 7 月 29 日
新特征	Amazon Aurora 已正式发布。Amazon Aurora 数据库引擎支持一个数据库集群中的多个数据库实例。有关详细信息，请参阅什么是 Amazon Aurora？	2015 年 7 月 27 日
新特征	更新为支持将标签复制到数据库快照。	2015 年 7 月 20 日
新特征	经过更新，为所有数据库引擎支持更大的存储，并提高了 SQL Server 的配置的 IOPS。	2015 年 6 月 18 日
新特征	更新了用于预留数据库实例的选项。	2015 年 6 月 15 日
新特征	更新为支持使用 TDE 将 Amazon CloudHSM 与 Oracle 数据库实例结合使用。	2015 年 1 月 8 日

更改	描述	更改日期
新特征	更新为支持加密静态数据和新的 API 版本 (2014 年 10 月 31 日发布)。	2015 年 1 月 6 日
新特征	更新为包含新的 Amazon 数据库引擎：Aurora。Amazon Aurora 数据库引擎支持一个数据库集群中的多个数据库实例。Amazon Aurora 目前为预览版，可能会发生变化。有关详细信息，请参阅什么是 Amazon Aurora？	2014 年 11 月 12 日
新特征	更新为支持 PostgreSQL 只读副本。	2014 年 11 月 10 日
新 API 和特征	更新为支持 GP2 存储类型和新 API 版本 2014-09-01。更新为支持复制现有选项组或参数组以创建新的选项组或参数组的功能。	2014 年 10 月 7 日
新特征	更新为支持运行 MySQL 5.6.19 版和更高版本的数据库实例的 InnoDB 缓存预热。	2014 年 9 月 3 日
新特征	更新为支持在连接到 MySQL 版本 5.6、SQL Server 和 PostgreSQL 数据库引擎时进行 SSL 证书验证。	2014 年 8 月 5 日
新特征	更新为支持具有突增功能的 db.t2 数据库实例类。	2014 年 8 月 4 日
新特征	更新为支持 db.r3 内存优化型数据库实例类，用于 MySQL（版本 5.6）、SQL Server 和 PostgreSQL 数据库引擎。	2014 年 5 月 28 日
新特征	更新为支持使用 SQL Server 镜像进行 SQL Server 多可用区部署。	2014 年 5 月 19 日
新特征	更新为支持从 MySQL 版本 5.5 升级到版本 5.6。	2014 年 4 月 23 日
新特征	更新为支持 Oracle GoldenGate。	2014 年 4 月 3 日
新特征	更新为支持 M3 数据库实例类。	2014 年 2 月 20 日
新特征	更新为支持 Oracle 时区选项。	2014 年 1 月 13 日

更改	描述	更改日期
新特征	更新为支持在不同区域中的 MySQL 数据库实例之间进行复制。	2013 年 11 月 26 日
新特征	更新以支持 PostgreSQL 数据库引擎。	2013 年 11 月 14 日
新特征	更新以支持 SQL Server 透明数据加密 (TDE)。	2013 年 11 月 7 日
新 API 和新增特征	更新以支持跨区域数据库快照副本；新 API 版本，2013-09-09。	2013 年 10 月 31 日
新特征	更新为支持 Oracle Statspack。	2013 年 9 月 26 日
新特征	更新为支持通过复制功能在 Amazon RDS 中运行的 MySQL 实例之间、在本地或 Amazon EC2 上运行的 MySQL 实例之间导入或导出数据。	2013 年 9 月 5 日
新特征	更新为支持 MySQL 5.6 的 db.cr1.8xlarge 数据库实例类。	2013 年 9 月 4 日
新特征	更新为支持复制只读副本。	2013 年 8 月 28 日
新特征	更新为支持并行创建只读副本。	2013 年 7 月 22 日
新特征	更新为支持针对所有 Amazon RDS 资源细化控制的权限和标记。	2013 年 7 月 8 日
新特征	已更新，支持新实例的 MySQL 5.6，包括支持 MySQL 5.6 memcached 接口和二进制日志访问。	2013 年 7 月 1 日
新特征	已更新，支持主要版本从 MySQL 5.1 升级到 MySQL 5.5。	2013 年 6 月 20 日
新特征	已更新数据库参数组，允许对参数值使用表达式。	2013 年 6 月 20 日
新 API 和新增特征	更新为支持只读副本状态；新 API 版本（2013 年 5 月 15 日发布）。	2013 年 5 月 23 日

