Relational Database Service

Best Practices

Issue 01

Date 2022-10-30

HUAWEI TECHNOLOGIES CO., LTD.

Huawei Technologies Co., Ltd.

Address:	Huawei Industrial Base Bantian, Longgang Shenzhen 518129 People's Republic of China
Website:	https://e.huawei.com

Security Declaration

Product Lifecycle

Huawei's regulations on product lifecycle are subject to the *Product End of Life Policy*. For details about this policy, visit the following web page:
https://support.huawei.com/ecolumnsweb/en/warranty-policy

Vulnerability

Huawei's regulations on product vulnerability management are subject to the *Vul. Response Process*. For details about this process, visit the following web page:
https://www.huawei.com/en/psirt/vul-response-process
For vulnerability information, enterprise customers can visit the following web page:
https://securitybulletin.huawei.com/enterprise/en/security-advisory

Preconfigured Digital Certificate

The digital certificates preconfigured on Huawei devices are subject to the *Rights and Responsibilities of Preconfigured Digital Certificates on Huawei Devices*. For details about this document, visit the following web page:
https://support.huawei.com/enterprise/en/bulletins-service/ENEWS2000015789

Huawei Enterprise End User License Agreement

This agreement is the end user license agreement between you (an individual, company, or any other entity) and Huawei for the use of the Huawei Software. Your use of the Huawei Software will be deemed as your acceptance of the terms mentioned in this agreement. For details about this agreement, visit the following web page:
https://e.huawei.com/en/about/eula

Lifecycle of Product Documentation

Huawei after-sales user documentation is subject to the *Product Documentation Lifecycle Policy*. For details about this policy, visit the following web page:
https://support.huawei.com/enterprise/en/bulletins-website/ENEWS2000017761

Contents

1 Overview

This document provides best practices for Huawei Cloud Relational Database Service (RDS) and guides you through using RDS to best suit your business needs.

DB Engine	Reference	Description
MySQL	**Migrating Data from Self-Managed MySQL Databases to RDS for MySQL**	Describes how to migrate data from self-managed MySQL databases to RDS for MySQL.
	Configuring Remote Single-Active DR for an RDS for MySQL Instance Using DRS	Describes how to use DRS to establish a remote single-active DR relationship for an RDS for MySQL instance.
	Migrating MySQL Databases from Other Clouds to RDS for MySQL	Describes how to migrate data from MySQL databases on other clouds to RDS for MySQL.
	Using RDS for MySQL to Set Up WordPress	Describes how to set up WordPress in a LAMP environment using Huawei Cloud Virtual Private Cloud (VPC), Elastic Cloud Server (ECS), and RDS for MySQL.
	Using RDS for MySQL to Set Up Discuz!	Describes how to set up Discuz! in a LAMP environment using Huawei Cloud Virtual Private Cloud (VPC), Elastic Cloud Server (ECS), and RDS for MySQL.
	Description of innodb_flush_log_at_trx_commit and sync_binlog	Describes the impact of the **innodb_flush_log_at_trx_commit** and **sync_binlog** parameters on performance and security.

DB Engine	Reference	Description
	Resolving Database Operation Failures Caused by Metadata Locks on RDS for MySQL	Describes how to use Data Admin Service (DAS) to solve the issue that database operations cannot be performed due to metadata locks.
	How Do I Use the utf8mb4 Character Set to Store Emojis in an RDS for MySQL DB Instance?	Describes how to store emojis in an RDS for MySQL DB instance.
	Performance Tuning	Describes how to troubleshoot high vCPU usage, high memory usage, insufficient storage, and slow queries for RDS for MySQL instances.
PostgreSQL	**Performing Basic Operations Using pgAdmin**	Describes how to use pgAdmin to connect to RDS for PostgreSQL and create databases and tables.
	Using PoWA	Describes how to use PoWA to monitor the performance of RDS for PostgreSQL.
	Viewing Slow Query Logs of RDS for PostgreSQL DB Instances	Describes how to query slow query logs of RDS for PostgreSQL.
SQL Server	**Restoring Data from Backup Files to RDS Microsoft SQL Server DB Instances**	Describes the version restrictions on RDS for SQL Server backup and restoration.
	Migrating Data from a Self-Managed SQL Server Database on an ECS to an RDS for SQL Server DB Instance	Describes how to migrate a self-managed SQL Server database on an ECS to an RDS for SQL Server DB instance.
	Modifying Parameters of RDS for SQL Server Instances	Describes how to modify parameter templates of RDS for SQL Server DB instances.
	Supporting DMVs	Describes how to dynamically manage views through DMV on RDS for SQL Server.

DB Engine	Reference	Description
	Using the Import and Export Function to Migrate Data from a Local Database to an RDS Microsoft SQL Server DB Instance	Describes how to migrate an on-premises SQL Server database to an RDS for SQL Server DB instance.
	Creating a Subaccount of rdsuser	Describes the permissions of the **rdsuser** account and how to create and manage IAM users under the **rdsuser** account.
	Creating tempdb Files	Describes how to create tempdb temporary data files on RDS for SQL Server.
	Microsoft SQL Server Publication and Subscription	Describes how RDS for SQL Server provides the subscription function.
	Installing a C# CLR Assembly in RDS for SQL Server	Describes how to add a c#CLR assembly on RDS for SQL Server.
	Creating a Linked Server for an RDS for SQL Server DB Instance	Describes how to create a linked server to access another RDS for SQL Server DB instance.
	Deploying SQL Server Reporting Services (SSRS) on RDS for SQL Server	Describes how to deploy SQL Server Reporting Services (SSRS) in RDS for SQL Server.
	Shrinking an RDS for SQL Server Database	Describes how to use a stored procedure to shrink the size of the data and log files in a specified RDS for SQL Server database.
	Using DAS to Create and Configure Agent Job and DBLink on the Master and Slave Databases for RDS for SQL Server Instances	Describes how to use DAS to create and configure agent jobs and DBLink on primary and standby RDS for SQL Server DB instances.
	Creating a Job for Scheduled Instance Maintenance	Describes how to create a scheduled SQL agent job to re-create indexes, update statistics, and shrink the database.

2 RDS for MySQL

2.1 Suggestions on Using RDS for MySQL

Database Naming

- The names of database objects like databases, tables, and columns should be in lowercase. Different words in the name are separated with underscores (_).
- Reserved words and keywords cannot be used to name database objects in RDS for MySQL.
  - Reserved words and keywords for MySQL 8.0: **https://dev.mysql.com/doc/refman/8.0/en/keywords.html**
  - Reserved words and keywords for MySQL 5.7: **https://dev.mysql.com/doc/refman/5.7/en/keywords.html**
  - Reserved words and keywords for MySQL 5.6: **https://dev.mysql.com/doc/refman/5.6/en/keywords.html**
- Each database object name must be explainable and contain a maximum of 32 characters.
- Each temporary table in databases is prefixed with **tmp** and suffixed with a date.
- Each backup table in databases is prefixed with **bak** and suffixed with a date.
- All columns storing the same data in different databases or tables must have the same name and be of the same type.

Database Design

- All tables use the InnoDB storage engine unless otherwise specified. InnoDB supports transactions and row locks. It delivers excellent performance, making it easy to recover data.
- Databases and tables all use the UTF8 character set to avoid characters getting garbled by character set conversion.
- All tables and fields require comments that can be added using the COMMENT clause to maintain the data dictionary from the beginning of the design.

- The length of a single row in the table cannot exceed 1024 bytes.
- To avoid cross-partition queries, RDS for MySQL partitioned tables are not recommended. Cross-partition queries will decrease the query efficiency. A partitioned table is logically a single table, but the data is actually stored in multiple different files.
- Do not create too many columns in one table. Store cold and warm data separately to reduce the width of a table. In doing so, more rows of data can be stored in each memory page, decreasing disk I/O and making more efficient use of the cache.
- Columns that are frequently used together should be in the same table to avoid JOIN operations.
- Do not create reserved fields in a table. Otherwise, modifying the column type will lock the table, which has a greater impact than adding a field.
- Do not store binary data such as images and files in databases.
- Full-text indexes are not recommended because there are many limitations on full-text indexes for MySQL Community Edition.

Field Design

- Ensure that each table contains no more than 50 fields.
- Select a small data type for each column as much as possible. Numeric data is preferred, followed by dates or binary data, and the least preferred is characters. The larger the column data type, the more the space required for creating indexes. As a result, there are fewer indexes on a page and more I/O operations required, so database performance deteriorates.
- If the integer type is used as the database field type, select the shortest column type. If the value is a non-negative number, it must be the unsigned type.
- Each field should have the NOT NULL attribute. The default value for the numeric type such as INT is recommended to be **0**, and that for the character type such as VARCHAR is recommended to be an empty string.
- Do not use the ENUM type. Instead, use the TINYINT type.

  Change ENUM values using ALTER. The ORDER BY operations on ENUM values are inefficient and require extra operations.

  If you have specified that ENUM values cannot be numeric, other data types (such as char) can be used.
- If the numeric data type is required, use DECIMAL instead of FLOAT or DOUBLE.

  FLOAT and DOUBLE data cannot be stored precisely, and value comparison results may be incorrect.
- When you want to record a date or specific time, use the DATETIME or TIMESTAMP type instead of the string type.
- Store IP addresses using the INT UNSIGNED type. You can convert IP addresses into numeric data using function inet_aton or inet_ntoa.
- The VARCHAR data should be as short as possible. Although the VARCHAR data varies in length dynamically on disks, it occupies the maximum length in memory.

- Use VARBINARY to store variable-length character strings that are case-sensitive. VARBINARY is case-sensitive by default and quick to process because no character sets are involved.

Index Design

- Create a primary key for each InnoDB table. Neither use a frequently-updated column as the primary key nor a multi-column primary key. Do not use the UUID, MD5, or character string column as the primary key. Use a column whose values can increment continuously as the primary key. So, the auto-increment ID column is recommended.
- Use no more than 5 indexes in a single table. Indexes speed up queries, but too many indexes may slow down writes. Inappropriate indexes sometimes reduce query efficiency.
- Do not create an independent index for each column in a table. A well-designed composite index is much more efficient than a separate index on each column.
- Create an index on the following columns:
  - Columns specified in the WHERE clause of SELECT, UPDATE, or DELETE statements
  - Columns specified in ORDER BY, GROUP BY, or DISTINCT
  - Columns associated for joining multiple tables.
- The index column order is as follows:
  - Put the column with the highest selectivity on the far left when creating a composite index. Selectivity = Different values in a column/Total rows in the column
  - Put the column with the smallest field length on the far left of the composite index. The smaller length a field has, the more data one page stores, and the better the I/O performance is.
  - Put the most frequently used column on the left of the composite index, so you can create fewer indexes.
- Avoid using redundant indexes, such as primary key (id), index (id), and unique index (id).
- Avoid using duplicate indexes, such as index(a,b,c), index(a,b), and index(a). Duplicate and redundant indexes may slow down queries because the RDS for MySQL query optimizer does not know which index it should use.
- When creating an index on the VARCHAR field, specify the index length based on selectivity. Do not index the entire field.

  If an index with the length of 20 bytes is the string type, its selectivity can reach 90% or above. In this case, use **count(distinct left(column name, index length))/count(*)** to check index selectivity.
- Use covering indexes for frequent queries.

  A covering index is a special type of index where all required fields for a query are included in the index. The index itself contains columns specified in WHERE and GROUP BY clauses, but also column combinations queried in SELECT, without having to execute additional queries.
- Constraints on foreign keys are as follows:

The character sets of the columns for which a foreign key relationship is established must be the same, or the character sets of the parent and child tables for which a foreign key relationship is established must be the same.

SQL Statement Development

- Use prepared statements to perform database operations in programs. Prepared statements can be executed multiple times in a program once they are written, more efficient than SQL statements.
- Avoid implicit conversions because they may cause index to become invalid.

  Do not perform function conversions or math calculations on columns in the WHERE clause. Otherwise, the index becomes invalid.
- Do not use double percent signs (%%) or place % before a query condition, or the index cannot be used.
- Do not use **select *** for queries because using **select ***:
  - Consumes more CPUs, IP addresses, and bandwidth.
  - Causes covering indexes to become unavailable.
  - Increases the impact of table structure changes on code.
- Do not use subqueries. Subqueries generate temporary tables that do not have any indexes. If there is a lot of data, the query efficiency is severely affected. Convert subqueries into associated queries.
- Minimize the use of JOIN operations for more than 5 tables. Use the same data type for the fields that require JOIN operations.

  Each JOIN operation on a table occupies extra memory (controlled by **join_buffer_size**) and requires temporary table operations, affecting query efficiency. Do not use NATURAL JOIN.
- Reduce interactions with the same database as much as possible. The database is more suitable for processing batch operations.
- Replace OR operations with IN operations. IN operations can effectively use indexes. The number of IN values cannot exceed 500.
- Do not perform reverse queries, for example, NOT IN and NOT LIKE.
- Do not use ORDER BY RAND() for random sorting.

  This operation loads all data that meets the conditions from the table to the memory for sorting, consuming more CPUs, I/O, and memory resources.

  Obtain a random value from the program and retrieve data from the involved database based on the value.
- If deduplication is not required, use UNION ALL instead of UNION.

  UNION ALL does not sort out result sets.
- Combine multiple operations and perform them in batches. The database is good for batch processing.

  This reduces interactions with the same database.
- If there are more than 1 million rows of write operations, perform them in multiple batches.

  A large number of batch writes may result in excessive primary/standby latency.
- If ORDER BY is used, use the order of indexes.

  - The last field of ORDER BY is a part of a composite index and is placed at the end of the composite index order.
  - Avoid file_sort to speed up queries.

  Correct example: in **where a=? and b=? order by c;**, index: **a_b_c**

  Wrong example: If an index supports range search, the index order cannot be used. For example, **WHERE a>10 ORDER BY b;**, index: **a_b** (sorting is not allowed)

- Use ANSI-standard SQL statements instead of MySQL extended SQL statements for DML operations. Common MySQL extended SQL statements include:
  - REPLACE INTO
  - INSERT ... ON DUPLICATE KEY UPDATE
- Stored procedures are not recommended because they are difficult to debug, extend, and transplant.
- To avoid logical dependency on the database, do not use triggers, event schedulers, or views for service logic.
- Large transactions are not recommended. If possible, a transaction should contain no more than five SQL statements because large transactions have problems such as long data lock time, too many caches, and connection consumption.
- TRUNCATE TABLE is faster than DELETE and uses fewer system and log resources. If the table to be deleted does not have a trigger and the entire table needs to be deleted, TRUNCATE TABLE is recommended.
- Do not run the **flush logs** command frequently to prevent automatic binlog deletion failures.

2.2 Migrating Data from Self-Managed MySQL Databases to RDS for MySQL

2.2.1 Overview

Scenarios

This chapter includes the following content:

- How to migrate data from self-managed MySQL databases to RDS for MySQL instances

RDS for MySQL Advantages

- **More Services at Lower Costs**

  You pay for only RDS instances. There is no hardware or management investment needed.
- **Ultimate User Experience**
  - Fully compatible with MySQL

  - Excellent performance for high concurrency
  - Support for a great number of connections and quicker response
- **High Security**
  - End-to-end database security, including network isolation, access control, transmission encryption, storage encryption, and anti-DDoS
  - Highest-level certification by the NIST-CSF, with 108 key security capabilities
- **High Reliability**

  Multiple deployment and DR solutions, including data backup, data restoration, dual-host hot standby, remote DR, and intra-city DR

Service List

- Virtual Private Cloud (VPC)
- Elastic Cloud Server (ECS)
- RDS
- Data Replication Service (DRS)

Notes on Usage

- The resource planning in this best practice is for demonstration only. Adjust it as needed.
- All settings in this best practice are for reference only. For more information about MySQL migration, see **From MySQL to MySQL**.

Prerequisites

- You have registered with Huawei Cloud.
- Your account balance is greater than or equal to $0 USD.

2.2.2 Resource Planning

Table 2-1 Resource planning description

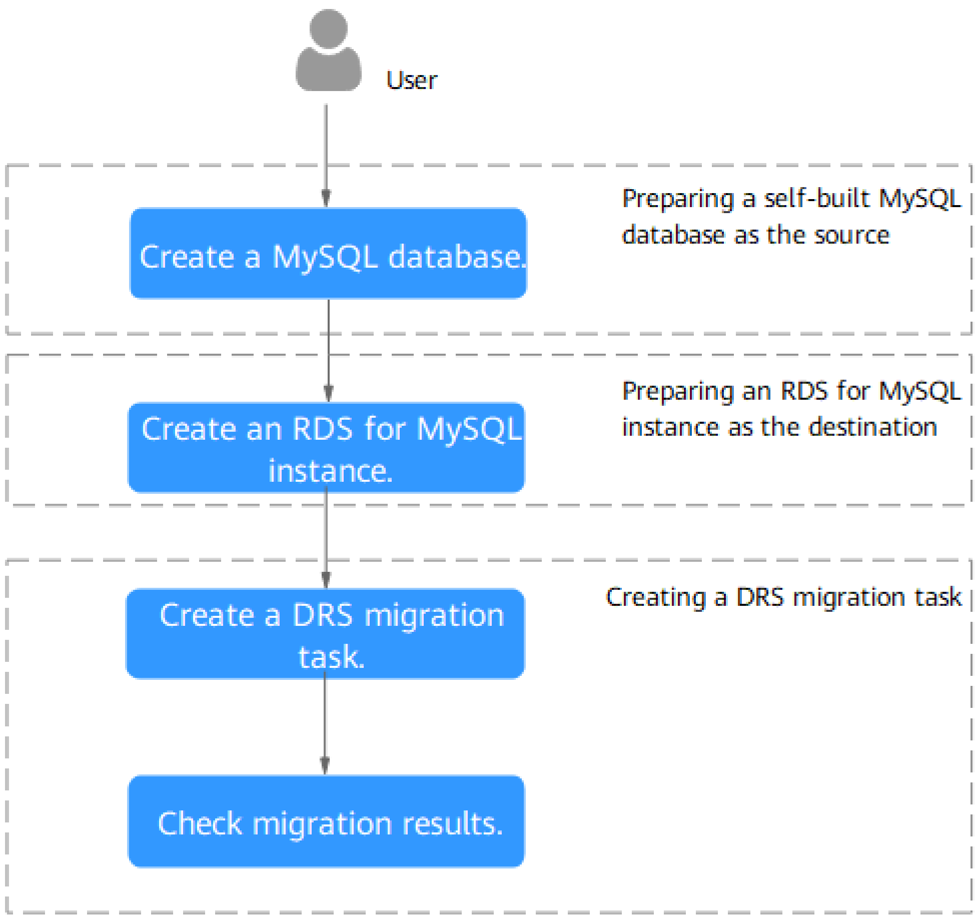
Category	Subcategory	Planned Value	Remarks
RDS	RDS instance name	rds-mysql	Customize a name for easy identification.
	DB engine version	MySQL 5.7	-
	Instance type	Single	In this practice, select a single instance. To improve service reliability, selecting a primary/standby instance is recommended.

Category	Subcategory	Planned Value	Remarks
	Storage type	Cloud SSD	-
	AZ	AZ3	In this practice, select a single instance. To improve service reliability, create a primary/standby instance and then deploy them in two different AZs.
	Specifications	General-purpose 4 vCPUs | 8 GB	-
DRS migration task	Task name	DRS-mysql	Custom
	Source DB engine	MySQL	In this practice, the source is a MySQL database built on an ECS.
	Destination DB engine	MySQL	In this practice, the destination is an RDS for MySQL instance.
	Network type	VPC	In this practice, select the VPC network.

2.2.3 Operation Process

The following figure shows the process of creating a MySQL database on an ECS, buying an RDS for MySQL instance, and migrating data from the MySQL database to the RDS instance.

Figure 2-1 Flowchart


User
Create a MySQL database.
Preparing a self-built MySQL database as the source
Create an RDS for MySQL instance.
Preparing an RDS for MySQL instance as the destination
Create a DRS migration task.
Creating a DRS migration task
Check migration results.

2.2.4 Cloud Migration

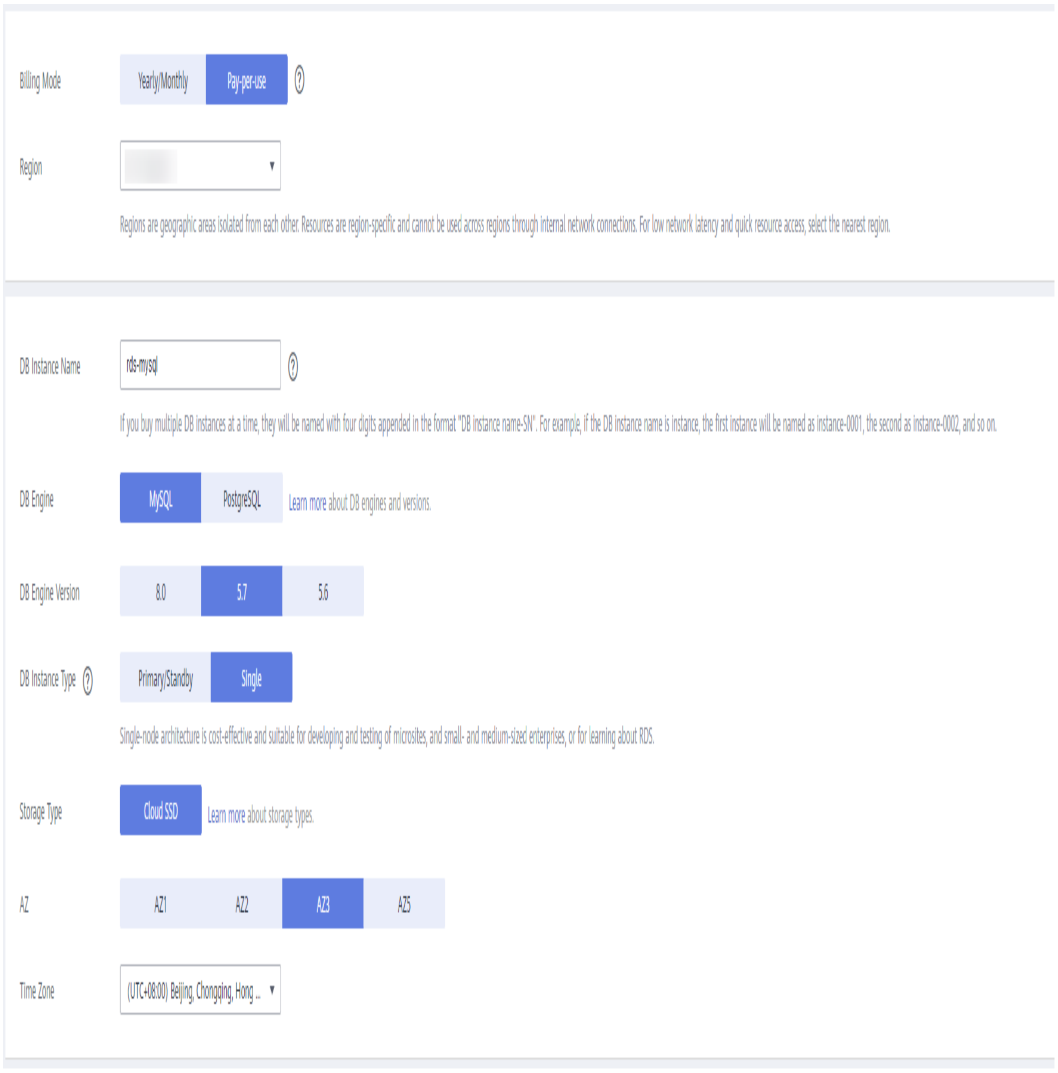
This topic describes how to migrate data from a self-managed MySQL database to an RDS instance. The involved tasks include buying an RDS instance and creating a DRS migration task.

2.2.4.1 Creating an RDS Instance

Create an RDS instance that is in the same VPC and security group as the self-managed MySQL database.

Step 1 Go to the **Buy DB Instance** page.

Step 2 Configure the instance name and basic information. Select **CN-Hong Kong** for **Region**.



Billing Mode: Yearly/Monthly [✓ Pay-per-use]

Region:

Regions are geographic areas isolated from each other. Resources are region-specific and cannot be used across regions through internal network connections. For low network latency and quick resource access, select the nearest region.

DB Instance Name: rds-mysql

If you buy multiple DB instances at a time, they will be named with four digits appended in the format "DB instance name-SN". For example, if the DB instance name is instance, the first instance will be named as instance-0001, the second as instance-0002, and so on.

DB Engine: [✓ MySQL] PostgreSQL Learn more about DB engines and versions.

DB Engine Version: 8.0 [✓ 5.7] 5.6

DB Instance Type: Primary/Standby [✓ Single]

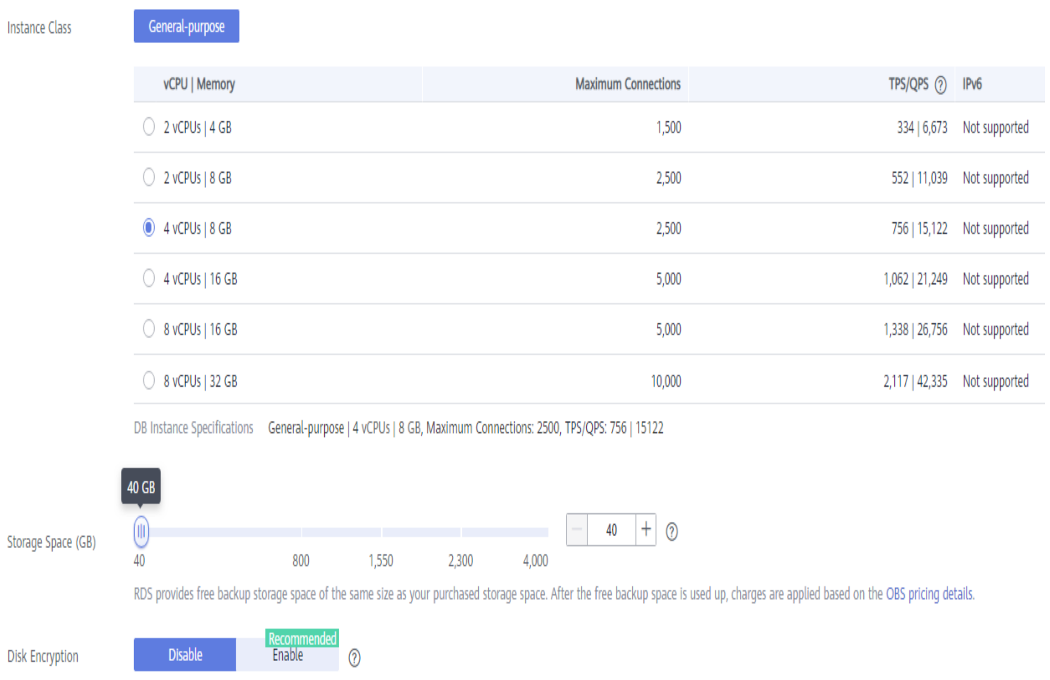
Single-node architecture is cost-effective and suitable for developing and testing of microsites, and small- and medium-sized enterprises, or for learning about RDS.

Storage Type: [✓ Cloud SSD] Learn more about storage types.

AZ: AZ1 AZ2 [✓ AZ3] AZ5

Time Zone: (UTC+08:00) Beijing, Chongqing, Hong ...

Step 3 Select an instance class.



Instance Class: General-purpose

vCPU | Memory	Maximum Connections	TPS/QPS	IPv6
2 vCPUs | 4 GB	1,500	334 | 6,673	Not supported
2 vCPUs | 8 GB	2,500	552 | 11,039	Not supported
[✓ 4 vCPUs | 8 GB]	2,500	756 | 15,122	Not supported
4 vCPUs | 16 GB	5,000	1,062 | 21,249	Not supported
8 vCPUs | 16 GB	5,000	1,338 | 26,756	Not supported
8 vCPUs | 32 GB	10,000	2,117 | 42,335	Not supported

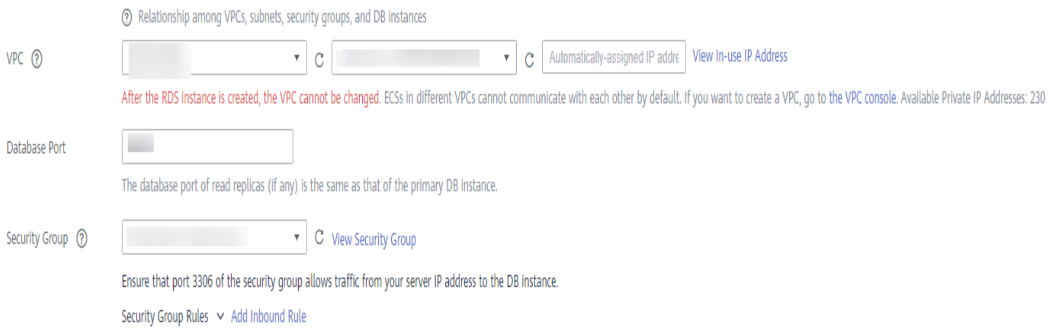
DB Instance Specifications: General-purpose | 4 vCPUs | 8 GB, Maximum Connections: 2500, TPS/QPS: 756 | 15122

Storage Space (GB): 40 GB

RDS provides free backup storage space of the same size as your purchased storage space. After the free backup space is used up, charges are applied based on the OBS pricing details.

Disk Encryption: [✓ Disable] Enable

Step 4 Select a VPC and security group for the instance and configure the database port.



Relationship among VPCs, subnets, security groups, and DB instances

VPC: Automatically-assigned IP addre View In-use IP Address

After the RDS instance is created, the VPC cannot be changed. ECSs in different VPCs cannot communicate with each other by default. If you want to create a VPC, go to the VPC console. Available Private IP Addresses: 230

Database Port

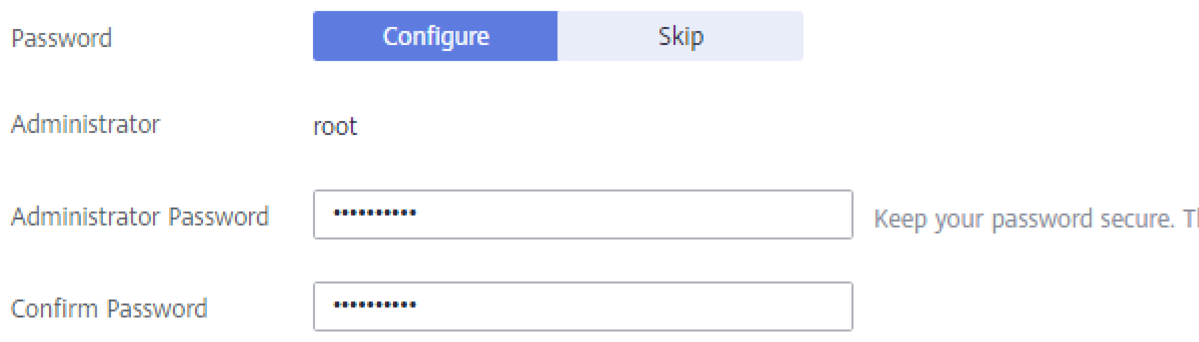
The database port of read replicas (if any) is the same as that of the primary DB instance.

Security Group: View Security Group

Ensure that port 3306 of the security group allows traffic from your server IP address to the DB instance.

Security Group Rules Add Inbound Rule

Step 5 Configure the instance password.



Password: [✓ Configure] Skip

Administrator: root

Administrator Password: •••••••••• Keep your password secure. T

Confirm Password: ••••••••••

Step 6 Click **Next**.

Step 7 Confirm the settings.

- To modify your settings, click **Previous**.

- If you do not need to modify your settings, click **Submit**.

Step 8 Return to the instance list.

If the instance status becomes available, the instance has been created.

----End

2.2.4.2 Creating a Migration Task

This topic describes how to create a DRS migration task to migrate the **loadtest** database from the self-managed MySQL server to an RDS for MySQL instance.

Pre-migration Check

Before creating a migration task, check migration conditions to ensure smooth migration.

This example describes how to migrate data from a self-managed MySQL database to an RDS for MySQL instance. For more information, see **From MySQL to MySQL**.

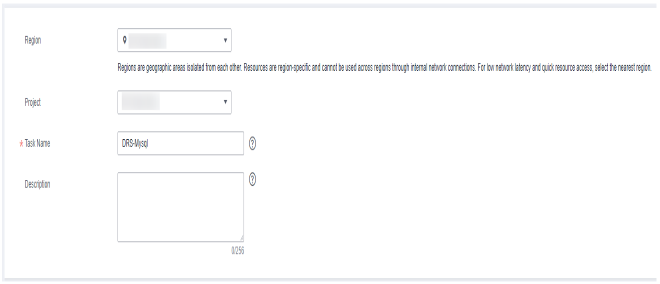
Procedure

Migrate the **loadtest** database from a self-managed MySQL server to an RDS for MySQL instance.

Step 1 Go to the **Create Migration Task** page.

Step 2 Configure parameters as needed.

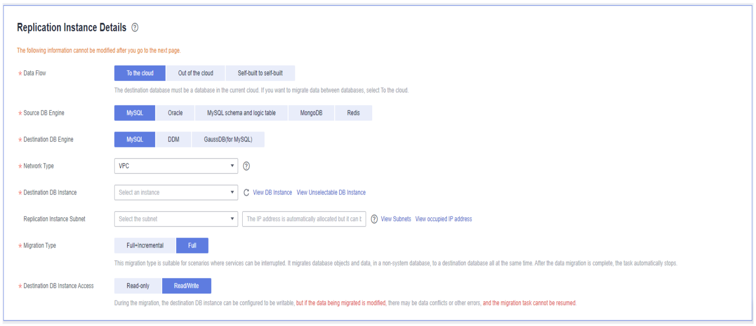
1. Specify a migration task name. Select **CN-Hong Kong** for **Region**.



Region	
	Regions are geographic areas isolated from each other. Resources are region-specific and cannot be used across regions through internal network connections. For low network latency and quick resource access, select the nearest region.
Project	
* Task Name	DRS-Mysql
Description	0/256

2. Configure replication instance information.

   Select the instance created in **Creating an RDS Instance** as the destination instance.



Replication Instance Details

The following information cannot be modified after you go to the next page.

* Data Flow	[✓ To the cloud] Out of the cloud Self-built to self-built
	The destination database must be a database in the current cloud. If you want to migrate data between databases, select To the cloud.
* Source DB Engine	[✓ MySQL] Oracle MySQL schema and logic table MongoDB Redis
* Destination DB Engine	[✓ MySQL] DDM GaussDB(for MySQL)
* Network Type	VPC
* Destination DB Instance	Select an instance — View DB Instance View Unselectable DB Instance
Replication Instance Subnet	Select the subnet — The IP address is automatically allocated but it can t — View Subnets View occupied IP address
* Migration Type	Full+Incremental [✓ Full]
	This migration type is suitable for scenarios where services can be interrupted. It migrates database objects and data, in a non-system database, to a destination database all at the same time. After the data migration is complete, the task automatically stops.
* Destination DB Instance Access	Read-only [✓ Read/Write]
	During the migration, the destination DB instance can be configured to be writable, but if the data being migrated is modified, there may be data conflicts or other errors, and the migration task cannot be resumed.

3. Select **default** for **Enterprise Project**.

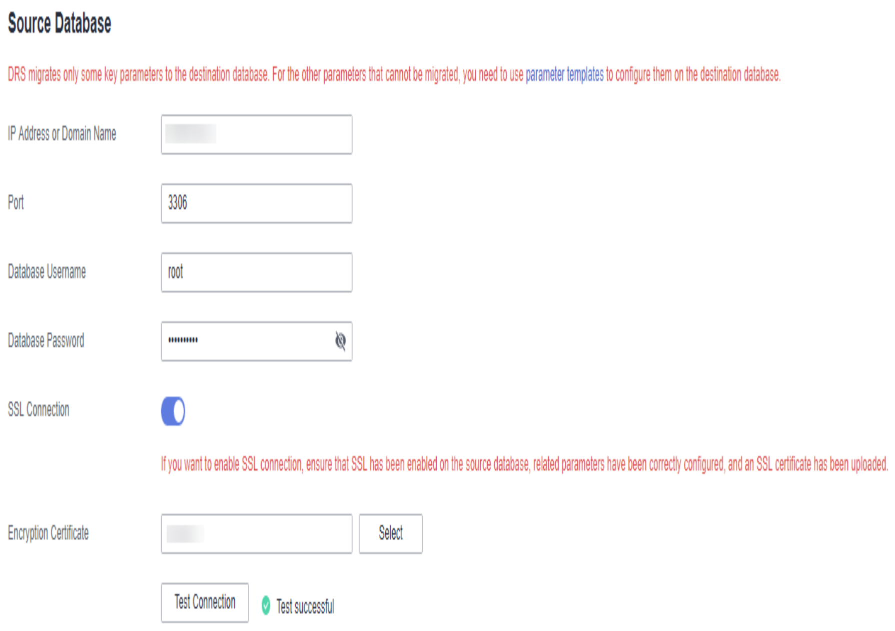
Step 3 Click **Create Now**.

It takes about 5 to 10 minutes to create a replication task.

Step 4 Configure task information.

1. Configure source database information.
2. Click **Test Connection**.

   If a successful test message is returned, login to the source is successful.



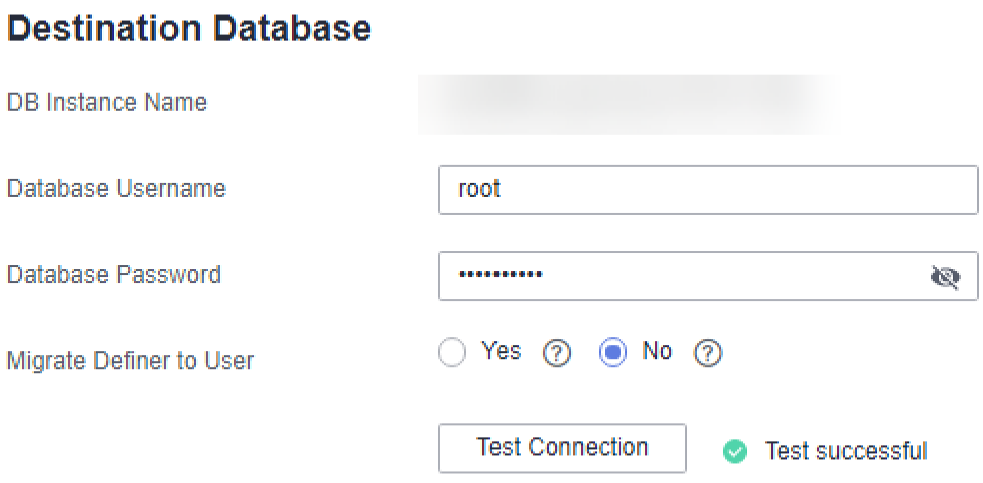
Source Database

DRS migrates only some key parameters to the destination database. For the other parameters that cannot be migrated, you need to use parameter templates to configure them on the destination database.

IP Address or Domain Name	
Port	3306
Database Username	root
Database Password	••••••••••
SSL Connection	
	If you want to enable SSL connection, ensure that SSL has been enabled on the source database, related parameters have been correctly configured, and an SSL certificate has been uploaded.
Encryption Certificate	

3. Specify a username and password for the destination database.
4. Click **Test Connection**.

   If a successful test message is returned, login to the destination is successful.



Destination Database

DB Instance Name	
Database Username	root
Database Password	•••••••••••
Migrate Definer to User	Yes [✓ No]

Step 5 Click **Next**.

Step 6 Confirm the migration user and migration objects.

Select **All** for **Migration Object**.

Step 7 Click **Next**.

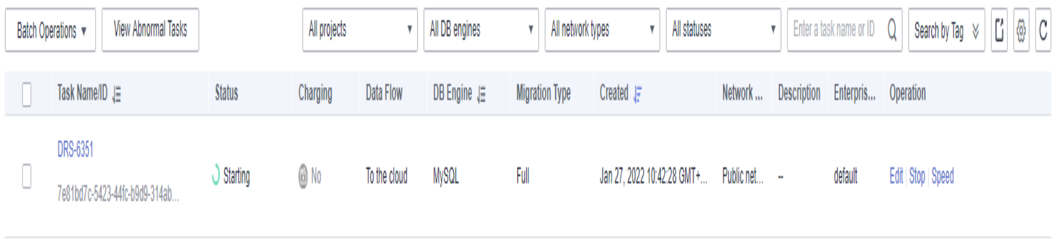
Step 8 View pre-check results.

Step 9 If the results of all check items are **Passed**, click **Next**.

Step 10 Click **Submit**.

Return to the **Online Migration Management** page and check the migration task status.

It takes several minutes to complete.



Task Name/ID	Status	Charging	Data Flow	DB Engine	Migration Type	Created	Network ...	Description	Enterpris...	Operation
DRS-6351 7e81bd7c-5423-44fc-b9d9-314ab...	Starting	No	To the cloud	MySQL	Full	Jan 27, 2022 10:42:28 GMT+...	Public net...	--	default	Edit | Stop | Speed

If the status changes to **Completed**, the migration task is complete.

----End

2.2.4.3 Confirming Migration Results

You can check migration results with either of the following methods:

Automatic: **Viewing Migration Results on the DRS Console**. DRS automatically compares migration objects, users, and data of source and destination databases and provides migration results.

Manual: **Viewing Migration Results on the RDS Console**. You can log in to the destination instance to check whether the databases, tables, and data are migrated.

Viewing Migration Results on the DRS Console

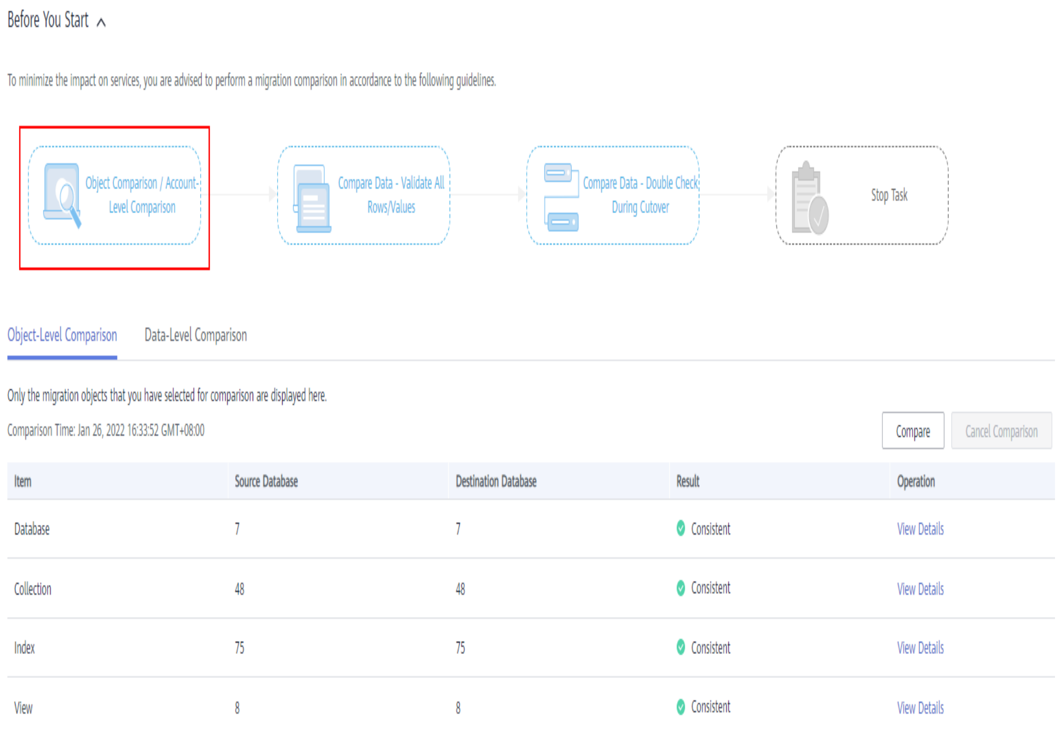
Step 1 Log in to the **management console**.

Step 2 Click in the upper left corner and select **CN-Hong Kong**.

Step 3 Click the service list icon on the left and choose **Databases** > **Data Replication Service**.

Step 4 Locate the required DRS instance and click its name.

Step 5 Click **Migration Comparison**.


Before You Start
Object Comparison / Account-Level Comparison
Compare Data - Validate All Rows/Values
Compare Data - Double Check During Cutover
Stop Task
Object-Level Comparison
Data-Level Comparison
Compare
Cancel Comparison

To minimize the impact on services, you are advised to perform a migration comparison in accordance to the following guidelines.

Only the migration objects that you have selected for comparison are displayed here.

Comparison Time: Jan 26, 2022 16:33:52 GMT+08:00

Item	Source Database	Destination Database	Result	Operation
Database	7	7	✓ Consistent	View Details
Collection	48	48	✓ Consistent	View Details
Index	75	75	✓ Consistent	View Details
View	8	8	✓ Consistent	View Details

Step 6 Select **Compare Data - Validate All Rows/Values** and **Compare Data -Double Check During Cutover** to check whether the objects of the source database have been migrated to the destination database.

If any check fails, rectify the fault by referring to **Solutions to Failed Check Items**.

----End

Viewing Migration Results on the RDS Console

Step 1 Log in to the **management console**.

Step 2 Click in the upper left corner and select **CN-Hong Kong**.

Step 3 Click the service list icon on the left and choose **Databases** > **Relational Database Service**.

Step 4 Locate the required RDS instance and click **Log In** in the **Operation** column.

Step 5 In the displayed dialog box, enter the password and click **Test Connection**.

Step 6 After the connection test is successful, click **Log In**.

Step 7 Check and confirm the destination database name and table name. Check whether the data migration is complete.

----End

2.3 Configuring Remote Single-Active DR for an RDS for MySQL Instance Using DRS

2.3.1 Overview

Scenarios

This best practice involves two tasks:

- Create an RDS for MySQL instance.
- Use DRS to establish a remote single-active DR relationship for the RDS for MySQL instance.

Prerequisites

- You have registered with Huawei Cloud.
- Your account balance is at least $0 USD.

How Cross-Region DR Works

RDS for MySQL instances are deployed in the production and DR data centers. DRS replicates data from the production center to the DR center, keeping data synchronous between your primary instance and the DR instance.

Service List

- Virtual Private Cloud (VPC)
- Elastic IP (EIP)
- Relational Database Service (RDS)
- Data Replication Service (DRS)

Notes on Usage

- The resource planning in this best practice is for demonstration only. Adjust it as needed.
- All settings in this best practice are for reference only. For more information about RDS for MySQL instance DR, see **From MySQL to MySQL (Single-Active DR)**.

2.3.2 Resource Planning

Table 2-2 Resource planning

Category	Subcategory	Planned Value	Description
VPC in the production center	VPC name	vpc-01	Specify a name that is easy to identify.
	Region	CN-Hong Kong	To achieve lower network latency, select the region nearest to you.
	AZ	AZ2	-

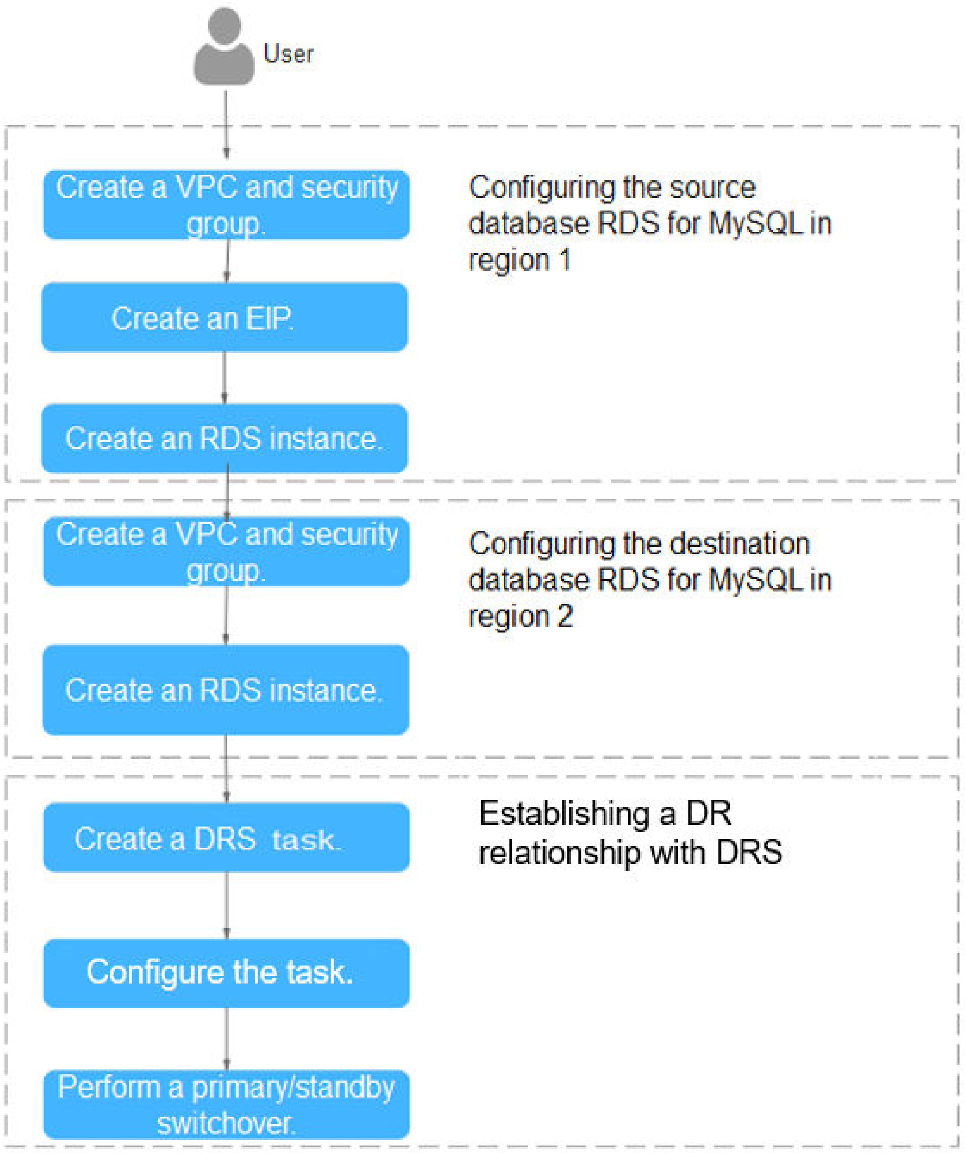
Category	Subcategory	Planned Value	Description
	Subnet	192.168.0.0/24	Select a subnet with sufficient network resources.
	Subnet name	subnet-3c29	Specify a name that is easy to identify.
VPC in the DR center	VPC name	vpc-DR	Specify a name that is easy to identify.
	Region	AP-Singapore	To achieve lower network latency, select the region nearest to you.
	AZ	AZ1	-
	Subnet	192.168.0.0/24	Select a subnet with sufficient network resources.
	Subnet name	subnet-ac27	Specify a name that is easy to identify.
RDS for MySQL instance in the production center	Instance name	rds-database-01	Specify a name that is easy to identify.
	Region	CN-Hong Kong	To achieve lower network latency, select the region nearest to you.
	DB engine version	MySQL 8.0	-
	Instance type	Single	A single instance is used in this example. To improve service reliability, select a primary/standby instance.
	Storage type	Ultra-high I/O	-
	AZ	AZ2	AZ2 is selected in this example. To improve service reliability, select the primary/standby instance type and deploy the primary and standby instances in different AZs.
	Instance specifications	General-enhanced 2 vCPUs | 4 GB	-

Category	Subcategory	Planned Value	Description
RDS for MySQL instance in the DR center	Instance name	rds-DR	Specify a name that is easy to identify.
	Region	AP-Singapore	To achieve lower network latency, select the region nearest to you.
	DB engine version	MySQL 8.0	-
	Instance type	Single	A single instance is used in this example. To improve service reliability, select a primary/standby instance.
	Storage type	Cloud SSD	-
	AZ	AZ1	AZ1 is selected in this example. To improve service reliability, select the primary/standby instance type and deploy the primary and standby instances in different AZs.
	Instance specifications	General-purpose 2 vCPUs | 8 GB	-
DRS DR task	DR task name	DRS-DR-Task	Specify a name that is easy to identify.
	Source DB engine	MySQL	In this example, the primary instance created in CN-Hong Kong is used as the source database.
	Destination DB engine	MySQL	In this example, the DR instance created in AP-Singapore is used as the destination database.
	Network type	Public network	Public network is used in this example.

2.3.3 Operation Process

You can create a single RDS instance and a DR instance and migrate data from the single instance to the DR instance.