更改	描述	更改日期
新特征	已更新，支持 Oracle Advanced Security 的本机网络加密和 Oracle 透明数据加密特征。	2013 年 4 月 18 日
新特征	已更新，支持 SQL Server 的主要版本升级和其他配置的 IOPS 功能。	2013 年 3 月 13 日
新特征	已更新，默认支持 RDS 上的 VPC。	2013 年 3 月 11 日
新 API 和特征	已更新，支持日志访问；新 API 版本 2013 - 02 - 12	2013 年 3 月 4 日
新特征	已更新，支持 RDS 事件通知订阅。	2013 年 2 月 4 日
新 API 和特征	已更新，支持数据库实例重命名和 VPC 中的数据库安全组成员迁移到 VPC 安全组。	2013 年 1 月 14 日
新特征	更新为支持 Amazon GovCloud（美国西部）。	2012 年 12 月 17 日
新特征	更新为支持 m1.medium 和 m1.xlarge 数据库实例类。	2012 年 11 月 6 日
新特征	更新为支持只读副本升级。	2012 年 10 月 11 日
新特征	更新为在 Microsoft SQL Server 数据库实例中支持 SSL。	2012 年 10 月 10 日
新特征	更新为支持 Oracle 微数据库实例。	2012 年 9 月 27 日
新特征	更新为支持 SQL Server 2012。	2012 年 9 月 26 日
新 API 和特征	更新以支持配置的 IOPS。API 版本 2012 - 09 - 17。	2012 年 9 月 25 日
新特征	更新了对 VPC 中数据库实例的 SQL Server 支持和对 Data Pump 的 Oracle 支持。	2012 年 9 月 13 日
新特征	更新对 SQL Server Agent 的支持。	2012 年 8 月 22 日
新特征	更新了对数据库实例标记的支持。	2012 年 8 月 21 日

更改	描述	更改日期
新特征	更新了对 VPC 中 Oracle APEX 和 XML 数据库、Oracle 时区和 Oracle 数据库实例的支持。	2012 年 8 月 16 日
新特征	更新了对 VPC 中 SQL Server Database Engine Tuning Advisor 和 Oracle 数据库实例的支持。	2012 年 7 月 18 日
新特征	更新对选项组和第一选项 Oracle Enterprise Manager Database Control 的支持。	2012 年 5 月 29 日
新特征	更新为在 Amazon Virtual Private Cloud 中支持只读副本。	2012 年 5 月 17 日
新特征	更新 Microsoft SQL Server 支持。	2012 年 5 月 8 日
新特征	更新了对强制失效转移、数据库实例的多可用区部署和 Oracle 数据库实例非默认字符集的支持。	2012 年 5 月 2 日
新特征	更新为支持 Amazon Virtual Private Cloud (VPC)。	2012 年 2 月 13 日
更新的内容	更新新预留实例类型。	2011 年 12 月 19 日
新特征	更新 Oracle 引擎支持。	2011 年 5 月 23 日
更新的内容	控制台更新。	2011 年 5 月 13 日
更新的内容	编辑缩短的备份和维护时段的内容。	2011 年 2 月 28 日
新特征	增加了对 MySQL 5.5 的支持。	2011 年 1 月 31 日
新特征	增加了对只读副本的支持。	2010 年 10 月 4 日
新特征	增加了对 Amazon Identity and Access Management（IAM）的支持。	2010 年 9 月 2 日
新特征	添加了数据库引擎版本管理。	2010 年 8 月 16 日
新特征	添加了预留数据库实例。	2010 年 8 月 16 日

更改	描述	更改日期
新特征	Amazon RDS 现已支持与数据库实例的 SSL 连接。	2010 年 6 月 28 日
新指南	这是 Amazon RDS 用户指南 的第一个版本。	2010 年 6 月 7 日

Amazon 术语表

有关最新的 Amazon 术语，请参阅 Amazon Web Services 词汇表 参考中的 Amazon 词汇表。