Figure 2-2 Flowchart


User
Create a VPC and security group.
Create an EIP.
Create an RDS instance.
Configuring the source database RDS for MySQL in region 1
Create a VPC and security group.
Create an RDS instance.
Configuring the destination database RDS for MySQL in region 2
Create a DRS task.
Configure the task.
Perform a primary/standby switchover.
Establishing a DR relationship with DRS

2.3.4 Configuring an RDS for MySQL Instance in the Production Center

2.3.4.1 Creating a VPC and Security Group

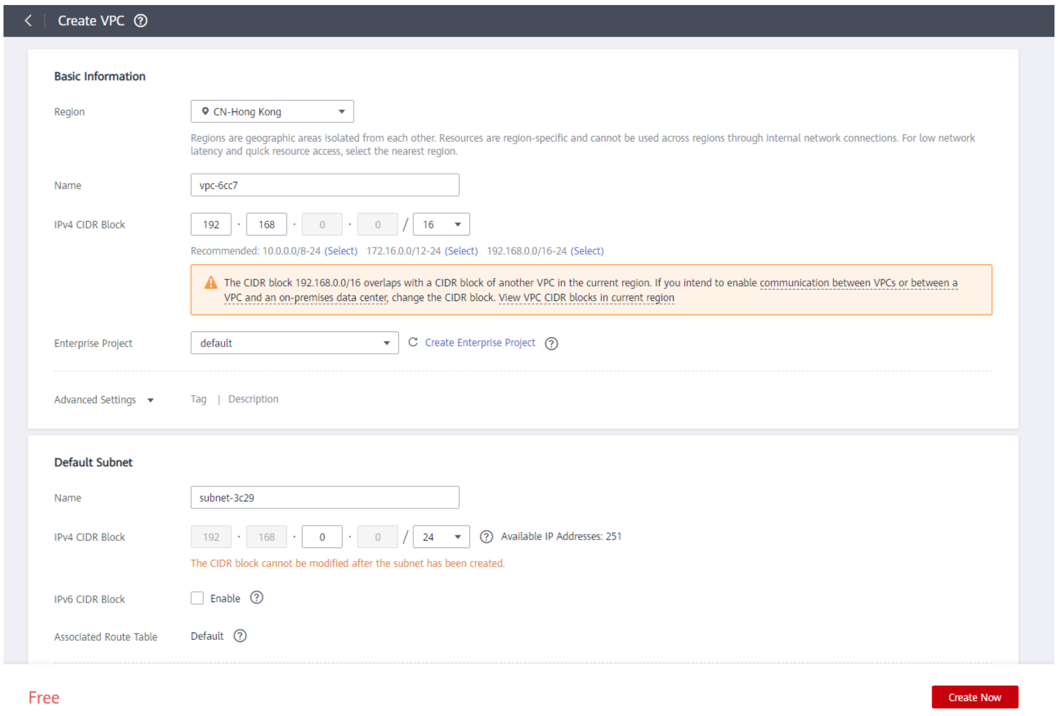
Create a VPC and security group for a DB instance in the production center.

Creating a VPC

Step 1 Go to the **Create VPC** page.

Step 2 Configure the basic information, subnet, and IP address. Select **CN-Hong Kong** for **Region**.

Figure 2-3 Creating a VPC



Basic Information

Region: CN-Hong Kong

Regions are geographic areas isolated from each other. Resources are region-specific and cannot be used across regions through internal network connections. For low network latency and quick resource access, select the nearest region.

Name: vpc-6cc7

IPv4 CIDR Block: 192 . 168 . 0 . 0 / 16

Recommended: 10.0.0.0/8-24 (Select) 172.16.0.0/12-24 (Select) 192.168.0.0/16-24 (Select)

The CIDR block 192.168.0.0/16 overlaps with a CIDR block of another VPC in the current region. If you intend to enable communication between VPCs or between a VPC and an on-premises data center, change the CIDR block. View VPC CIDR blocks in current region

Enterprise Project: default

Default Subnet

Name: subnet-3c29

IPv4 CIDR Block: 192 . 168 . 0 . 0 / 24 Available IP Addresses: 251

The CIDR block cannot be modified after the subnet has been created.

IPv6 CIDR Block: Enable

Associated Route Table: Default

Step 3 Click **Create Now**.

----End

Creating a Security Group

Step 1 Log in to the **management console**.

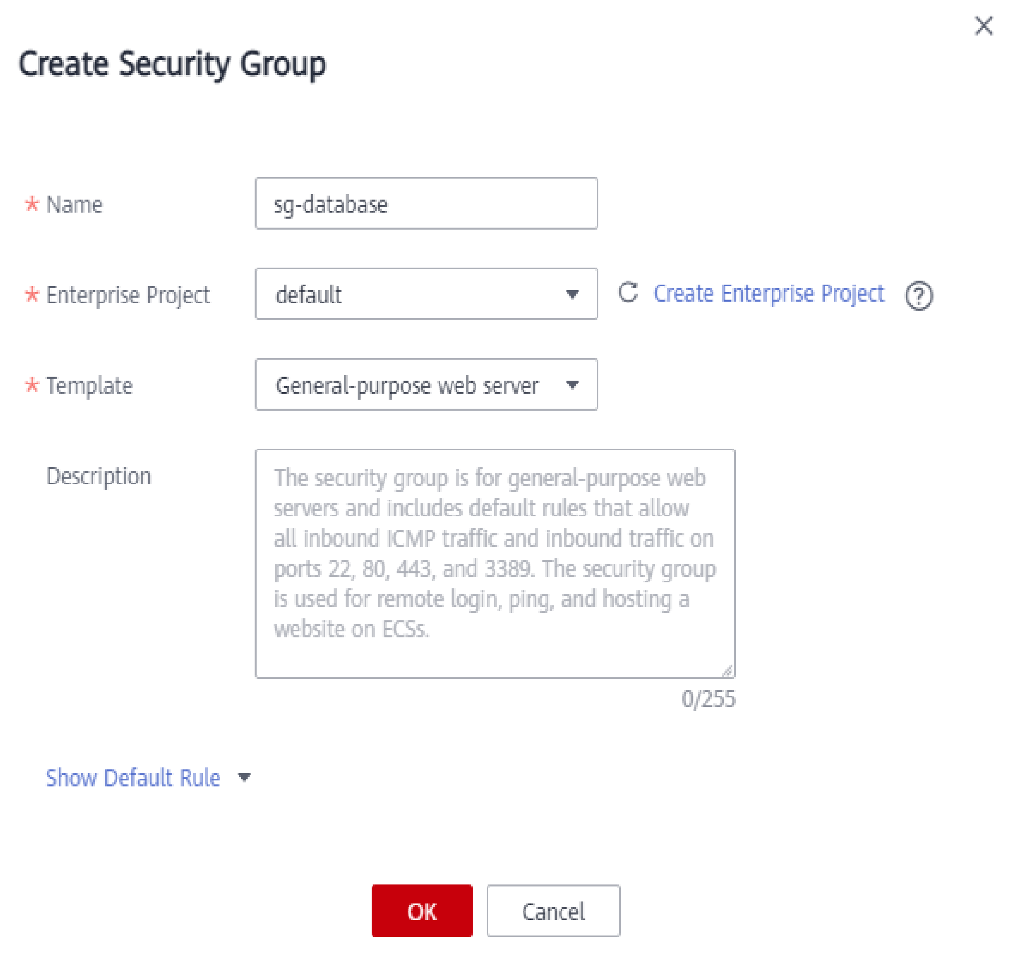
Step 2 Click in the upper left corner of the management console and select **CN-Hong Kong**.

Step 3 Under the service list, choose **Networking** > **Virtual Private Cloud**.

Step 4 In the navigation pane on the left, choose **Access Control** > **Security Groups**.

Step 5 Click **Create Security Group**.

Figure 2-4 Creating a security group


Create Security Group

* Name	sg-database
* Enterprise Project	default
* Template	General-purpose web server
Description	The security group is for general-purpose web servers and includes default rules that allow all inbound ICMP traffic and inbound traffic on ports 22, 80, 443, and 3389. The security group is used for remote login, ping, and hosting a website on ECSs.

0/255

Step 6 Click **OK**.

----End

2.3.4.2 Creating an EIP

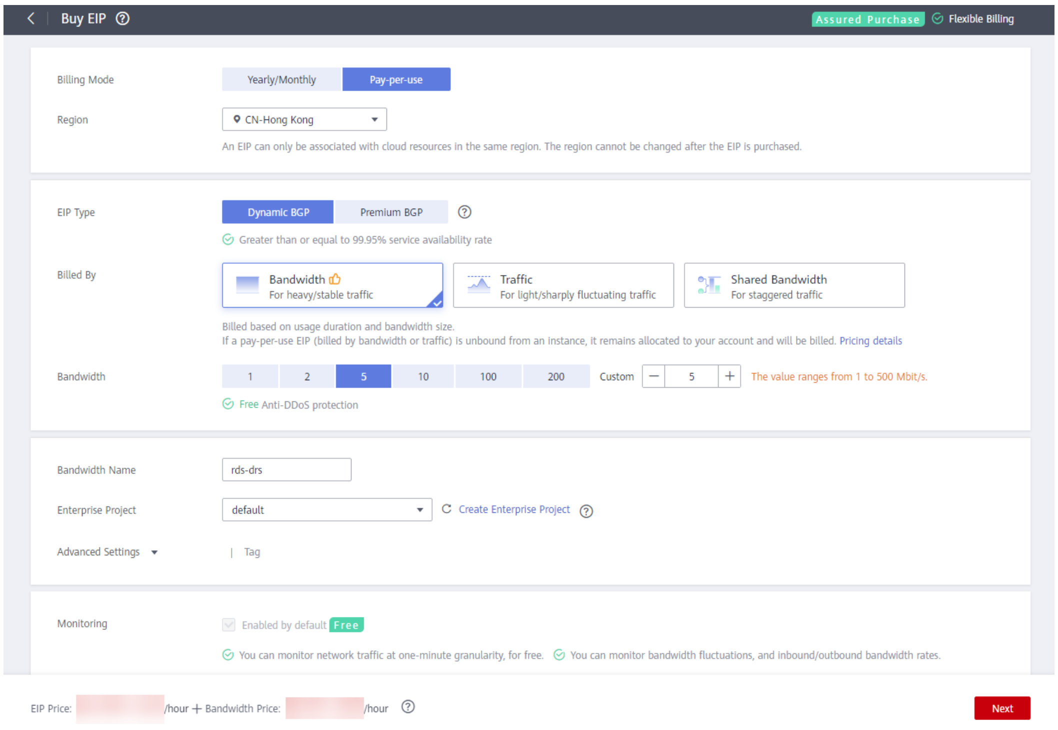
Create an EIP for your source DB instance. Using the EIP, external systems can access your application and DRS can connect to the source DB instance.

Procedure

Step 1 Go to the **Buy EIP** page.

Step 2 Configure required parameters. Select **CN-Hong Kong** for **Region**.

Figure 2-5 Buying an EIP



Buy EIP
Assured Purchase | Flexible Billing

Billing Mode: Yearly/Monthly | Pay-per-use
Region: CN-Hong Kong
An EIP can only be associated with cloud resources in the same region. The region cannot be changed after the EIP is purchased.

EIP Type: Dynamic BGP | Premium BGP
Greater than or equal to 99.95% service availability rate
Billed By: Bandwidth (For heavy/stable traffic) | Traffic (For light/sharply fluctuating traffic) | Shared Bandwidth (For staggered traffic)
Billed based on usage duration and bandwidth size.
If a pay-per-use EIP (billed by bandwidth or traffic) is unbound from an instance, it remains allocated to your account and will be billed. Pricing details
Bandwidth: 1 | 2 | 5 | 10 | 100 | 200 | Custom 5
The value ranges from 1 to 500 Mbit/s.
Free Anti-DDoS protection

Bandwidth Name: rds-drs
Enterprise Project: default
Create Enterprise Project
Advanced Settings | Tag

Monitoring: Enabled by default Free
You can monitor network traffic at one-minute granularity, for free. You can monitor bandwidth fluctuations, and inbound/outbound bandwidth rates.

EIP Price: [redacted] /hour + Bandwidth Price: [redacted] /hour
Next

Step 3 Click **Next**.

Step 4 Confirm the information and click **Submit**.

----End

2.3.4.3 Creating an RDS for MySQL Instance

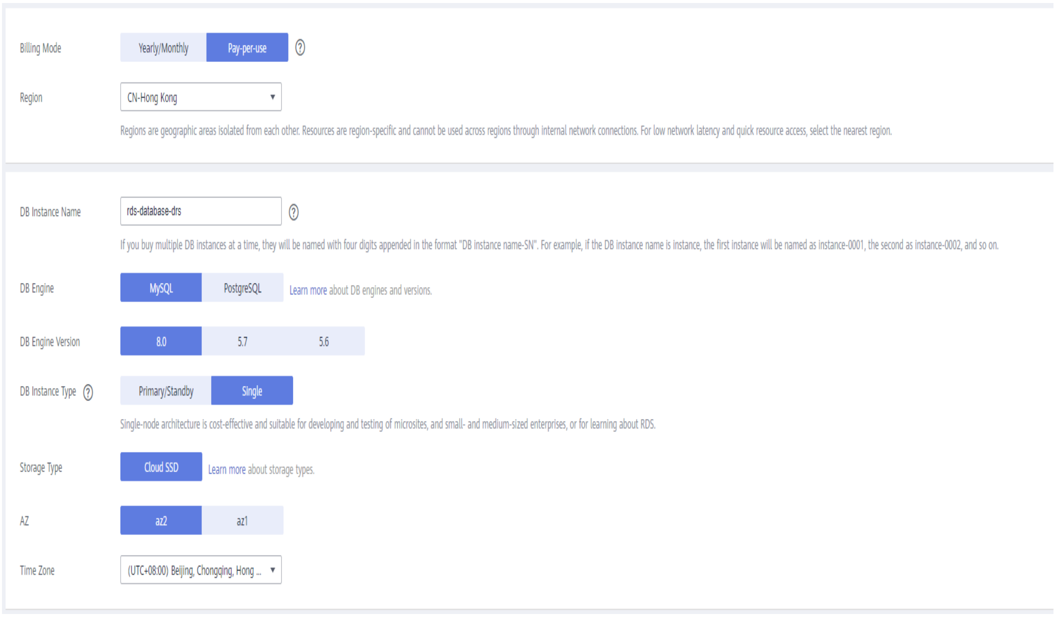
Create an RDS for MySQL instance (source database), and select the VPC and EIP you configured for the instance.

Procedure

Step 1 Go to the **Buy DB Instance** page.

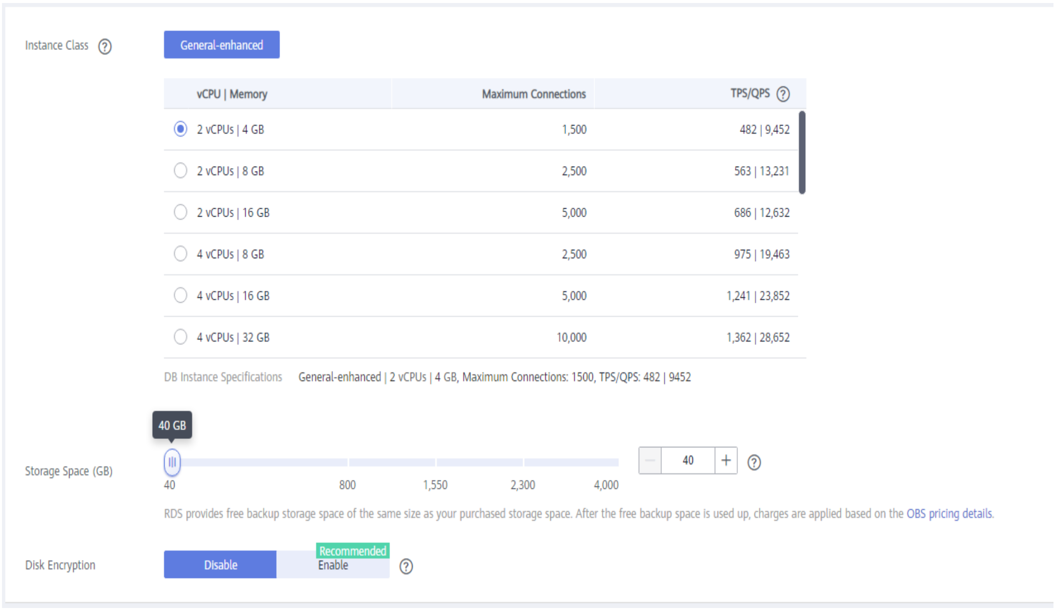
Step 2 Select **CN-Hong Kong** for **Region**. Configure instance information and click **Next**.

Figure 2-6 Selecting a DB engine



Billing Mode	Yearly/Monthly [✓ Pay-per-use]
Region	CN-Hong Kong
	Regions are geographic areas isolated from each other. Resources are region-specific and cannot be used across regions through internal network connections. For low network latency and quick resource access, select the nearest region.
DB Instance Name	rds-database-drs
	If you buy multiple DB instances at a time, they will be named with four digits appended in the format "DB instance name-SN". For example, if the DB instance name is instance, the first instance will be named as instance-0001, the second as instance-0002, and so on.
DB Engine	[✓ MySQL] PostgreSQL Learn more about DB engines and versions.
DB Engine Version	[✓ 8.0] 5.7 5.6
DB Instance Type	Primary/Standby [✓ Single]
	Single-node architecture is cost-effective and suitable for developing and testing of microsites, and small- and medium-sized enterprises, or for learning about RDS.
Storage Type	[✓ Cloud SSD] Learn more about storage types.
AZ	[✓ az2] az1
Time Zone	(UTC+08:00) Beijing, Chongqing, Hong ...

Figure 2-7 Selecting specifications



Instance Class: [✓ General-enhanced]

	vCPU | Memory	Maximum Connections	TPS/QPS
●	2 vCPUs | 4 GB	1,500	482 | 9,452
○	2 vCPUs | 8 GB	2,500	563 | 13,231
○	2 vCPUs | 16 GB	5,000	686 | 12,632
○	4 vCPUs | 8 GB	2,500	975 | 19,463
○	4 vCPUs | 16 GB	5,000	1,241 | 23,852
○	4 vCPUs | 32 GB	10,000	1,362 | 28,652

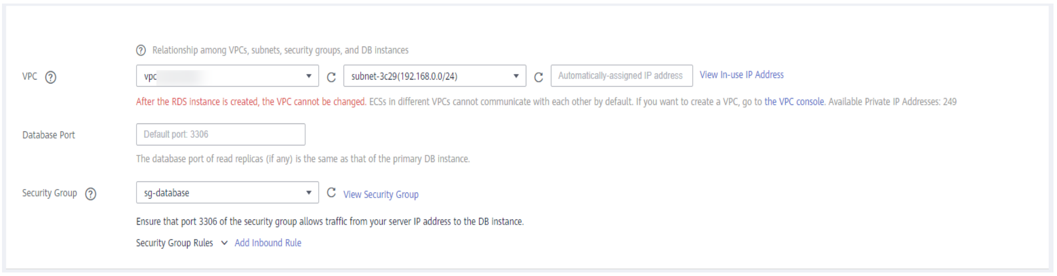
DB Instance Specifications General-enhanced | 2 vCPUs | 4 GB, Maximum Connections: 1500, TPS/QPS: 482 | 9452

Storage Space (GB): 40 GB

RDS provides free backup storage space of the same size as your purchased storage space. After the free backup space is used up, charges are applied based on the OBS pricing details.

Disk Encryption: [✓ Disable] Enable (Recommended)

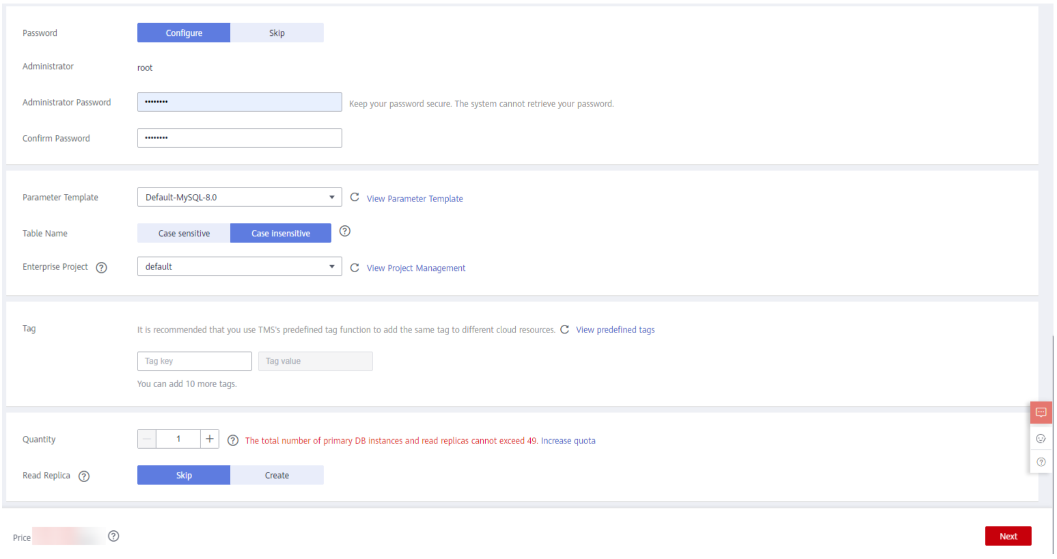
Figure 2-8 Configuring network information as planned



Relationship among VPCs, subnets, security groups, and DB instances

VPC	vpc[redacted] subnet-3c29(192.168.0.0/24) Automatically-assigned IP address View In-use IP Address
	After the RDS instance is created, the VPC cannot be changed. ECSs in different VPCs cannot communicate with each other by default. If you want to create a VPC, go to the VPC console. Available Private IP Addresses: 249
Database Port	Default port: 3306
	The database port of read replicas (if any) is the same as that of the primary DB instance.
Security Group	sg-database View Security Group
	Ensure that port 3306 of the security group allows traffic from your server IP address to the DB instance.
	Security Group Rules Add Inbound Rule

Figure 2-9 Setting an administrator password



Password	Configure / Skip
Administrator	root
Administrator Password	•••••••• Keep your password secure. The system cannot retrieve your password.
Confirm Password	••••••••
Parameter Template	Default-MySQL-8.0 View Parameter Template
Table Name	Case sensitive / Case insensitive
Enterprise Project	default View Project Management
Tag	It is recommended that you use TMS's predefined tag function to add the same tag to different cloud resources. View predefined tags Tag key Tag value You can add 10 more tags.
Quantity	1 The total number of primary DB instances and read replicas cannot exceed 49. Increase quota
Read Replica	Skip / Create

Price [redacted]

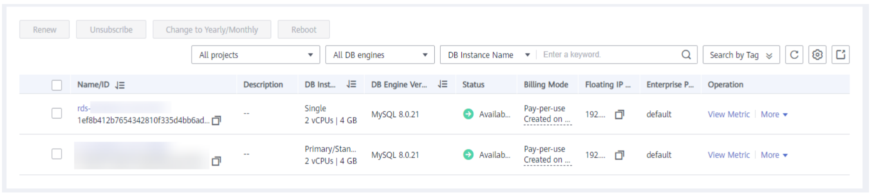
Step 3 Confirm the settings.

- To modify your settings, click **Previous**.
- If there is no need to modify your settings, click **Submit**.

Step 4 Bind an EIP to the created instance.

1. On the **Instances** page, click the instance name to go to the **Basic Information** page.

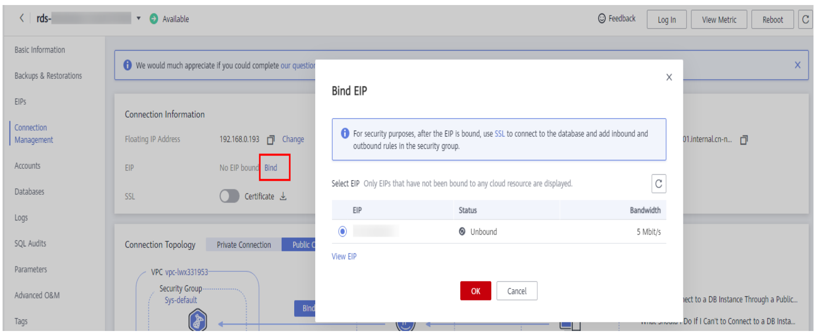
Figure 2-10 Locating your instance in the list



Name/ID	Description	DB Inst...	DB Engine Ver...	Status	Billing Mode	Floating IP ...	Enterprise P...	Operation
rds-[redacted] 1ef8b412b7654342810f335d4bb6ad...	--	Single 2 vCPUs | 4 GB	MySQL 8.0.21	Availab...	Pay-per-use Created on ...	192....	default	View Metric | More
[redacted]	--	Primary/Stan... 2 vCPUs | 4 GB	MySQL 8.0.21	Availab...	Pay-per-use Created on ...	192....	default	View Metric | More

2. In the navigation pane on the left, choose **Connectivity & Security**. In the **Connection Information** area, click **Bind** next to the **EIP** field.
3. In the displayed dialog box, all unbound EIPs are listed. Select the EIP you have created for the instance and click **Yes**.

Figure 2-11 Binding an EIP



We would much appreciate if you could complete our questio

Connection Information

Floating IP Address	192.168.0.193 Change
EIP	No EIP bound Bind
SSL	Certificate

01.internal.cn-n...

Bind EIP

For security purposes, after the EIP is bound, use SSL to connect to the database and add inbound and outbound rules in the security group.

Select EIP Only EIPs that have not been bound to any cloud resource are displayed.

EIP	Status	Bandwidth
[redacted]	Unbound	5 Mbit/s

View EIP

ect to a DB Instance Through a Public...
Do If I Can't to Connect to a DB Insta...

----End

2.3.5 Configuring an RDS for MySQL Instance in the DR Center

2.3.5.1 Creating a VPC and Security Group

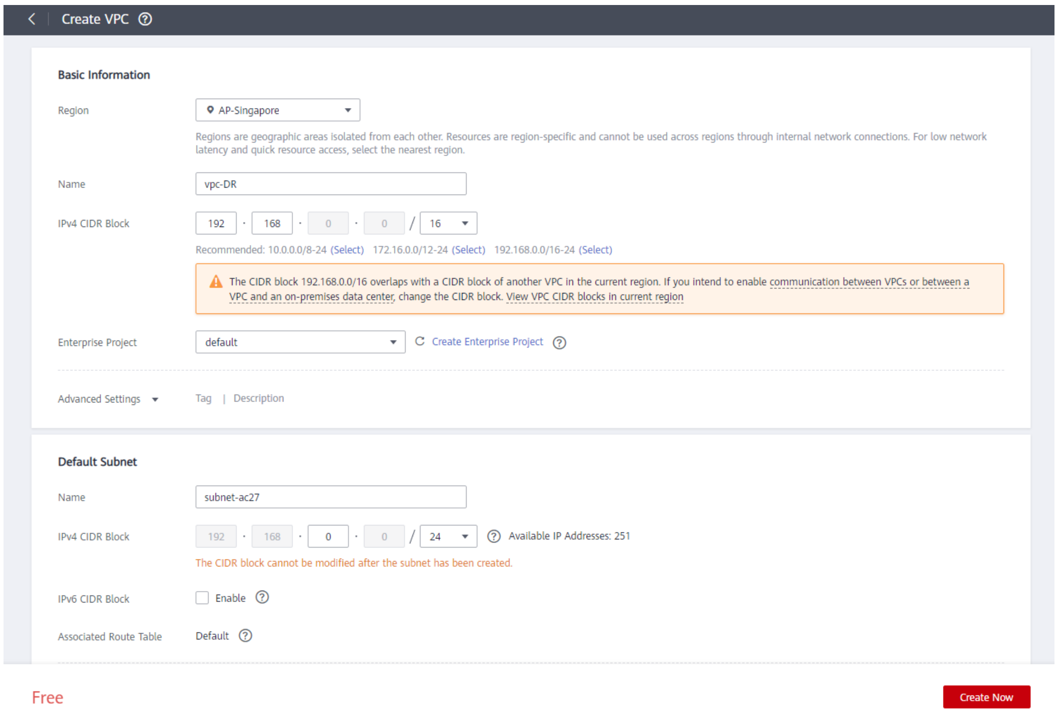
Create a VPC and security group for the DR instance to be configured, ensuring that it is in a different region from the instance created for production center.

Creating a VPC

Step 1 Go to the **Create VPC** page.

Step 2 Configure the basic information, subnet, and IP address. Select **AP-Singapore** for **Region**.

Figure 2-12 Creating a VPC



Create VPC

Basic Information

Region	AP-Singapore Regions are geographic areas isolated from each other. Resources are region-specific and cannot be used across regions through internal network connections. For low network latency and quick resource access, select the nearest region.
Name	vpc-DR
IPv4 CIDR Block	192 · 168 · 0 · 0 / 16 Recommended: 10.0.0.0/8-24 (Select) 172.16.0.0/12-24 (Select) 192.168.0.0/16-24 (Select) The CIDR block 192.168.0.0/16 overlaps with a CIDR block of another VPC in the current region. If you intend to enable communication between VPCs or between a VPC and an on-premises data center, change the CIDR block. View VPC CIDR blocks in current region
Enterprise Project	default Create Enterprise Project
Advanced Settings	Tag | Description

Default Subnet

Name	subnet-ac27
IPv4 CIDR Block	192 · 168 · 0 · 0 / 24 Available IP Addresses: 251 The CIDR block cannot be modified after the subnet has been created.
IPv6 CIDR Block	Enable
Associated Route Table	Default

Free

Create Now

Step 3 Click **Create Now**.

----End

Creating a Security Group

Step 1 Log in to the **management console**.

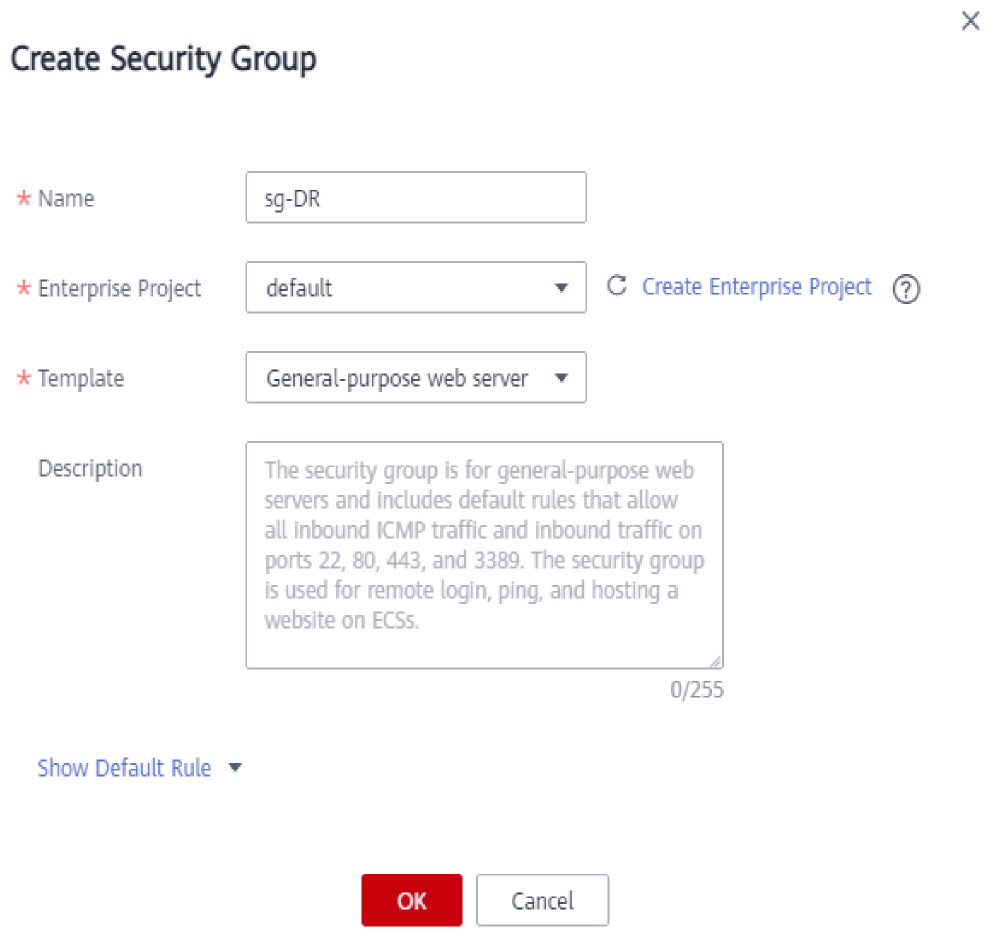
Step 2 Click in the upper left corner of the management console and select **AP-Singapore**.

Step 3 Under the service list, choose **Networking** > **Virtual Private Cloud**.

Step 4 In the navigation pane on the left, choose **Access Control** > **Security Groups**.

Step 5 Click **Create Security Group**.

Figure 2-13 Creating a security group



Create Security Group

* Name	sg-DR
* Enterprise Project	default
* Template	General-purpose web server
Description	The security group is for general-purpose web servers and includes default rules that allow all inbound ICMP traffic and inbound traffic on ports 22, 80, 443, and 3389. The security group is used for remote login, ping, and hosting a website on ECSs.

Step 6 Click **OK**.

----End

2.3.5.2 Creating an RDS for MySQL Instance

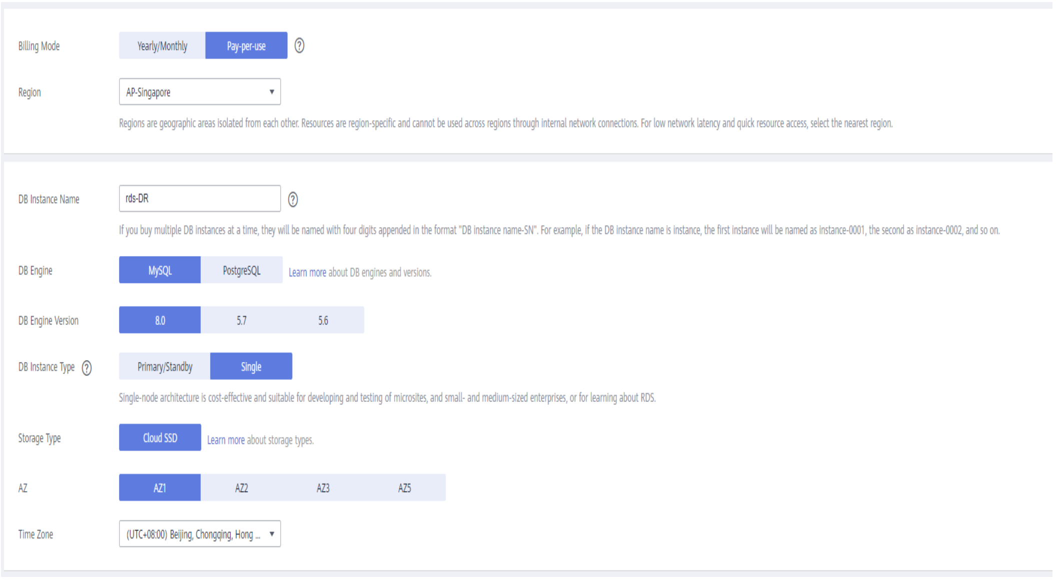
Create an RDS for MySQL instance as a DR instance and select the VPC you configured for the instance.

Procedure

Step 1 Go to the **Buy DB Instance** page.

Step 2 Select **AP-Singapore** for **Region**. Configure instance information and click **Next**.

Figure 2-14 Selecting a DB engine



Billing Mode: Yearly/Monthly [✓ Pay-per-use]

Region: AP-Singapore

Regions are geographic areas isolated from each other. Resources are region-specific and cannot be used across regions through internal network connections. For low network latency and quick resource access, select the nearest region.

DB Instance Name: rds-DR

If you buy multiple DB instances at a time, they will be named with four digits appended in the format "DB instance name-SN". For example, if the DB instance name is instance, the first instance will be named as instance-0001, the second as instance-0002, and so on.

DB Engine: [✓ MySQL] PostgreSQL Learn more about DB engines and versions.

DB Engine Version: [✓ 8.0] 5.7 5.6

DB Instance Type: Primary/Standby [✓ Single]

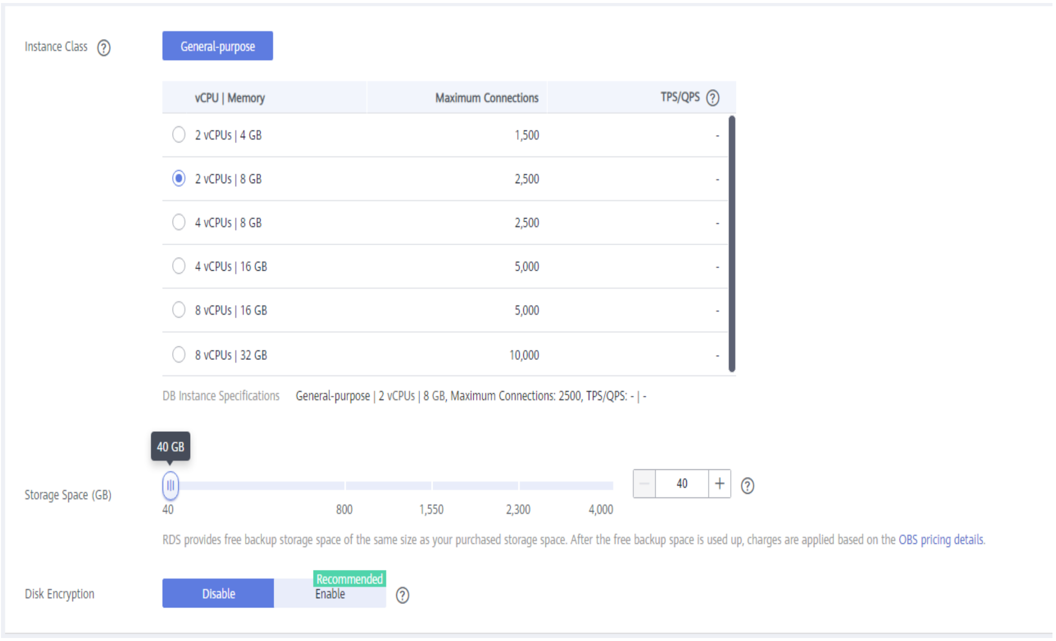
Single-node architecture is cost-effective and suitable for developing and testing of microsites, and small- and medium-sized enterprises, or for learning about RDS.

Storage Type: [✓ Cloud SSD] Learn more about storage types.

AZ: [✓ AZ1] AZ2 AZ3 AZ5

Time Zone: (UTC+08:00) Beijing, Chongqing, Hong ...

Figure 2-15 Selecting specifications



Instance Class: [✓ General-purpose]

	vCPU | Memory	Maximum Connections	TPS/QPS
○	2 vCPUs | 4 GB	1,500	-
●	2 vCPUs | 8 GB	2,500	-
○	4 vCPUs | 8 GB	2,500	-
○	4 vCPUs | 16 GB	5,000	-
○	8 vCPUs | 16 GB	5,000	-
○	8 vCPUs | 32 GB	10,000	-

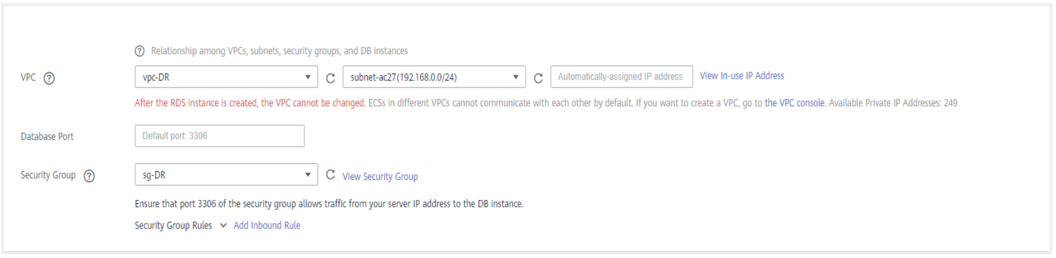
DB Instance Specifications General-purpose | 2 vCPUs | 8 GB, Maximum Connections: 2500, TPS/QPS: - | -

Storage Space (GB): 40 GB

RDS provides free backup storage space of the same size as your purchased storage space. After the free backup space is used up, charges are applied based on the OBS pricing details.

Disk Encryption: [✓ Disable] Enable Recommended

Figure 2-16 Configuring network information as planned



Relationship among VPCs, subnets, security groups, and DB instances

VPC: vpc-DR subnet-ac27(192.168.0.0/24) Automatically-assigned IP address View In-use IP Address

After the RDS instance is created, the VPC cannot be changed. ECSs in different VPCs cannot communicate with each other by default. If you want to create a VPC, go to the VPC console. Available Private IP Addresses: 249

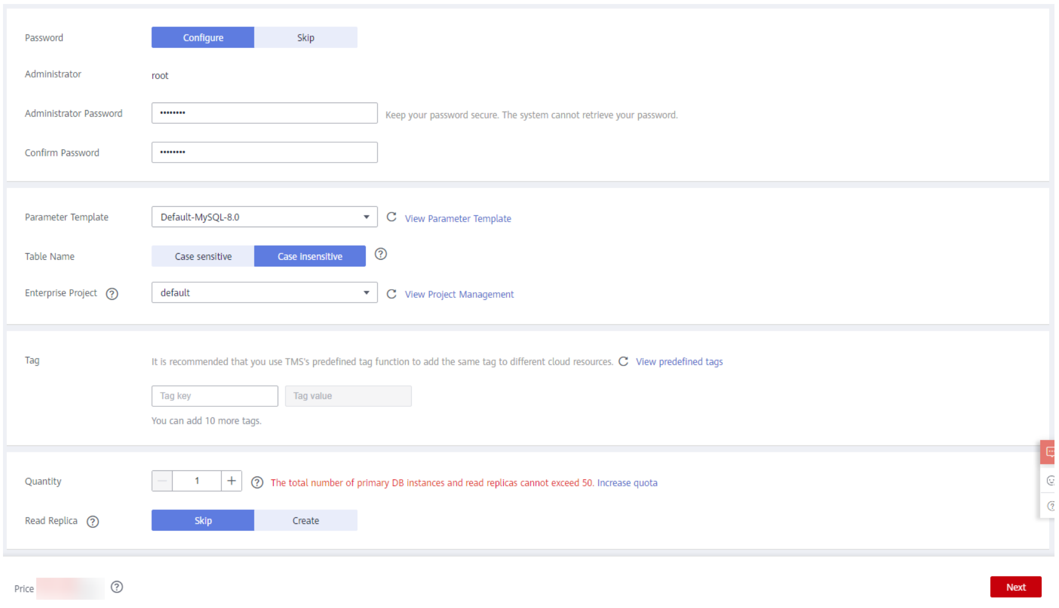
Database Port: Default port: 3306

Security Group: sg-DR View Security Group

Ensure that port 3306 of the security group allows traffic from your server IP address to the DB instance.

Security Group Rules Add Inbound Rule

Figure 2-17 Setting an administrator password



Password: Configure | Skip

Administrator: root

Administrator Password: •••••••• Keep your password secure. The system cannot retrieve your password.

Confirm Password: ••••••••

Parameter Template: Default-MySQL-8.0 View Parameter Template

Table Name: Case sensitive | Case insensitive

Enterprise Project: default View Project Management

Tag: It is recommended that you use TMS's predefined tag function to add the same tag to different cloud resources. View predefined tags

Tag key | Tag value

You can add 10 more tags.

Quantity: 1 The total number of primary DB instances and read replicas cannot exceed 50. Increase quota

Read Replica: Skip | Create

Price

Next

Step 3 Confirm the settings.

- To modify your settings, click **Previous**.
- If there is no need to modify your settings, click **Submit**.

----End

2.3.6 Configuring Remote Disaster Recovery

2.3.6.1 Creating a DRS Disaster Recovery Task

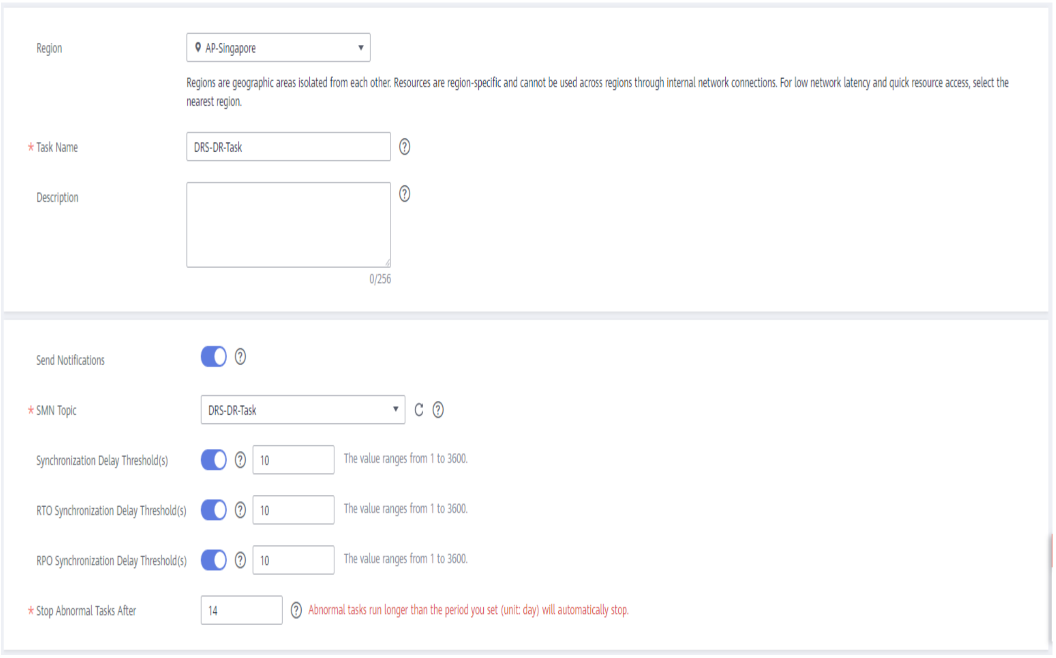
Create a DRS disaster recovery task in the same region as the RDS for MySQL instance configured for the DR center.

Procedure

Step 1 Go to the **Create Disaster Recovery Task** page.

Step 2 Select **AP-Singapore** for **Region**. Set **DR Type** to **Single-active**, **Disaster Recovery Relationship** to **Current cloud as standby**, and **DR DB Instance** to the RDS for MySQL DR instance created in the AP-Singapore region, and click **Create Now**.

Figure 2-18 Configuring basic information



Region: AP-Singapore

Regions are geographic areas isolated from each other. Resources are region-specific and cannot be used across regions through internal network connections. For low network latency and quick resource access, select the nearest region.

* Task Name: DRS-DR-Task

Description: 0/256

Send Notifications: on

* SMN Topic: DRS-DR-Task

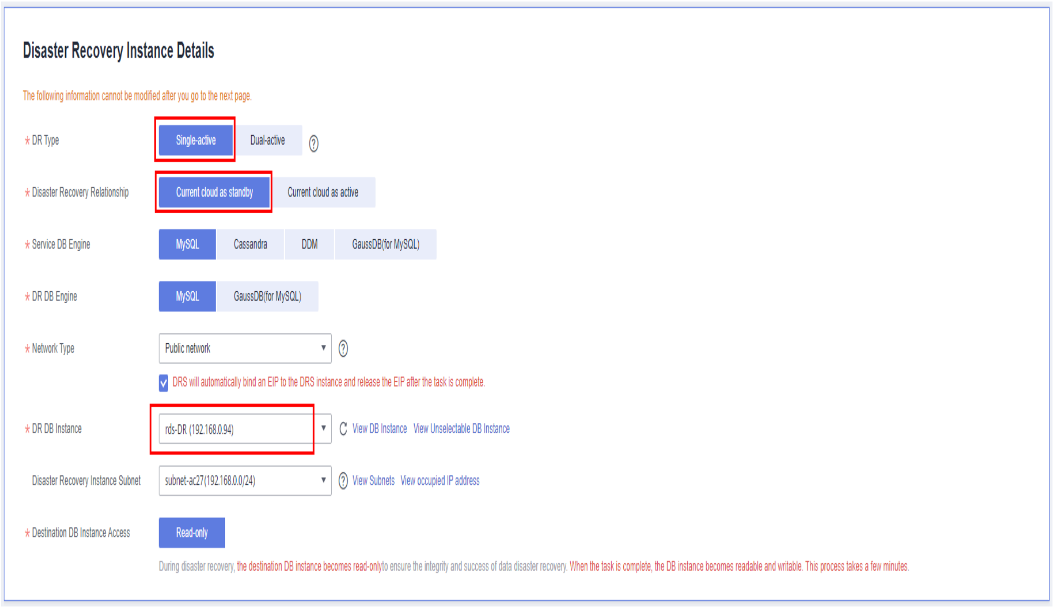
Synchronization Delay Threshold(s): 10 The value ranges from 1 to 3600.

RTO Synchronization Delay Threshold(s): 10 The value ranges from 1 to 3600.

RPO Synchronization Delay Threshold(s): 10 The value ranges from 1 to 3600.

* Stop Abnormal Tasks After: 14 Abnormal tasks run longer than the period you set (unit: day) will automatically stop.

Figure 2-19 Setting DR instance information



Disaster Recovery Instance Details

The following information cannot be modified after you go to the next page.

* DR Type: [✓ Single-active] Dual-active

* Disaster Recovery Relationship: [✓ Current cloud as standby] Current cloud as active

* Service DB Engine: [✓ MySQL] Cassandra DDM GaussDB(for MySQL)

* DR DB Engine: [✓ MySQL] GaussDB(for MySQL)

* Network Type: Public network

✓ DRS will automatically bind an EIP to the DRS instance and release the EIP after the task is complete.

* DR DB Instance: rds-DR (192.168.0.94) View DB Instance View Unselectable DB Instance

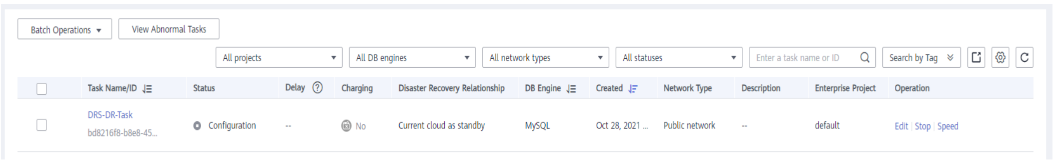
Disaster Recovery Instance Subnet: subnet-ac27(192.168.0.0/24) View Subnets View occupied IP address

* Destination DB Instance Access: [✓ Read-only]

During disaster recovery, the destination DB instance becomes read-onlyto ensure the integrity and success of data disaster recovery. When the task is complete, the DB instance becomes readable and writable. This process takes a few minutes.

Step 3 Return to the **Disaster Recovery Management** page and check the status of the task.

Figure 2-20 Disaster recovery task created



Task Name/ID	Status	Delay	Charging	Disaster Recovery Relationship	DB Engine	Created	Network Type	Description	Enterprise Project	Operation
DRS-DR-Task bd8216f8-b8e8-45...	Configuration	--	No	Current cloud as standby	MySQL	Oct 28, 2021 ...	Public network	--	default	Edit | Stop | Speed

----End

2.3.6.2 Configuring the Disaster Recovery Task

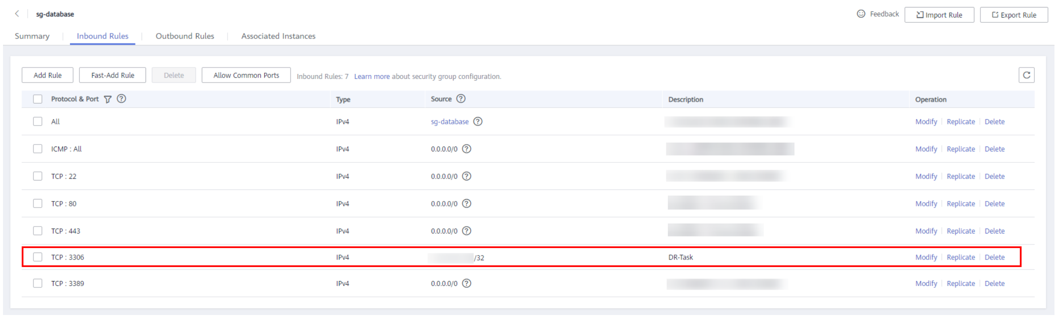
Configure the disaster recovery task, including setting the source and destination databases.

Procedure

Step 1 On the **Disaster Recovery Management** page, locate the created disaster recovery task and click **Edit** in the **Operation** column.

Step 2 Add the EIP of the DRS instance to the inbound rule of the security group associated with the RDS for MySQL instance in the production center, select TCP, and set the port number to that of the RDS for MySQL instance of the production center.

Figure 2-21 Adding a security group rule

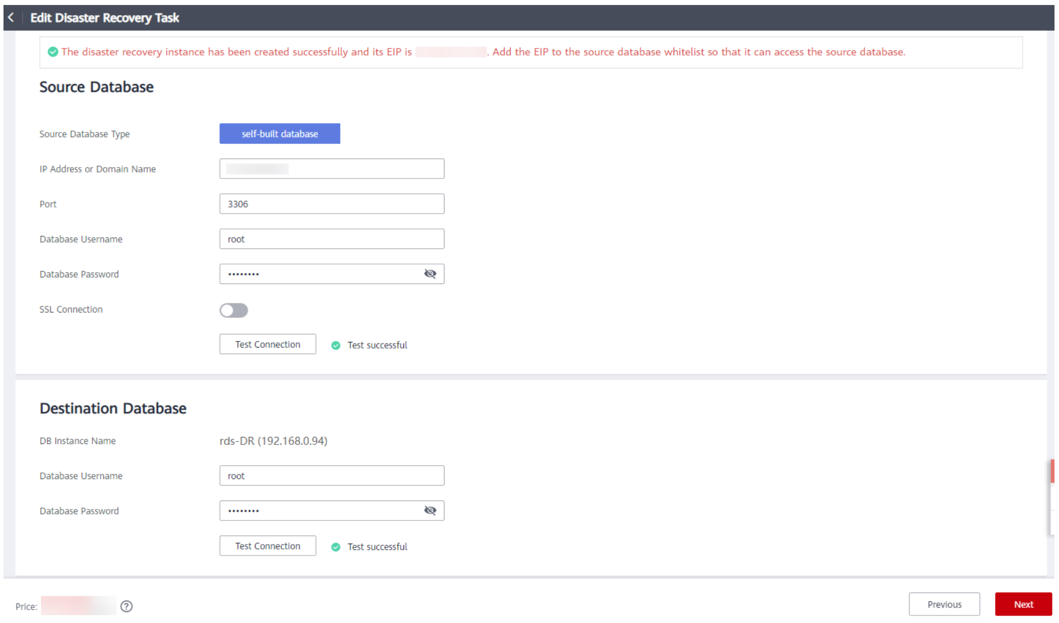


Inbound Rules: 7 Learn more about security group configuration.

Protocol & Port	Type	Source	Description	Operation
All	IPv4	sg-database		Modify | Replicate | Delete
ICMP : All	IPv4	0.0.0.0/0		Modify | Replicate | Delete
TCP : 22	IPv4	0.0.0.0/0		Modify | Replicate | Delete
TCP : 80	IPv4	0.0.0.0/0		Modify | Replicate | Delete
TCP : 443	IPv4	0.0.0.0/0		Modify | Replicate | Delete
TCP : 3306	IPv4	/32	DR-Task	Modify | Replicate | Delete
TCP : 3389	IPv4	0.0.0.0/0		Modify | Replicate | Delete

In the **Source Database** area, set **IP Address or Domain Name** and **Port** to the EIP and port of the RDS for MySQL instance in the production center. When the connection test is successful, click **Next**.

Figure 2-22 Editing a disaster recovery task



The disaster recovery instance has been created successfully and its EIP is . Add the EIP to the source database whitelist so that it can access the source database.

Source Database

Source Database Type	self-built database
IP Address or Domain Name	
Port	3306
Database Username	root
Database Password	••••••••
SSL Connection	
Test Connection	Test successful

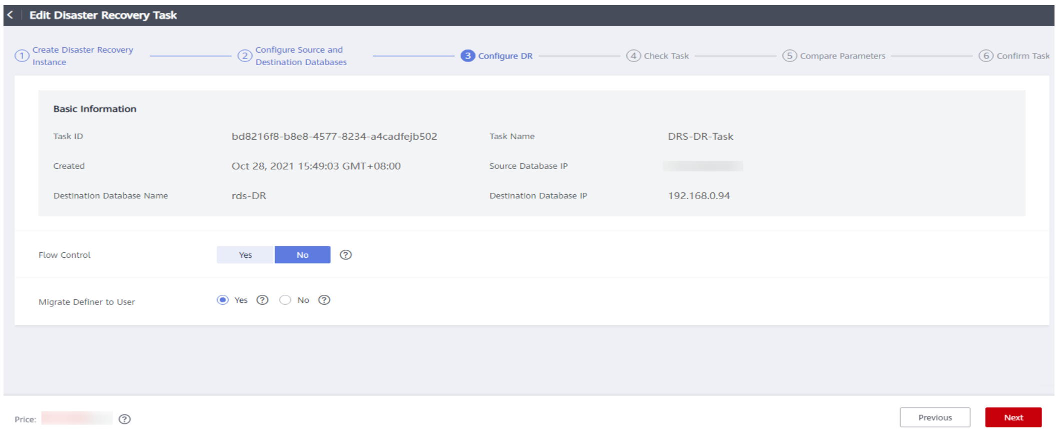
Destination Database

DB Instance Name	rds-DR (192.168.0.94)
Database Username	root
Database Password	••••••••
Test Connection	Test successful

Price:

Step 3 Configure the flow control and click **Next**.

Figure 2-23 Configuring flow control



Edit Disaster Recovery Task

1 Create Disaster Recovery Instance — 2 Configure Source and Destination Databases — 3 Configure DR — 4 Check Task — 5 Compare Parameters — 6 Confirm Task

Basic Information

Task ID	bd8216f8-b8e8-4577-8234-a4cadfejb502	Task Name	DRS-DR-Task
Created	Oct 28, 2021 15:49:03 GMT+08:00	Source Database IP	[redacted]
Destination Database Name	rds-DR	Destination Database IP	192.168.0.94

Flow Control: Yes [✓ No]

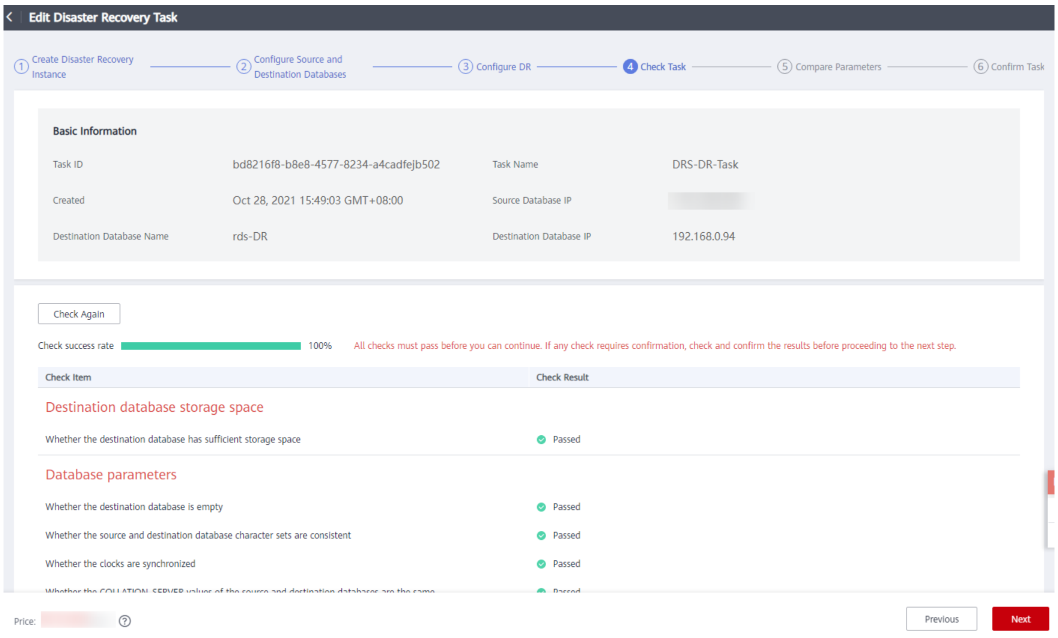
Migrate Definer to User: [✓ Yes] No

Price: [redacted]

Previous Next

Step 4 Check the disaster recovery task. When the check success rate reaches 100%, click **Next**.

Figure 2-24 Checking the task



Edit Disaster Recovery Task

1 Create Disaster Recovery Instance — 2 Configure Source and Destination Databases — 3 Configure DR — 4 Check Task — 5 Compare Parameters — 6 Confirm Task

Basic Information

Task ID	bd8216f8-b8e8-4577-8234-a4cadfejb502	Task Name	DRS-DR-Task
Created	Oct 28, 2021 15:49:03 GMT+08:00	Source Database IP	[redacted]
Destination Database Name	rds-DR	Destination Database IP	192.168.0.94

Check Again

Check success rate 100% All checks must pass before you can continue. If any check requires confirmation, check and confirm the results before proceeding to the next step.

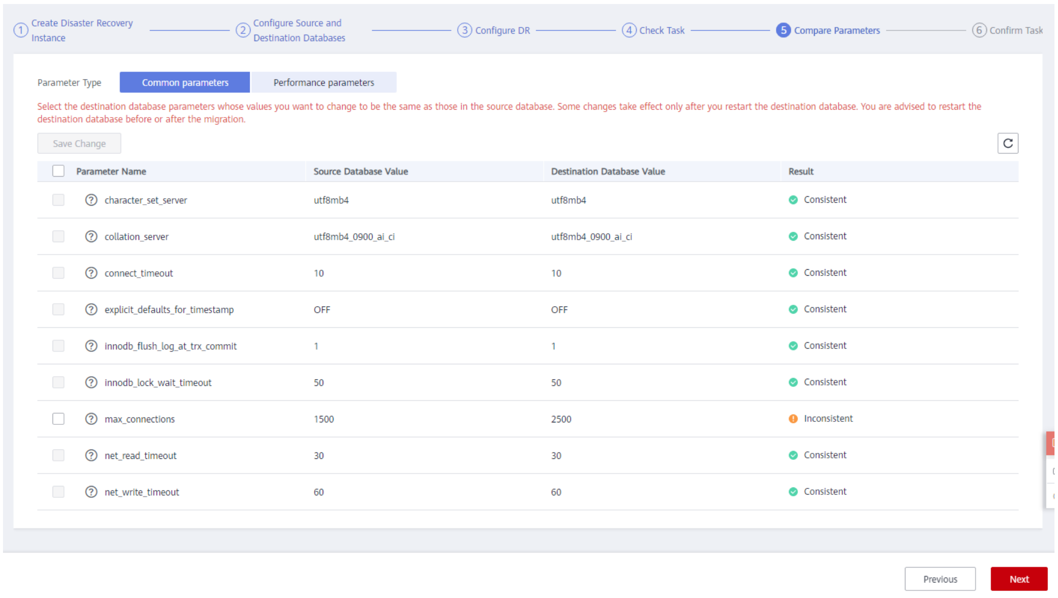
Check Item	Check Result
Destination database storage space	
Whether the destination database has sufficient storage space	✓ Passed
Database parameters	
Whether the destination database is empty	✓ Passed
Whether the source and destination database character sets are consistent	✓ Passed
Whether the clocks are synchronized	✓ Passed
Whether the COLLATION_SERVER values of the source and destination databases are the same	✓ Passed

Price: [redacted]

Previous Next

Step 5 Configure parameters and click **Next**.

Figure 2-25 Configuring parameters

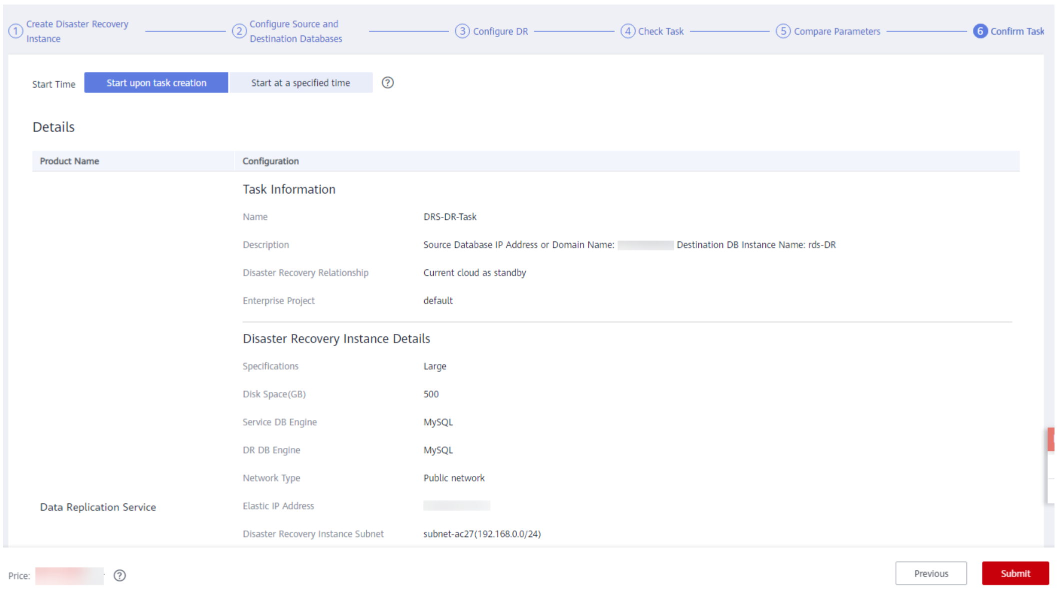


Select the destination database parameters whose values you want to change to be the same as those in the source database. Some changes take effect only after you restart the destination database. You are advised to restart the destination database before or after the migration.

Parameter Name	Source Database Value	Destination Database Value	Result
character_set_server	utf8mb4	utf8mb4	Consistent
collation_server	utf8mb4_0900_ai_ci	utf8mb4_0900_ai_ci	Consistent
connect_timeout	10	10	Consistent
explicit_defaults_for_timestamp	OFF	OFF	Consistent
innodb_flush_log_at_trx_commit	1	1	Consistent
innodb_lock_wait_timeout	50	50	Consistent
max_connections	1500	2500	Inconsistent
net_read_timeout	30	30	Consistent
net_write_timeout	60	60	Consistent

Step 6 Configure **Start Time** and click **Submit**.

Figure 2-26 Starting the task



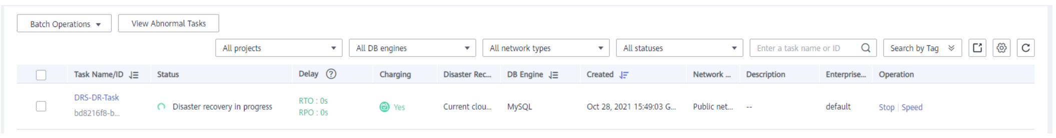
Details

Product Name	Configuration	
	Task Information	
	Name	DRS-DR-Task
	Description	Source Database IP Address or Domain Name: [redacted] Destination DB Instance Name: rds-DR
	Disaster Recovery Relationship	Current cloud as standby
	Enterprise Project	default
	Disaster Recovery Instance Details	
	Specifications	Large
	Disk Space(GB)	500
	Service DB Engine	MySQL
	DR DB Engine	MySQL
	Network Type	Public network
Data Replication Service	Elastic IP Address	[redacted]
	Disaster Recovery Instance Subnet	subnet-ac27(192.168.0.0/24)

Price: [redacted]

Step 7 On the **Disaster Recovery Management** page, check the task status. The status is **Disaster recovery in progress**.

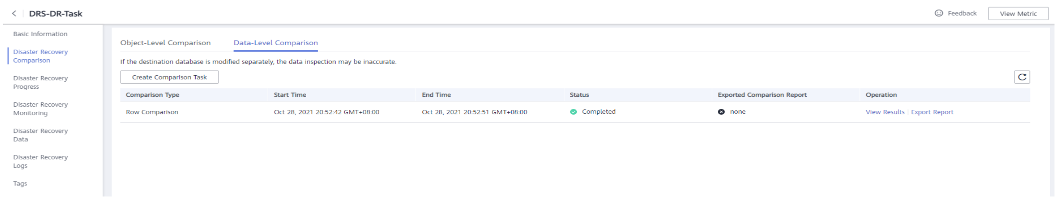
Figure 2-27 Checking the task status



Task Name/ID	Status	Delay	Charging	Disaster Rec...	DB Engine	Created	Network ...	Description	Enterprise...	Operation
DRS-DR-Task bd8216f8-b...	Disaster recovery in progress	RTO : 0s RPO : 0s	Yes	Current clou...	MySQL	Oct 28, 2021 15:49:03 G...	Public net...	--	default	Stop | Speed

Step 8 Click the task name to go to the **Basic Information** page and confirm the disaster recovery comparison, disaster recovery progress, and disaster recovery data.

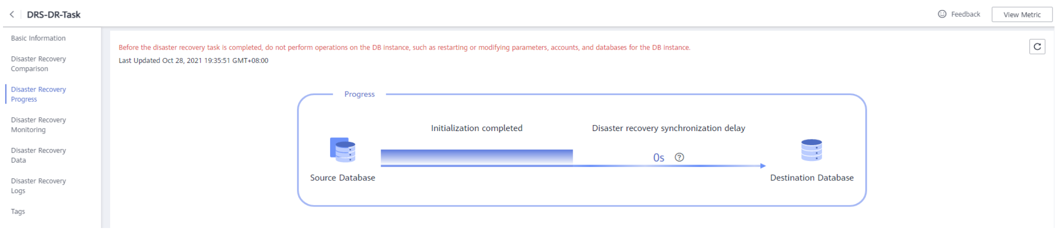
Figure 2-28 Disaster recovery comparison



If the destination database is modified separately, the data inspection may be inaccurate.

Comparison Type	Start Time	End Time	Status	Exported Comparison Report	Operation
Row Comparison	Oct 28, 2021 20:52:42 GMT+08:00	Oct 28, 2021 20:52:51 GMT+08:00	Completed	none	View Results | Export Report

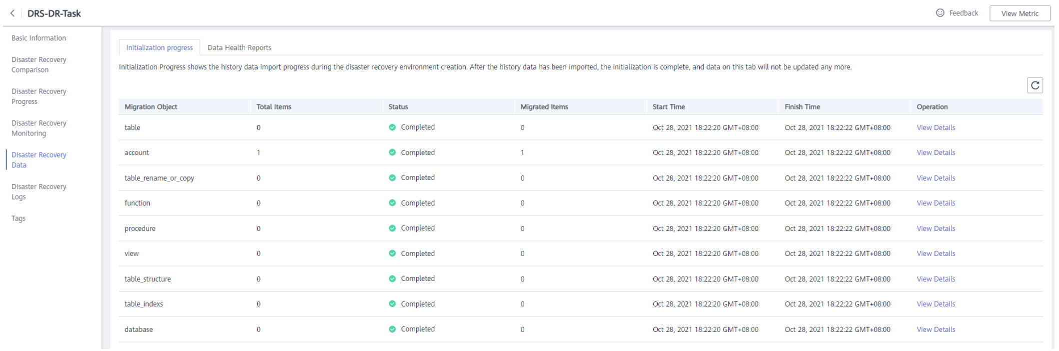
Figure 2-29 Disaster recovery progress



Before the disaster recovery task is completed, do not perform operations on the DB instance, such as restarting or modifying parameters, accounts, and databases for the DB instance.

Last Updated Oct 28, 2021 19:35:51 GMT+08:00

Figure 2-30 Disaster recovery data



Initialization Progress shows the history data import progress during the disaster recovery environment creation. After the history data has been imported, the initialization is complete, and data on this tab will not be updated any more.

Migration Object	Total Items	Status	Migrated Items	Start Time	Finish Time	Operation
table	0	Completed	0	Oct 28, 2021 18:22:20 GMT+08:00	Oct 28, 2021 18:22:22 GMT+08:00	View Details
account	1	Completed	1	Oct 28, 2021 18:22:20 GMT+08:00	Oct 28, 2021 18:22:22 GMT+08:00	View Details
table_rename_or_copy	0	Completed	0	Oct 28, 2021 18:22:20 GMT+08:00	Oct 28, 2021 18:22:22 GMT+08:00	View Details
function	0	Completed	0	Oct 28, 2021 18:22:20 GMT+08:00	Oct 28, 2021 18:22:22 GMT+08:00	View Details
procedure	0	Completed	0	Oct 28, 2021 18:22:20 GMT+08:00	Oct 28, 2021 18:22:22 GMT+08:00	View Details
view	0	Completed	0	Oct 28, 2021 18:22:20 GMT+08:00	Oct 28, 2021 18:22:22 GMT+08:00	View Details
table_structure	0	Completed	0	Oct 28, 2021 18:22:20 GMT+08:00	Oct 28, 2021 18:22:22 GMT+08:00	View Details
table_indexs	0	Completed	0	Oct 28, 2021 18:22:20 GMT+08:00	Oct 28, 2021 18:22:22 GMT+08:00	View Details
database	0	Completed	0	Oct 28, 2021 18:22:20 GMT+08:00	Oct 28, 2021 18:22:22 GMT+08:00	View Details

----End

2.3.6.3 Performing a Primary/Standby Switchover

If the source database in the production center is faulty, manually switch the DR instance to the read/write state. Then, data is written to the DR instance and synchronized to the source database.

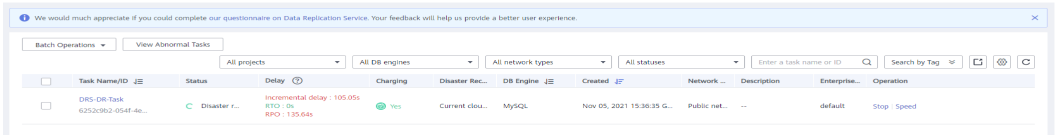
Procedure

Step 1 Find that the source database in the production center is faulty. For example, the source database cannot be connected, the source database execution is slow, or the CPU usage is high.

Step 2 Receive an SMN email notification

Step 3 Check the delay of the DR task.

Figure 2-31 Delay exception



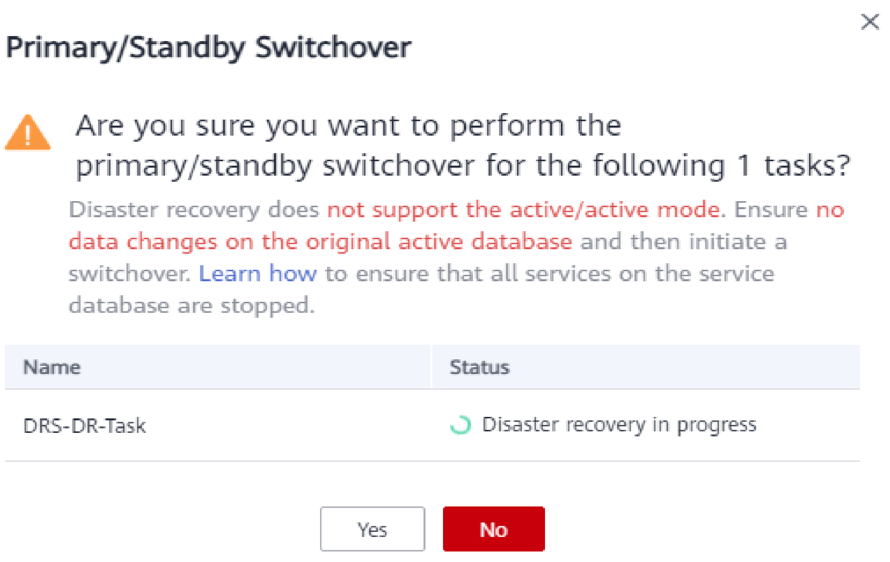
We would much appreciate if you could complete our questionnaire on Data Replication Service. Your feedback will help us provide a better user experience.

	Task Name/ID	Status	Delay	Charging	Disaster Rec...	DB Engine	Created	Network ...	Description	Enterprise...	Operation
☐	DRS-DR-Task 6252c9b2-054f-4e...	Disaster r...	Incremental delay : 105.05s RTO : 0s RPO : 135.64s	Yes	Current clou...	MySQL	Nov 05, 2021 15:36:35 G...	Public net...	--	default	Stop | Speed

Step 4 Check that the services of the source database have been stopped. For details, see **How Do I Ensure that All Services on the Database Are Stopped?**

Step 5 Select the task, click the **Batch Operation** drop-down box in the upper left corner and select **Primary/Standby Switchover**.

Figure 2-32 Primary/standby switchover



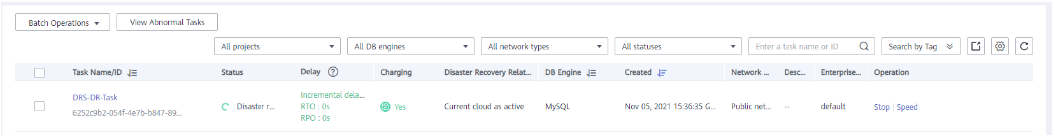
Primary/Standby Switchover

Are you sure you want to perform the primary/standby switchover for the following 1 tasks?

Disaster recovery does not support the active/active mode. Ensure no data changes on the original active database and then initiate a switchover. Learn how to ensure that all services on the service database are stopped.

Name	Status
DRS-DR-Task	Disaster recovery in progress

Figure 2-33 Switchover completed



Task Name/ID	Status	Delay	Charging	Disaster Recovery Relat...	DB Engine	Created	Network ...	Desc...	Enterprise...	Operation
DRS-DR-Task 6252c9b2-054f-4e7b-b847-89...	Disaster r...	Incremental dela... RTO : 0s RPO : 0s	Yes	Current cloud as active	MySQL	Nov 05, 2021 15:36:35 G...	Public net...	--	default	Stop | Speed

Step 6 Change the database IP address on your application and use it to connect to the database. Then data is properly read from and written to the database.

----End

2.4 Migrating MySQL Databases from Other Clouds to RDS for MySQL

2.4.1 Overview

Scenarios

This best practice includes the following tasks:

- Create an RDS for MySQL instance.
- Migrate data from a MySQL database on other clouds to RDS for MySQL.

Prerequisites

- You have registered with Huawei Cloud.
- Your account balance is at least $0 USD.

Service List

- Virtual Private Cloud (VPC)
- RDS
- Data Replication Service (DRS)

Before You Start

- The resource planning in this best practice is for demonstration only. Adjust it as needed.
- All settings in this best practice are for reference only. For more information about MySQL migration, see **From MySQL to MySQL**.

2.4.2 Resource Planning

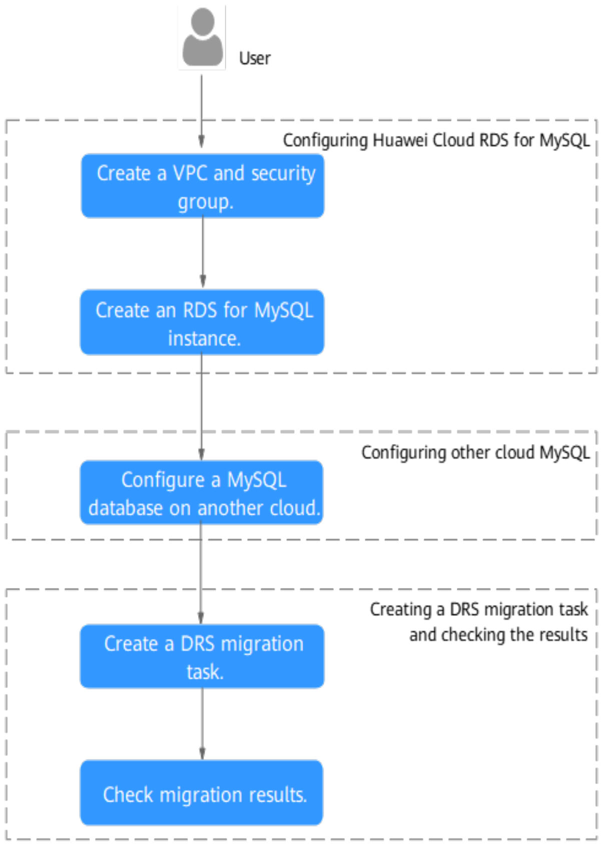
Table 2-3 Resource planning

Category	Subcategory	Planned Value	Description
VPC	VPC name	vpc-src-172	Specify a name that is easy to identify.
	Region	Test region	To achieve lower network latency, select the region nearest to you.
	AZ	AZ3	-
	Subnet	172.16.0.0/16	Select a subnet with sufficient network resources.
	Subnet name	subnet-src-172	Specify a name that is easy to identify.
MySQL on another cloud	Database version	MySQL 5.7	-
	IP address	10.154.217.42	Enter an IP address.
	Port	3306	-
RDS for MySQL instance	Instance name	rds-mysql	Specify a name that is easy to identify.
	DB engine version	MySQL 5.7	-

Category	Subcategory	Planned Value	Description
	Instance type	Single	A single instance is used in this example. To improve service reliability, select a primary/ standby instance.
	Storage type	Cloud SSD	-
	AZ	AZ1	AZ1 is selected in this example. To improve service reliability, select the primary/standby instance type and deploy the primary and standby instances in different AZs.
	Instance class	General-purpose 2 vCPUs | 8 GB	-
DRS migration task	Task name	DRS-mysql	Specify a name that is easy to identify.
	Source DB engine	MySQL	-
	Destination DB engine	MySQL	-
	Network type	Public network	Public network is used in this example.

2.4.3 Operation Process

Figure 2-34 Flowchart


User
Configuring Huawei Cloud RDS for MySQL
Create a VPC and security group.
Create an RDS for MySQL instance.
Configuring other cloud MySQL
Configure a MySQL database on another cloud.
Creating a DRS migration task and checking the results
Create a DRS migration task.
Check migration results.

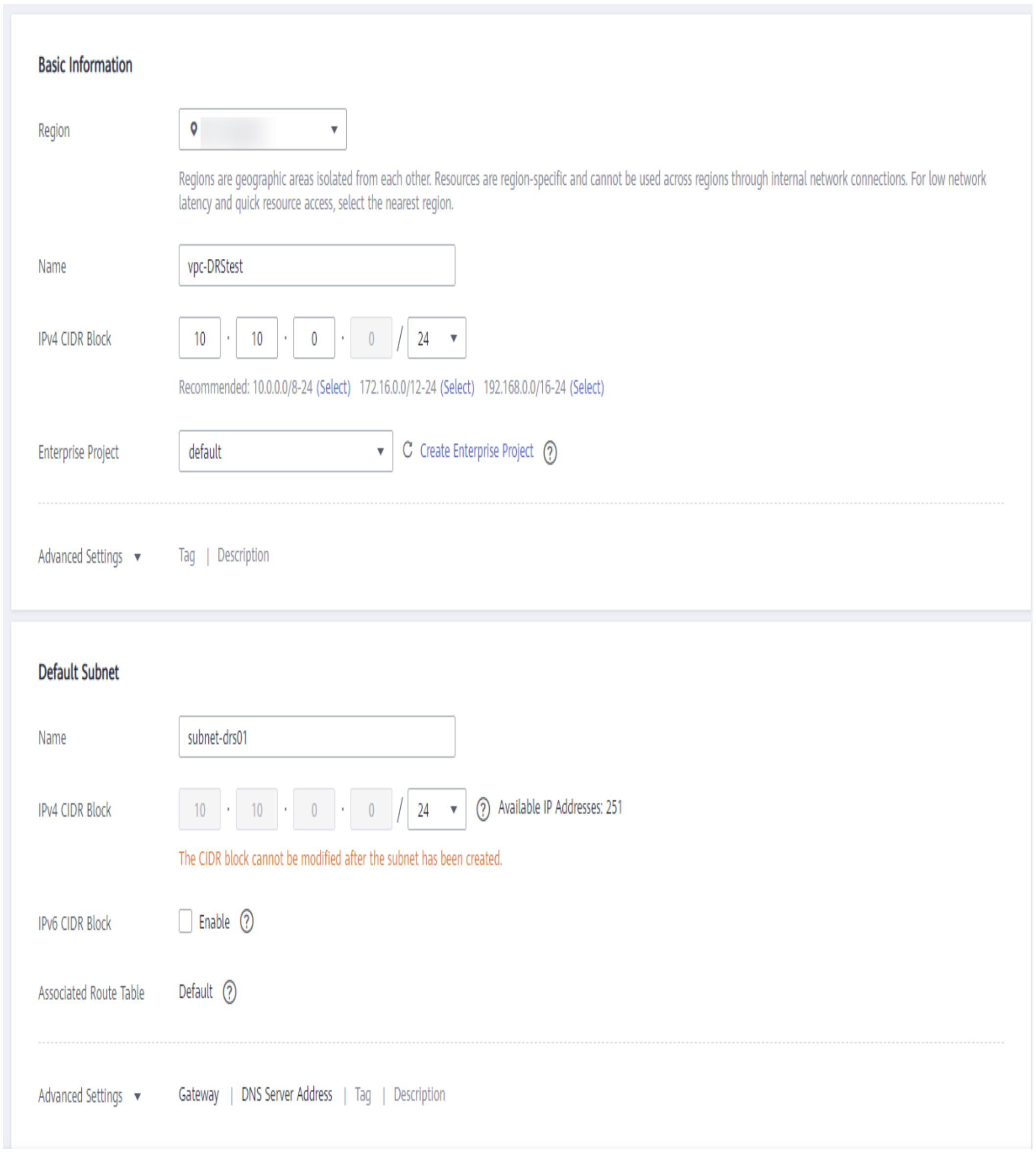
2.4.4 Creating a VPC and Security Group

Create a VPC and security group for an RDS for MySQL instance

Creating a VPC

Step 1 Go to the **Create VPC** page.

Step 2 Configure the basic information, subnet, and IP address.



Basic Information

Region

Regions are geographic areas isolated from each other. Resources are region-specific and cannot be used across regions through internal network connections. For low network latency and quick resource access, select the nearest region.

Name: vpc-DRStest

IPv4 CIDR Block: 10 . 10 . 0 . 0 / 24

Recommended: 10.0.0.0/8-24 (Select) 172.16.0.0/12-24 (Select) 192.168.0.0/16-24 (Select)

Enterprise Project: default   Create Enterprise Project

Advanced Settings   Tag | Description

Default Subnet

Name: subnet-drs01

IPv4 CIDR Block: 10 . 10 . 0 . 0 / 24   Available IP Addresses: 251

The CIDR block cannot be modified after the subnet has been created.

IPv6 CIDR Block: Enable

Associated Route Table: Default

Advanced Settings   Gateway | DNS Server Address | Tag | Description

Step 3 Click **Create Now**.

Step 4 Return to the VPC list and check whether the VPC is created.

If the VPC status becomes available, the VPC has been created.

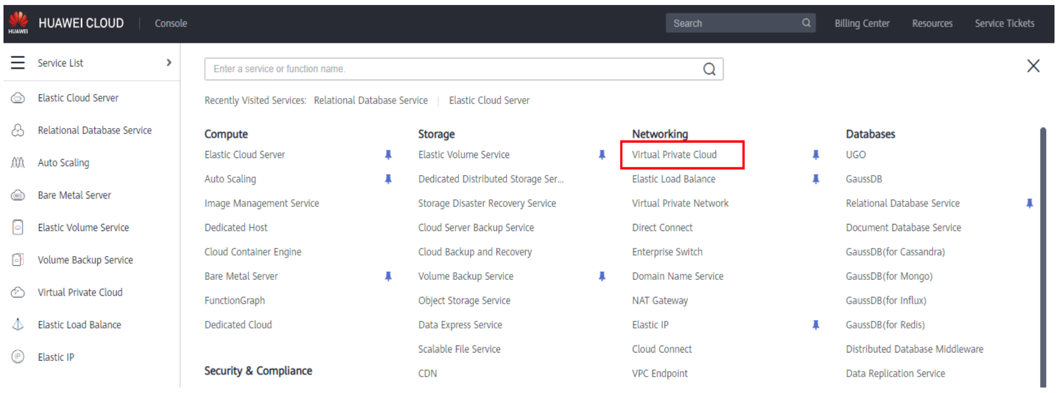
----End

Creating a Security Group

Step 1 Log in to the **management console**.

Step 2 Click in the upper left corner of the management console and select **CN-Hong Kong**.

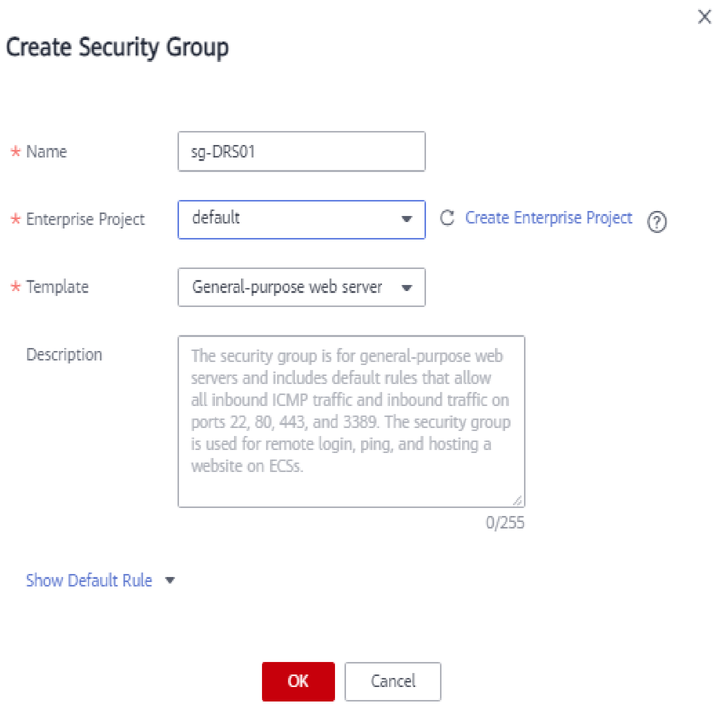
Step 3 Under the service list, choose **Networking** > **Virtual Private Cloud**.



Step 4 In the navigation pane, choose **Access Control** > **Security Groups**.

Step 5 Click **Create Security Group**.

Step 6 Configure parameters as needed.



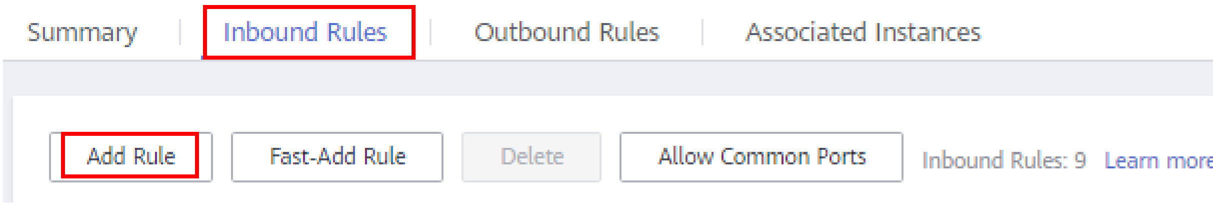
Create Security Group

* Name	sg-DRS01
* Enterprise Project	default
* Template	General-purpose web server
Description	The security group is for general-purpose web servers and includes default rules that allow all inbound ICMP traffic and inbound traffic on ports 22, 80, 443, and 3389. The security group is used for remote login, ping, and hosting a website on ECSs.

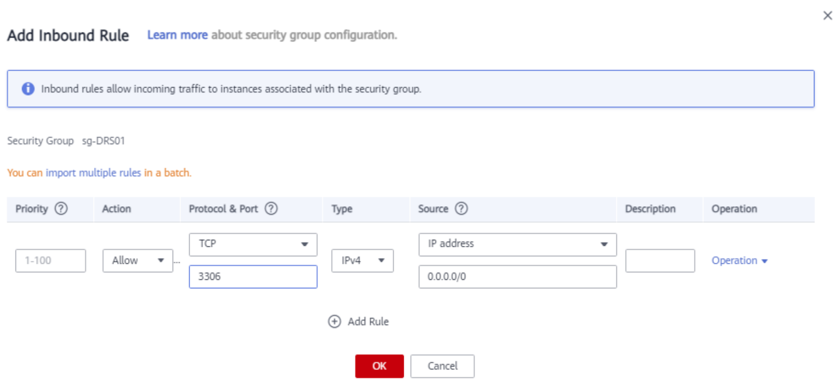
Step 7 Click **OK**.

Step 8 Return to the security group list and click the security group name (**sg-DRS01** in this example).

Step 9 Click the **Inbound Rules** tab, and then click **Add Rule**.



Step 10 Configure an inbound rule to allow access from database port **3306**.



Add Inbound Rule Learn more about security group configuration.

Inbound rules allow incoming traffic to instances associated with the security group.

Security Group sg-DRS01

You can import multiple rules in a batch.

Priority	Action	Protocol & Port	Type	Source	Description	Operation
1-100	Allow	TCP 3306	IPv4	IP address 0.0.0.0/0		Operation

Add Rule

OK Cancel

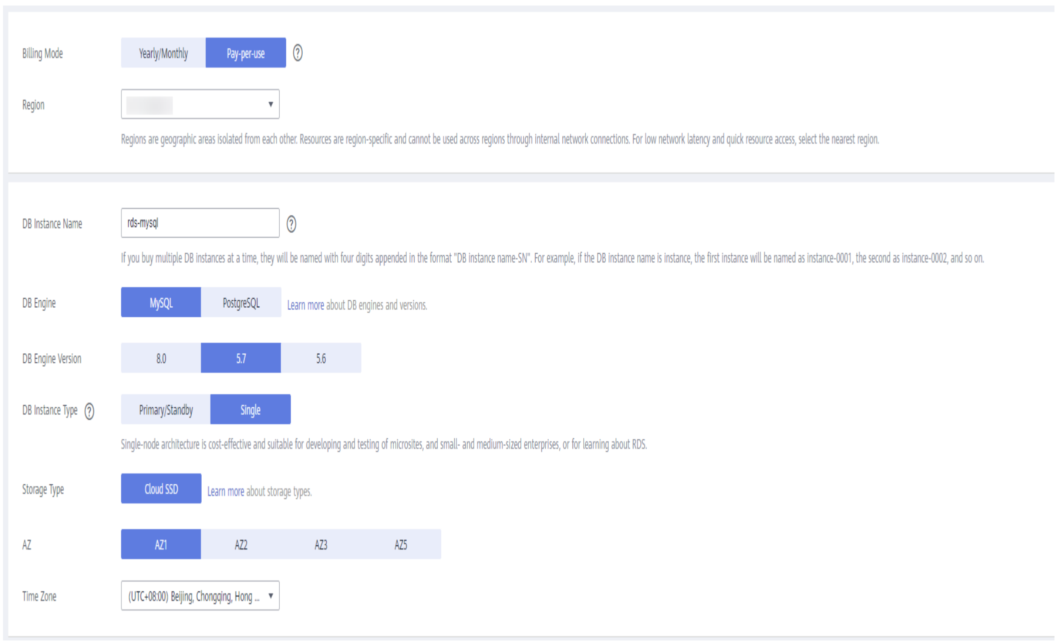
----End

2.4.5 Creating an RDS for MySQL Instance

Create an RDS for MySQL instance, and select the VPC and security group you configured for the instance.

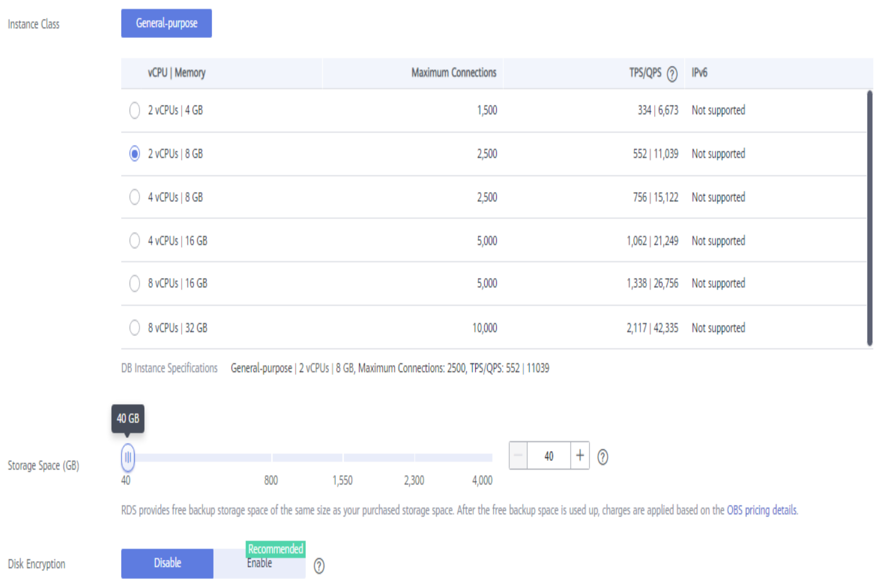
Step 1 Go to the **Buy DB Instance** page.

Step 2 Configure the instance name and basic information. Select **CN-Hong Kong** for **Region**.



Billing Mode	Yearly/Monthly [✓ Pay-per-use]
Region	
	Regions are geographic areas isolated from each other. Resources are region-specific and cannot be used across regions through internal network connections. For low network latency and quick resource access, select the nearest region.
DB Instance Name	rds-mysql
	If you buy multiple DB instances at a time, they will be named with four digits appended in the format "DB instance name-SN". For example, if the DB instance name is instance, the first instance will be named as instance-0001, the second as instance-0002, and so on.
DB Engine	[✓ MySQL] PostgreSQL Learn more about DB engines and versions.
DB Engine Version	8.0 [✓ 5.7] 5.6
DB Instance Type	Primary/Standby [✓ Single]
	Single-node architecture is cost-effective and suitable for developing and testing of microsites, and small- and medium-sized enterprises, or for learning about RDS.
Storage Type	[✓ Cloud SSD] Learn more about storage types.
AZ	[✓ AZ1] AZ2 AZ3 AZ5
Time Zone	(UTC+08:00) Beijing, Chongqing, Hong ...

Step 3 Configure instance specifications.



Instance Class: General-purpose

vCPU | Memory	Maximum Connections	TPS/QPS	IPv6
2 vCPUs | 4 GB	1,500	334 | 6,673	Not supported
● 2 vCPUs | 8 GB	2,500	552 | 11,039	Not supported
4 vCPUs | 8 GB	2,500	756 | 15,122	Not supported
4 vCPUs | 16 GB	5,000	1,062 | 21,249	Not supported
8 vCPUs | 16 GB	5,000	1,338 | 26,756	Not supported
8 vCPUs | 32 GB	10,000	2,117 | 42,335	Not supported

DB Instance Specifications: General-purpose | 2 vCPUs | 8 GB, Maximum Connections: 2500, TPS/QPS: 552 | 11039

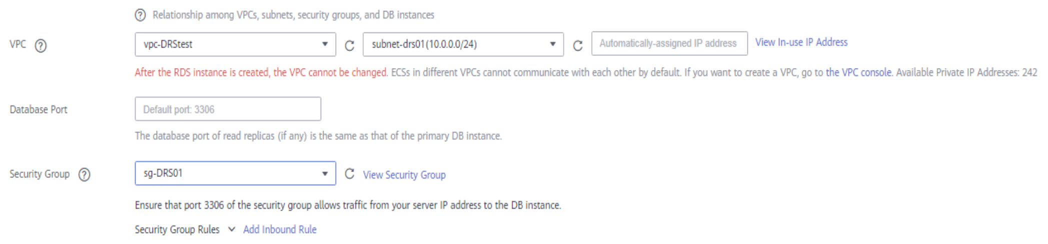
Storage Space (GB): 40 GB

RDS provides free backup storage space of the same size as your purchased storage space. After the free backup space is used up, charges are applied based on the OBS pricing details.

Disk Encryption: [✓ Disable] Enable

Step 4 Select a VPC and security group for the instance and configure the database port.

The VPC and security group have been created in **Creating a VPC and Security Group**.



Relationship among VPCs, subnets, security groups, and DB instances

VPC: vpc-DRStest; subnet-drs01(10.0.0.0/24); Automatically-assigned IP address

After the RDS instance is created, the VPC cannot be changed. ECSs in different VPCs cannot communicate with each other by default. If you want to create a VPC, go to the VPC console. Available Private IP Addresses: 242

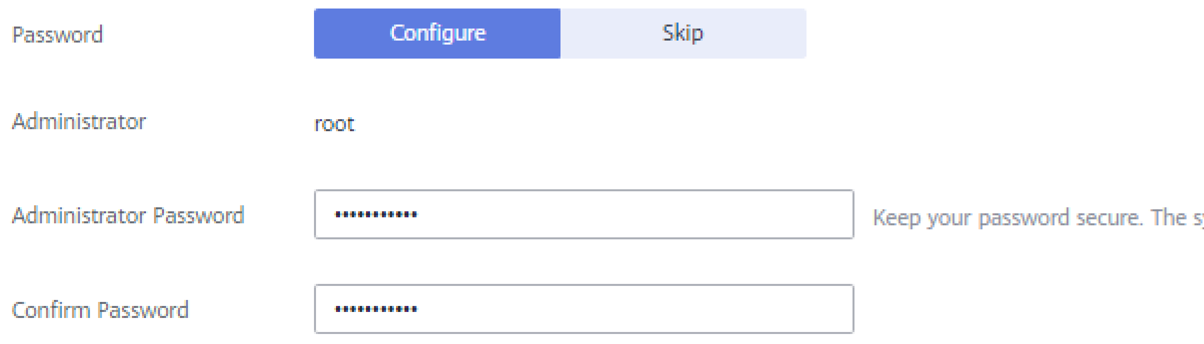
Database Port: Default port: 3306

The database port of read replicas (if any) is the same as that of the primary DB instance.

Security Group: sg-DRS01

Ensure that port 3306 of the security group allows traffic from your server IP address to the DB instance.

Step 5 Configure the password.



Password: [✓ Configure] Skip

Administrator: root

Administrator Password: ••••••••••••

Confirm Password: ••••••••••••

Keep your password secure. The s

Step 6 Click **Next**.

Step 7 Confirm the settings.

- To modify your settings, click **Previous**.
- If you do not need to modify your settings, click **Submit**.

Step 8 Return to the instance list. If the instance status becomes available, the instance has been created.

----End

2.4.6 Configuring a MySQL Instance on Another Cloud

Prerequisites

- You have purchased a MySQL instance from another cloud vendor platform.
- Your account has the migration permissions listed in **Permission Requirements**.

Permission Requirements

Table 2-4 lists the permissions required for migrating data from a MySQL instance on another cloud to RDS for MySQL using DRS. For details about the permissions, see **Which MySQL Permissions Are Required for DRS?**

Table 2-4 Migration permissions

Database	Full Migration Permission	Full+Incremental Migration Permission
Source database (MySQL)	SELECT, SHOW VIEW, and EVENT	SELECT, SHOW VIEW, EVENT, LOCK TABLES, REPLICATION SLAVE, and REPLICATION CLIENT

Network Configuration

You need to enable public accessibility for the source database.

Whitelist Settings

The EIP of the DRS replication instance must be on the whitelist of the source database for the connectivity between the DRS replication instance and the source database. To obtain the EIP of the DRS replication instance, see **Step 3** in **Creating a DRS Migration Task**. This method of configuring a whitelist varies depending on the cloud database vendors. For details, see their official documents.

2.4.7 Cloud Migration

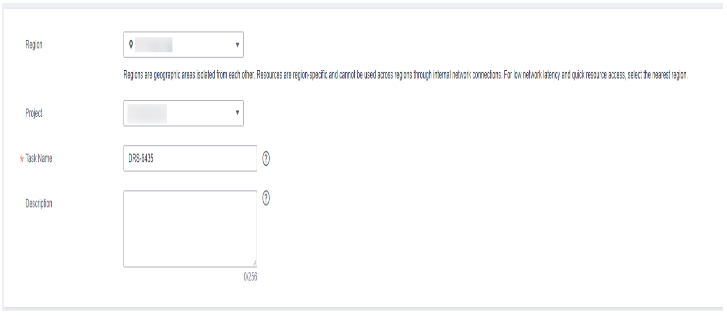
2.4.7.1 Creating a DRS Migration Task

Creating a Migration Task

Step 1 Go to the **Create Migration Task** page.

Step 2 Configure parameters as needed.

1. Enter the migration task name. Select the region hosting the destination DB instance for **Region**.



Region [redacted]

Regions are geographic areas isolated from each other. Resources are region-specific and cannot be used across regions through internal network connections. For low network latency and quick resource access, select the nearest region.

Project [redacted]

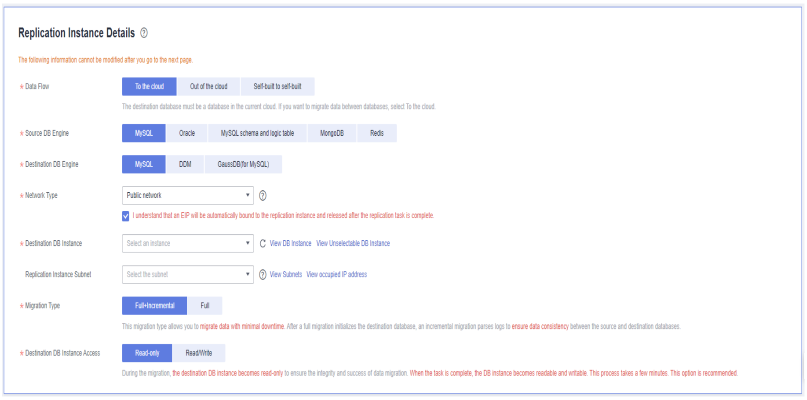
* Task Name: DRS-6435

Description

0/256

2. Configure the replication instance information.

   Select the RDS instance created in **Creating an RDS for MySQL Instance** as the destination database.



Replication Instance Details

The following information cannot be modified after you go to the next page.

* Data Flow: [✓ To the cloud] Out of the cloud | Self-built to self-built

The destination database must be a database in the current cloud. If you want to migrate data between databases, select To the cloud.

* Source DB Engine: [✓ MySQL] Oracle | MySQL schema and logic table | MongoDB | Redis

* Destination DB Engine: [✓ MySQL] DDM | GaussDB(for MySQL)

* Network Type: Public network

[✓] I understand that an EIP will be automatically bound to the replication instance and released after the replication task is complete.

* Destination DB Instance: Select an instance — View DB Instance | View Unselectable DB Instance

Replication Instance Subnet: Select the subnet — View Subnets | View occupied IP address

* Migration Type: [✓ Full+Incremental] Full

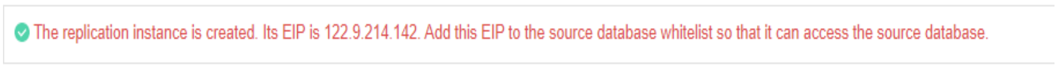
This migration type allows you to migrate data with minimal downtime. After a full migration initializes the destination database, an incremental migration parses logs to ensure data consistency between the source and destination databases.

* Destination DB Instance Access: [✓ Read-only] Read/Write

During the migration, the destination DB instance becomes read-only to ensure the integrity and success of data migration. When the task is complete, the DB instance becomes readable and writable. This process takes a few minutes. This option is recommended.

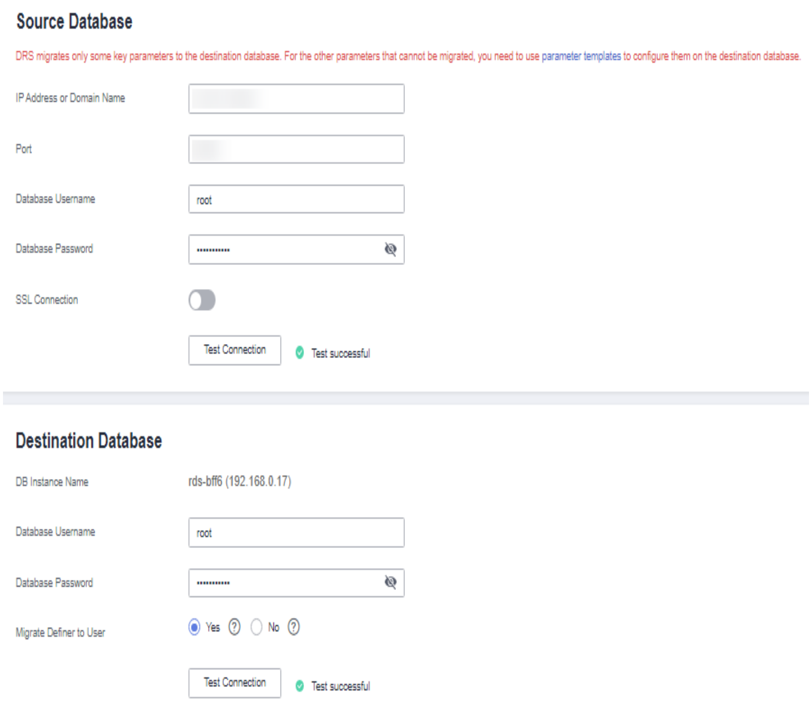
Step 3 Click **Create Now**.

It takes about 5 to 10 minutes to create a replication instance. After the replication instance is created, you can obtain its EIP.



The replication instance is created. Its EIP is 122.9.214.142. Add this EIP to the source database whitelist so that it can access the source database.

Step 4 Configure the source and destination database information.



Source Database

DRS migrates only some key parameters to the destination database. For the other parameters that cannot be migrated, you need to use parameter templates to configure them on the destination database.

IP Address or Domain Name	[redacted]
Port	[redacted]
Database Username	root
Database Password	••••••••••
SSL Connection	Off
	Test Connection ✓ Test successful

Destination Database

DB Instance Name	rds-bff6 (192.168.0.17)
Database Username	root
Database Password	••••••••••
Migrate Definer to User	[✓ Yes] No
	Test Connection ✓ Test successful

Step 5 Click **Next**.

Step 6 On the **Set Task** page, configure parameters as required.

- Set **Flow Control** to **No**.
- Set **Migration Object** to **All**.

Step 7 Click **Next**. On the **Check Task** page, check the migration task.

- If any check fails, review the cause and rectify the fault. After the fault is rectified, click **Check Again**.
- If all check items are successful, click **Next**.

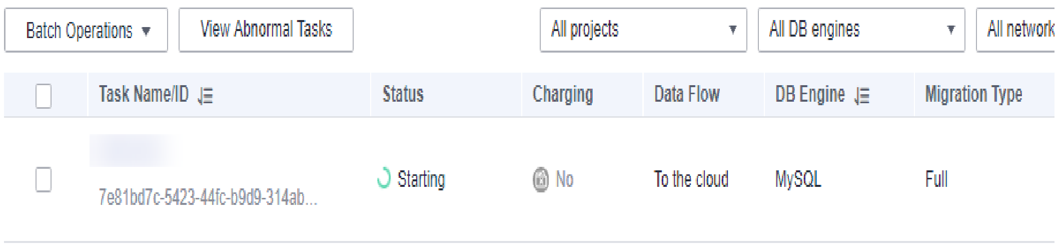
Step 8 Compare source and destination database parameters.

- Click **Next** to skip the comparison if you do not want to compare the parameters.
- Compare common parameters: If the parameter values in the list are inconsistent, click **Save Change** to change the destination database values to match those of the source database.

Step 9 Click **Submit** to submit the task.

Return to the **Online Migration Management** page and check the migration task status.

It takes several minutes to complete.



Task Name/ID	Status	Charging	Data Flow	DB Engine	Migration Type
[redacted] 7e81bd7c-5423-44fc-b9d9-314ab...	Starting	No	To the cloud	MySQL	Full

If the status changes to **Completed**, the migration task is complete.

----End

2.4.7.2 Checking Migration Results

You can use either of the following methods to check the migration results:

1. Use DRS to compare migration objects, users, and data of source and destination databases and obtain the migration results. For details, see **Checking the Migration Results on the DRS Console**.
2. Log in to the destination instance to check whether the databases, tables, and data are migrated. For details, see **Checking the Migration Results on the RDS Console**.

Checking the Migration Results on the DRS Console

Step 1 Log in to the **management console**.

Step 2 Click 📍 in the upper left corner and select your region.

Step 3 Under the service list, choose **Databases** > **Data Replication Service**.

Step 4 Click the DRS instance name.

Step 5 Click **Migration Comparison** in the navigation pane. Under the **Object-Level Comparison** tab, click **Compare** to check whether all objects have been migrated to the destination instance.

Step 6 Click the **Data-Level Comparison** tab. On the displayed page, click **Create Comparison Task** to check whether the databases and tables of the source and destination instances are the same.

Step 7 Click **Account-Level Comparison** and check whether the accounts and permissions of the source and destination instances are the same.

----End

Checking the Migration Results on the RDS Console

Step 1 Log in to the **management console**.

Step 2 Click 📍 in the upper left corner and select your region.

Step 3 Click the service list icon on the left and choose **Databases** > **Relational Database Service**.

Step 4 Locate the destination instance and click **Log In** in the **Operation** column.

Step 5 In the displayed dialog box, enter the password and click **Test Connection**.

Step 6 After the connection test is successful, click **Log In**.

Step 7 Check whether the databases and tables of the source instance have been migrated.

----End

Performing a Performance Test

After the migration is complete, you can perform a performance test as required.

2.5 Using RDS for MySQL to Set Up WordPress

WordPress is a blog platform developed based on PHP. It is usually used with RDS for MySQL database servers to help users build websites. This section describes how to set up WordPress in the Linux, Apache, MySQL and PHP (LAMP) environment using Huawei Cloud VPC, ECS, and RDS for MySQL.

1. **Configuring Network Information**
2. **Buying an ECS**
3. **Setting Up the LAMP Environment**
4. **Buying and Configuring an RDS DB Instance**
5. **Installing WordPress**

Preparations

During the setup, you will use the following services or tools:

- Cloud services: Huawei Cloud ECS and RDS for MySQL.
- MySQL client: a database configuration tool
- PuTTY: a remote login tool

NOTE

The previous software is provided by third-party websites. The information is just for your reference and not for commercial use.

Configuring Network Information

Step 1 Log in to the **management console**.

Step 2 Click in the upper left corner and select a region and a project.

Step 3 Choose **Networking** > **Virtual Private Cloud**.

Step 4 On the displayed page, click **Create VPC** to create a VPC, such as vpc-01.

Step 5 On the displayed page, enter a VPC name, set **IPv4 CIDR Block** to **192.168**, select an AZ as required, and add a subnet. Retain the default settings for other parameters. Then, click **Create Now**. After the VPC is created, return to the network console.

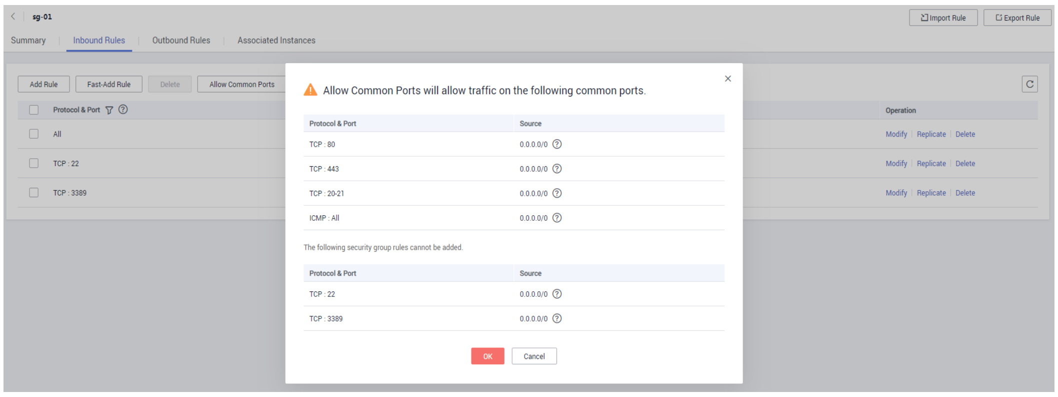
Step 6 On the network console, choose **Access Control** > **Security Groups** and click **Create Security Group**. The following uses sg-01 as an example.

Step 7 On the **Security Groups** page, locate the target security group and click **Manage Rule** in the **Operation** column.

Step 8 On the **Inbound Rules** page, click **Allow Common Ports** to enable common ports and network protocols.

Allow Common Ports: All inbound ICMP traffic and inbound traffic on ports 22, 80, 443, and 3389 are allowed by default. This option is suitable for cloud servers used in remote login, public network connection, and website services.

Figure 2-35 Adding a security group rule



Allow Common Ports will allow traffic on the following common ports.

Protocol & Port	Source
TCP : 80	0.0.0.0/0
TCP : 443	0.0.0.0/0
TCP : 20-21	0.0.0.0/0
ICMP : All	0.0.0.0/0

The following security group rules cannot be added.

Protocol & Port	Source
TCP : 22	0.0.0.0/0
TCP : 3389	0.0.0.0/0

----End

Buying an ECS

Step 1 Log in to the **management console**.

Step 2 Click in the upper left corner and select a region and a project.

Step 3 Choose **Compute** > **Elastic Cloud Server**. The **Elastic Cloud Server** page is displayed.

Step 4 On the ECS console, buy an ECS.

1. Configure basic settings: Select the pay-per-use billing mode, a region, and an image. Retain the default settings for other parameters.

   The public image **CentOS7.4 64bit for P2v(40GB)** is used as an example, as shown in **Figure 2-36**.

   Figure 2-36 Selecting an image

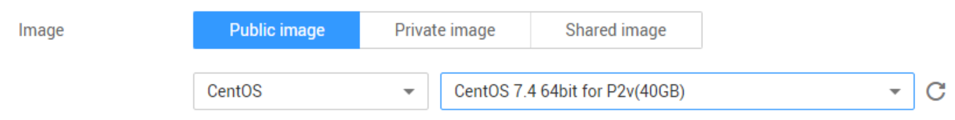
   

   Image: Public image | Private image | Shared image
   CentOS — CentOS 7.4 64bit for P2v(40GB)

2. Configure network: Select a VPC and security group, and purchase an EIP. Retain the default settings for other parameters.
   a. Select the created VPC vpc-01.
   b. Select the created security group sg-01.
   c. Select **Auto assign** for **EIP**.
3. Configure advanced settings: Enter an ECS name and password, and click **Next: Confirm**.
   a. Enter an ECS name, such as ***ecs-01***.
   b. Enter a password.

4. Confirm: Confirm the information and click **Next**.

Step 5 After the ECS is created, view and manage it on the ECS console.

----End

Setting Up the LAMP Environment

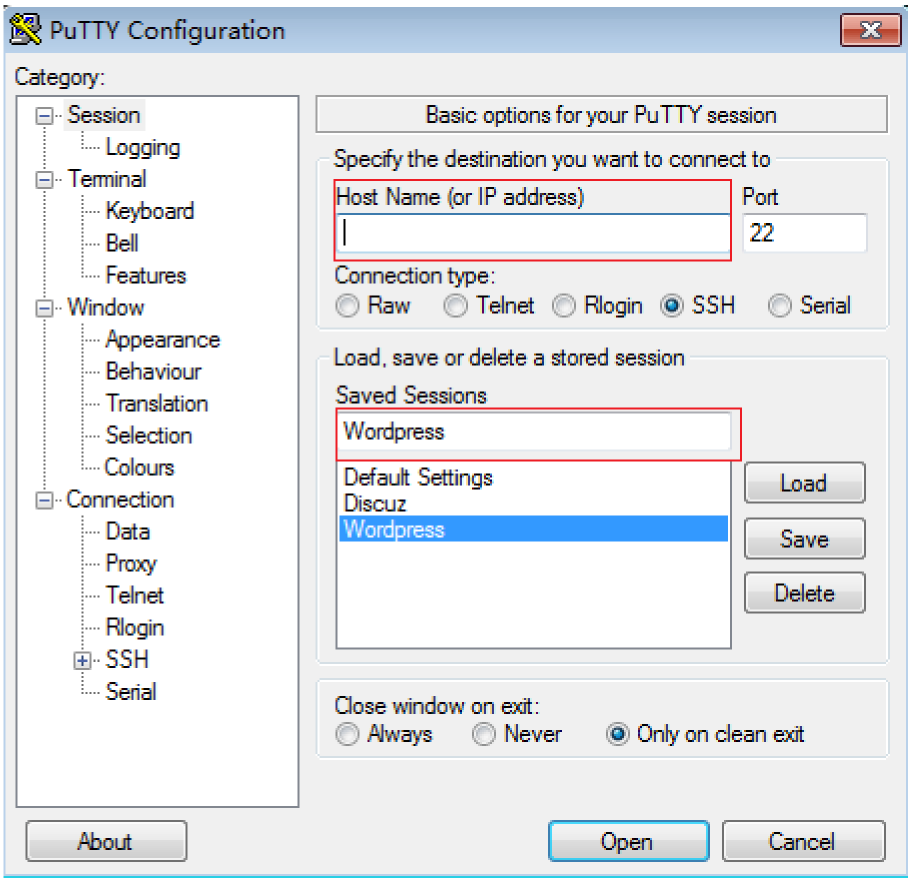
Step 1 Download the PuTTY client.

Step 2 Decompress the package, locate **putty** from the extracted files and double-click it.

Step 3 In the displayed PuTTY configuration dialog box, choose **Session** and specify basic options for your PuTTY session in the right pane. Then, click **Open** as shown in **Figure 2-37**.

1. Enter the EIP of your ECS in the **Host Name (or IP address)** text box.
2. Enter a session name in the **Saved Sessions** text box and click **Save**. **Wordpress** is used as an example. Retain the default settings for other parameters.

Figure 2-37 Configuring PuTTY



Step 4 In the displayed login window, enter the ECS username and password to log in to ECS.

Step 5 Obtain the **root** permissions so that you can enter commands in PuTTY.

Enter commands to install MySQL, PHP or other software. For example, run the following command to install PHP:

yum install -y httpd php php-fpm php-server php-mysql mysql

The installation is complete if the following command output is displayed:

```
Complete
```

Step 6 Run the following command to install a decompression software:

yum install -y unzip

Step 7 Run the following command to download and decompress the WordPress installation package:

wget -c https://wordpress.org/wordpress-4.9.1.tar.gz

tar xzf wordpress-4.9.1.tar.gz -C /var/www/html

chmod -R 777 /var/www/html

Step 8 After the installation is complete, run the following commands to start related services in sequence:

systemctl start httpd.service

systemctl start php-fpm.service

Step 9 Enable automatic start of the service during system startup.

systemctl enable httpd.service

----End

Buying and Configuring an RDS DB Instance

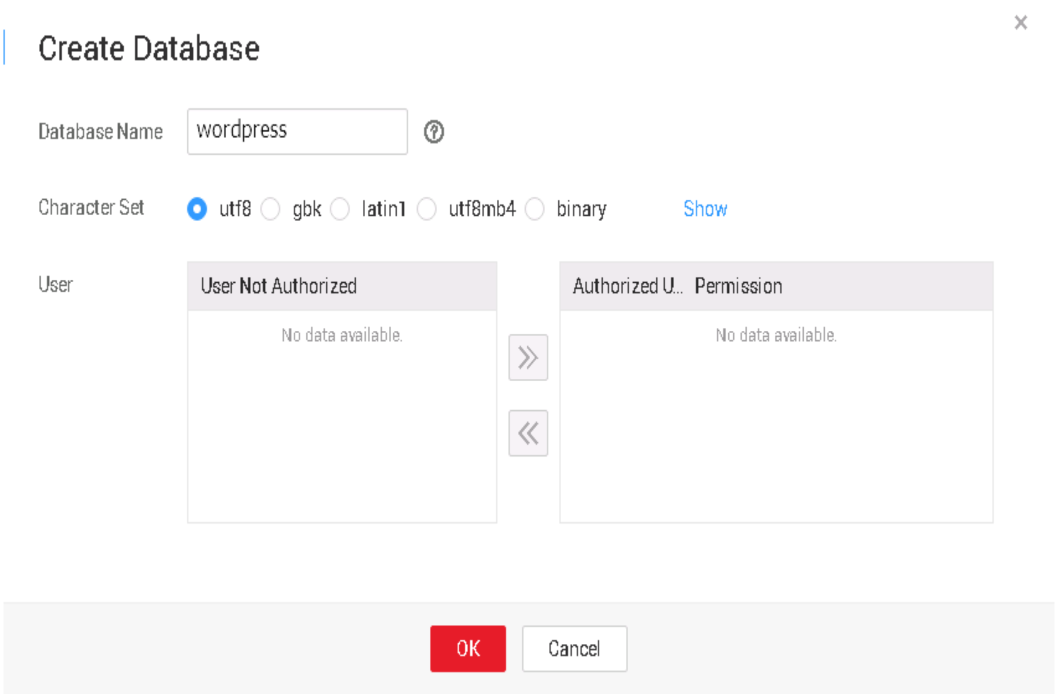
Step 1 **Buy a DB instance** as required.

- DB instance rds-01 is used as an example. Select MySQL 5.6 or 5.7.
- Ensure that the RDS DB instance uses the same security group as the ECS so that you can access the RDS DB instance through the ECS.
- Set the root user password and keep the password secure. The system cannot retrieve your password.

Step 2 Go to the RDS console. On the **Instances** page, click the target DB instance rds-01. The **Basic Information** page is displayed.

Step 3 Choose **Databases** in the navigation pane on the left and click **Create Database**. In the displayed dialog box, enter a database name, such as ***wordpress***, select a character set, and authorize permissions for database users. Then, click **OK**.

Figure 2-38 Creating a database


Create Database

Database Name: wordpress

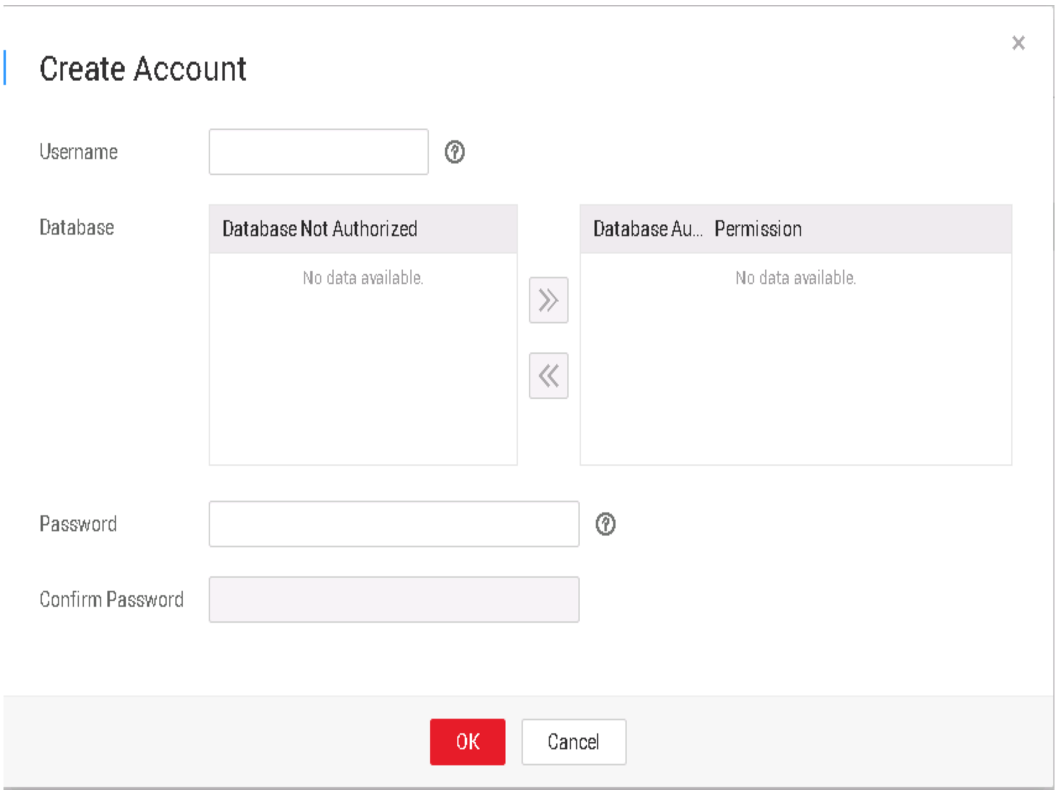
Character Set: [✓ utf8] gbk latin1 utf8mb4 binary Show

User

User Not Authorized	Authorized U... Permission
No data available.	No data available.

Step 4 Choose **Accounts** in the navigation pane on the left and click **Create Account**. In the displayed dialog box, enter the database username, such as ***tony***, authorize permissions for database ***wordpress*** created in **Step 3**, enter the password and confirm the password. Then, click **OK**.

Figure 2-39 Creating an account



----End

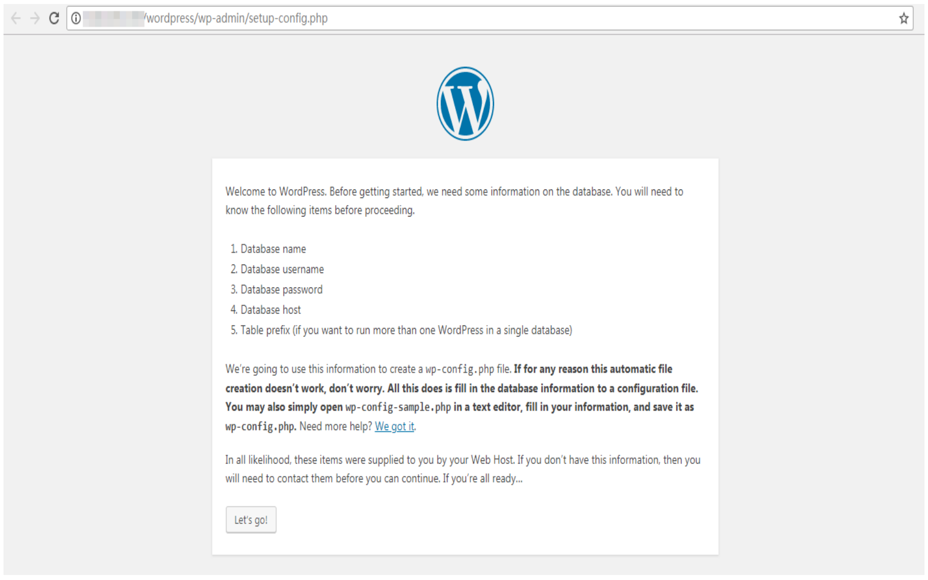
Installing WordPress

Step 1 On the **Elastic Cloud Server** page, locate the target ECS and click **Remote Login** in the **Operation** column.

Step 2 In the Internet Explorer, enter **http://***EIP***/wordpress** in the address box and click **Let's go!**

In the preceding URL, ***EIP*** indicates the EIP automatically assigned when you purchase the ECS in **Buying an ECS**.

Figure 2-40 Visiting WordPress



[redacted]/wordpress/wp-admin/setup-config.php

Welcome to WordPress. Before getting started, we need some information on the database. You will need to know the following items before proceeding.

1. Database name
2. Database username
3. Database password
4. Database host
5. Table prefix (if you want to run more than one WordPress in a single database)

We're going to use this information to create a `wp-config.php` file. **If for any reason this automatic file creation doesn't work, don't worry. All this does is fill in the database information to a configuration file. You may also simply open `wp-config-sample.php` in a text editor, fill in your information, and save it as `wp-config.php`.** Need more help? We got it.

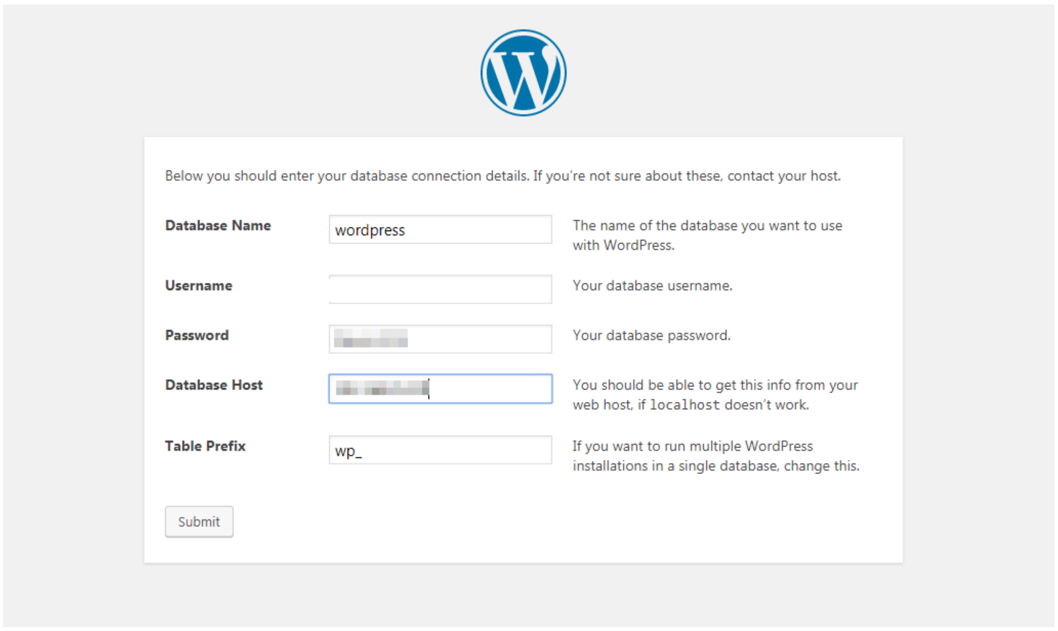
In all likelihood, these items were supplied to you by your Web Host. If you don't have this information, then you will need to contact them before you can continue. If you're all ready…

Let's go!

Step 3 Enter database connection information and click **Submit**.

- The database name is ***wordpress***.
- The username is ***tony***.
- The password is the one that you set for ***tony***.
- The database host is the floating IP address of DB instance rds-01.

Figure 2-41 Entering database connection information



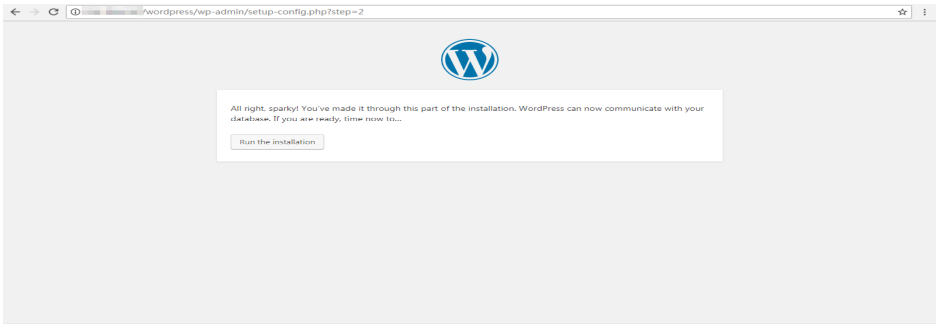
Below you should enter your database connection details. If you're not sure about these, contact your host.

Database Name	wordpress	The name of the database you want to use with WordPress.
Username		Your database username.
Password	[redacted]	Your database password.
Database Host	[redacted]	You should be able to get this info from your web host, if `localhost` doesn't work.
Table Prefix	wp_	If you want to run multiple WordPress installations in a single database, change this.

Submit

Step 4 After the database connection details are verified, click **Run the installation**.

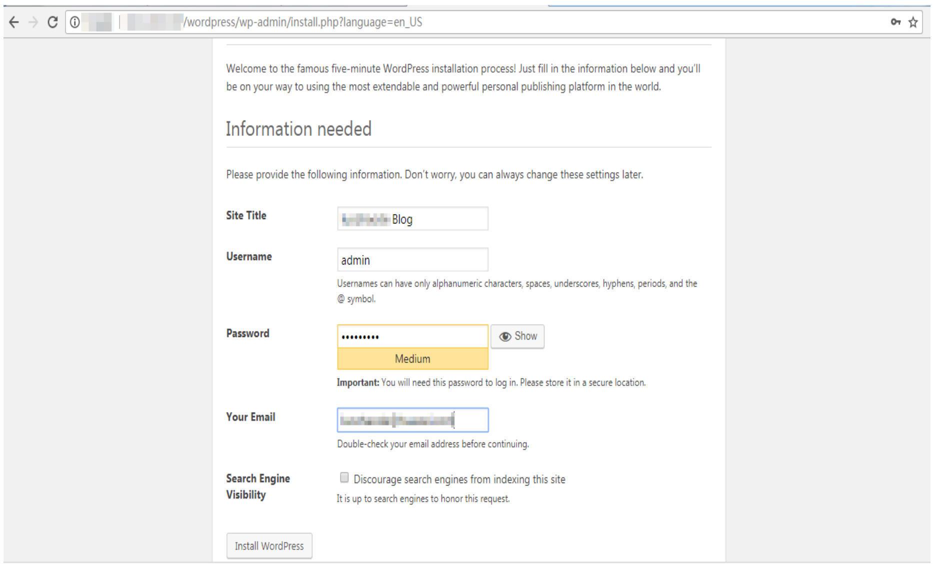
Figure 2-42 Running the installation


[redacted]/wordpress/wp-admin/setup-config.php?step=2

All right, sparky! You've made it through this part of the installation. WordPress can now communicate with your database. If you are ready, time now to…

Step 5 Set **Site Title**, **Username**, and **Password** for logging in to your blog. Then, click **Install WordPress**.

Figure 2-43 Setting basic information


[redacted]/wordpress/wp-admin/install.php?language=en_US

Welcome to the famous five-minute WordPress installation process! Just fill in the information below and you'll be on your way to using the most extendable and powerful personal publishing platform in the world.

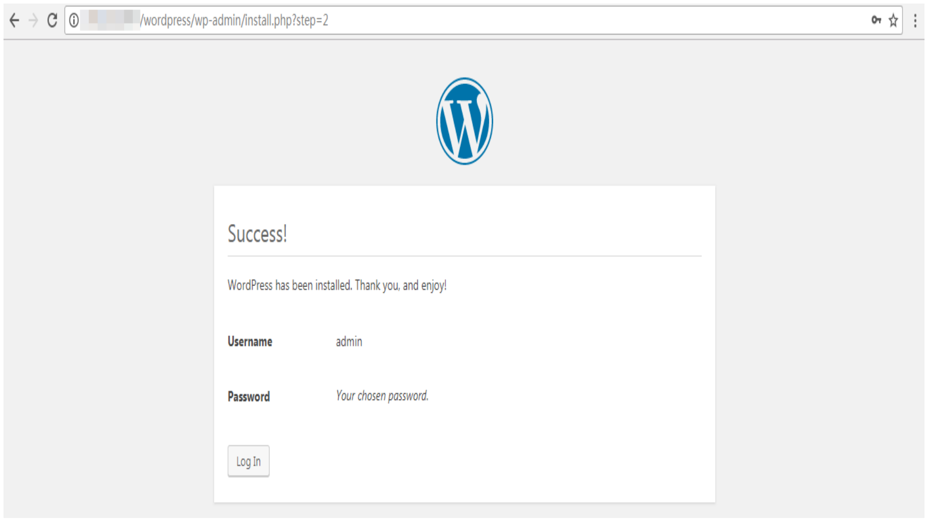
Information needed

Please provide the following information. Don't worry, you can always change these settings later.

Site Title	[redacted] Blog
Username	admin Usernames can have only alphanumeric characters, spaces, underscores, hyphens, periods, and the @ symbol.
Password	••••••••• Show Medium **Important:** You will need this password to log in. Please store it in a secure location.
Your Email	[redacted] Double-check your email address before continuing.
Search Engine Visibility	☐ Discourage search engines from indexing this site It is up to search engines to honor this request.

Step 6 Click **Log In** after WordPress has been successfully installed.

Figure 2-44 Successful installation


[redacted]/wordpress/wp-admin/install.php?step=2

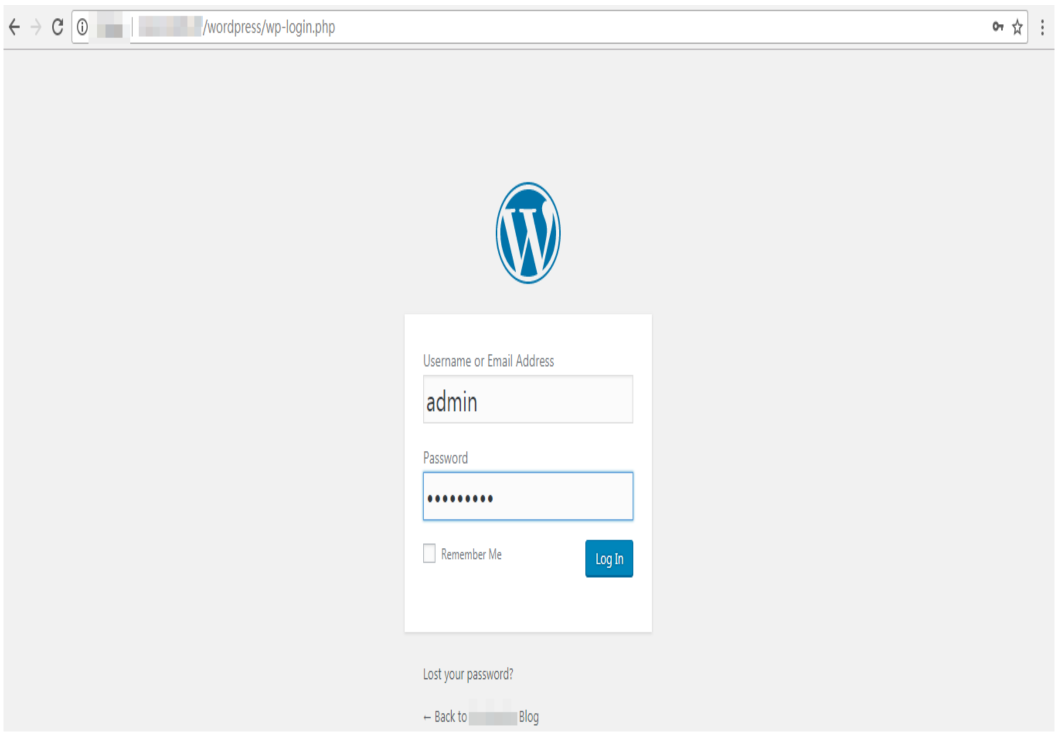
Success!

WordPress has been installed. Thank you, and enjoy!

Username	admin
Password	*Your chosen password.*

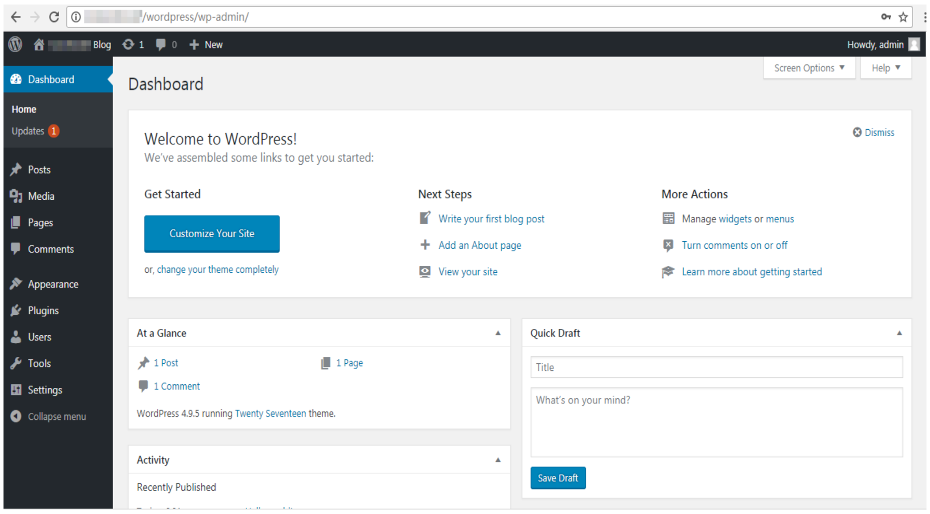
Step 7 Enter the username and password on the displayed login page. Then, click **Log In**.

Figure 2-45 Logging in



Step 8 Check that WordPress has been deployed successfully.

Figure 2-46 Verification



----End

2.6 Using RDS for MySQL to Set Up Discuz!

Crossday Discuz! Board (Discuz! for short) is a universal community forum software system. You can set up a customized forum with comprehensive functions and strong load capability on the Internet through simple installation and settings. This section describes how to set up Discuz! in the LAMP environment using Huawei Cloud VPC, ECS, and RDS for MySQL.

1. **Configuring Network Information**
2. **Creating an ECS**
3. **Setting Up the LAMP Environment**
4. **Buying and Configuring an RDS DB Instance**
5. **Installing Discuz!**

Preparations

During the setup, you will use the following services or tools:

- Cloud services: ECS and RDS on Huawei Cloud
- PuTTY: a remote login tool
- Installation packages
  - Apache Http Server 2.4.6
  - MySQL 5.4.16
  - PHP 5.4.16

NOTE

The previous software is provided by third-party websites. The information is just for your reference and not for commercial use.

Configuring Network Information

Step 1 Log in to the **management console**.

Step 2 Click in the upper left corner and select a region and a project.

Step 3 Choose **Networking** > **Virtual Private Cloud**.

Step 4 On the displayed page, click **Create VPC** to create a VPC, such as vpc-01.

Step 5 On the displayed page, enter a VPC name, set **IPv4 CIDR Block** to **192.168**, select an AZ as required, and add a subnet. Retain the default settings for other parameters. Then, click **Create Now**. After the VPC is created, return to the network console.

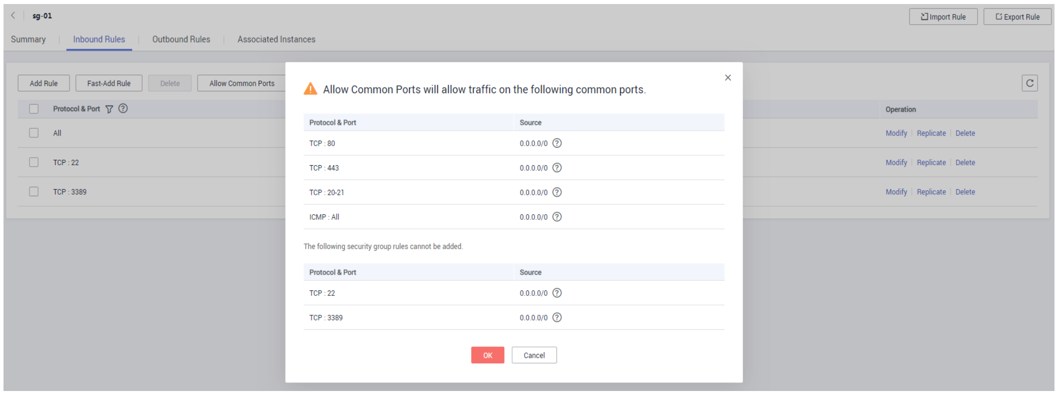
Step 6 On the network console, choose **Access Control** > **Security Groups** and click **Create Security Group**. The following uses sg-01 as an example.

Step 7 On the **Security Groups** page, locate the target security group and click **Manage Rule** in the **Operation** column.

Step 8 On the **Inbound Rules** page, click **Allow Common Ports** to enable common ports and network protocols.

Allow Common Ports: All inbound ICMP traffic and inbound traffic on ports 22, 80, 443, and 3389 are allowed by default. This option is suitable for cloud servers used in remote login, public network connection, and website services.

Figure 2-47 Adding a security group rule



Protocol & Port	Operation
All	Modify | Replicate | Delete
TCP : 22	Modify | Replicate | Delete
TCP : 3389	Modify | Replicate | Delete

Allow Common Ports will allow traffic on the following common ports.

Protocol & Port	Source
TCP : 80	0.0.0.0/0
TCP : 443	0.0.0.0/0
TCP : 20-21	0.0.0.0/0
ICMP : All	0.0.0.0/0

The following security group rules cannot be added.

Protocol & Port	Source
TCP : 22	0.0.0.0/0
TCP : 3389	0.0.0.0/0

----End

Buying an ECS

Step 1 Log in to the **management console**.

Step 2 Click in the upper left corner and select a region and a project.

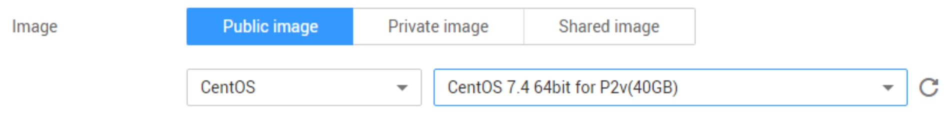
Step 3 Choose **Compute** > **Elastic Cloud Server**. The **Elastic Cloud Server** page is displayed.

Step 4 On the ECS console, buy an ECS.

1. Configure basic settings: Select the pay-per-use billing mode, a region, and an image. Retain the default settings for other parameters.

   The public image **CentOS7.4 64bit for P2v(40GB)** is used as an example, as shown in **Figure 2-48**.

   Figure 2-48 Selecting an image

2. Configure network: Select a VPC and security group, and purchase an EIP. Retain the default settings for other parameters.
   a. Select the created VPC vpc-01.
   b. Select the created security group sg-01.
   c. Select **Auto assign** for **EIP**.
3. Configure advanced settings: Enter an ECS name and password, and click **Next: Confirm**.
   a. Enter an ECS name, such as ***ecs-01***.
   b. Enter a password.
4. Confirm: Confirm the information and click **Next**.

Step 5 After the ECS is created, view and manage it on the ECS console.

----End

Setting Up the LAMP Environment

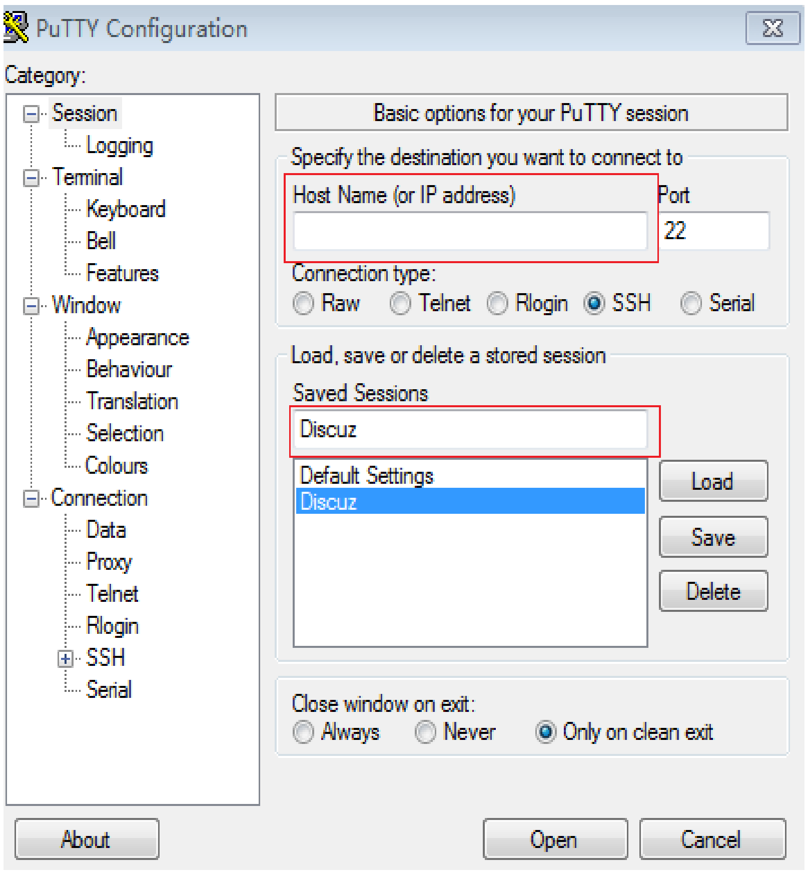
Step 1 Download the PuTTY client.

Step 2 Decompress the package, locate **putty** from the extracted files and double-click it.

Step 3 In the displayed PuTTY configuration dialog box, choose **Session** and specify basic options for your PuTTY session in the right pane. Then, click **Open** as shown in Figure 2-49.

1. Enter the EIP of your ECS in the **Host Name (or IP address)** text box.
2. Enter a session name in the **Saved Sessions** text box and click **Save**. **Discuz** is used as an example. Retain the default settings for other parameters.

Figure 2-49 Configuring PuTTY


PuTTY Configuration

Category:

Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address):

Port: 22

Connection type: Raw Telnet Rlogin [✓ SSH] Serial

Load, save or delete a stored session

Saved Sessions: Discuz

Default Settings
Discuz

Close window on exit: Always Never [✓ Only on clean exit]

Step 4 In the displayed login window, enter the ECS username and password to log in to ECS.

Step 5 Install Apache, MySQL, PHP and other software.

Obtain the **root** permissions so that you can enter commands in PuTTY.

Enter commands to install software. For example, run the following command to install PHP:

yum install -y httpd php php-fpm php-server php-mysql mysql

The installation is complete if the following command output is displayed:

```
Complete
```

Step 6 After the installation is complete, start related services in sequence.

systemctl start httpd.service

systemctl start php-fpm.service

----End

Buying and Configuring an RDS DB Instance

Step 1 **Buy a DB instance** as required.

- DB instance rds-01 is used as an example. Select MySQL 5.6 or 5.7.
- Ensure that the RDS DB instance uses the same security group as the ECS so that you can access the RDS DB instance through the ECS.
- Set the root user password and keep the password secure. The system cannot retrieve your password.

Step 2 After the RDS DB instance is created, view or manage it on the **management console**.

----End

Installing Discuz!

Step 1 Download the **Discuz! installation package**.

Step 2 Upload the installation package to the ECS using a data transfer tool.

1. Run the following command to decompress the Discuz! installation package:

   unzip Discuz_X3.3_SC_UTF8.zip
2. Run the following command to copy all files in **upload** to **/var/www/html/**.

   cp -R upload/* /var/www/html/
3. Run the following command to grant write permissions to other users.

   chmod -R 777 /var/www/html

Step 3 Enter **http://*EIP*/install** in the address box in a local Windows browser and install Discuz! following the guidance.

In the preceding URL, ***EIP*** indicates the EIP automatically assigned when you purchase the ECS in **Buying an ECS**. The **install** must be lowercase.

1. Confirm the agreement and click **I Agree**.
2. After the installation starts, check the installation environment and click **Next**.
3. Set the running environment and click **Next**.
4. Enter the database information and click **Next** to complete the installation.
   - The database address is the floating IP address of DB instance rds-01.
   - The database password is the root user password of DB instance rds-01.
   - Enter administrator information.

Step 4 After Discuz! is installed, enter **http://*EIP*/forum.php** in the browser address bar. If the forum homepage is displayed, the website is successfully built.

----End

2.7 Description of innodb_flush_log_at_trx_commit and sync_binlog

The **innodb_flush_log_at_trx_commit** and **sync_binlog** are key parameters for controlling the disk write policy and data security of RDS for MySQL. Different parameter values have different impacts on performance and security.

Table 2-5 Parameter description

Parameter	Allowed Values	Description
innodb_flush_log_at_trx_commit	0, 1, and 2	Controls the balance between strict ACID compliance for commit operations, and higher performance that is possible when commit-related I/O operations are rearranged and done in batches. The default value is **1**. For details, see **Parameter Description**.
sync_binlog	0 to 4, 294, 967, 295	Sync binlog (RDS for MySQL flushes binary logs to disks or relies on the OS).

Parameter Description

- **innodb_flush_log_at_trx_commit**:
  - **0**: The log buffer is written out to the log file once per second and the flush to disk operation is performed on the log file, but nothing is done at a transaction commit.
  - **1**: The log buffer is written out to the log file at each transaction commit and the flush to disk operation is performed on the log file.
  - **2**: The log buffer is written out to the file at each commit, but the flush to disk operation is not performed on it. However, the flushing on the log file takes place once per second.

  NOTE

  - A value of **0** is the fastest choice but less secure. Any mysqld process crash can erase the last second of transactions.
  - A value of **1** is the safest choice because in the event of a crash you lose at most one statement or transaction from the binary log. However, it is also the slowest choice.
  - A value of **2** is faster and more secure than **0**. Only an operating system crash or a power outage can erase the last second of transactions.

- **sync_binlog=1 or N**

  By default, the binary log is not every time synchronized to disk. In the event of a crash, the last statement in the binary log may get lost.

  To prevent this issue, you can use the **sync_binlog** global variable (**1** is the safest value, but also the slowest) to synchronize the binary log to disk after N binary log commit groups.

Recommended Configurations

Table 2-6 Recommended configurations

innodb_flush_log_at_trx_commit	sync_binlog	Description
1	1	High data security and strong disk write capability
1	0	High data security and insufficient disk write capability. Standby lagging behind or no replication is allowed.
2	0/N (0 < N < 100)	Low data security. A small amount of transaction log loss and replication delay is allowed.
0	0	Limited disk write capability. No replication or long replication delay is allowed.

NOTE

- When both **innodb_flush_log_at_trx_commit** and **sync_binlog** are set to **1**, the security is the highest but the write performance is the lowest. In the event of a crash you lose at most one statement or transaction from the binary log. This is also the slowest choice due to the increased number of disk writes.
- When **sync_binlog** is set to *N* (*N*>1) and **innodb_flush_log_at_trx_commit** is set to **2**, the RDS for MySQL write operation achieves the optimal performance.

2.8 Resolving Database Operation Failures Caused by Metadata Locks on RDS for MySQL

RDS for MySQL uses metadata locking to manage concurrent access to database objects and to ensure data consistency. Metadata locks have been introduced since MySQL 5.5. A metadata lock on a table prevents any data from being read or written, resulting in SQL statements being blocked. You can use Data Admin Service (DAS) to resolve this issue.

Procedure

Step 1 Log in to the management console.

Step 2 Click in the upper left corner and select a region and a project.

Step 3 Click in the upper left corner of the page and choose **Databases** > **Relational Database Service**.

Step 4 On the **Instances** page, locate the DB instance and click **Log In** in the **Operation** column.

Alternatively, click the instance name on the **Instances** page. On the displayed **Basic Information** page, click **Log In** in the upper right corner.

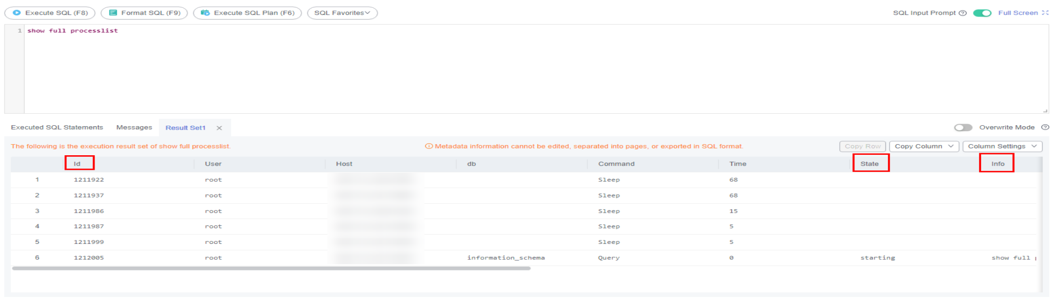
Step 5 On the displayed login page, enter the username and password and click **Log In**.

Step 6 On the top menu bar, choose **SQL Operations** > **SQL Query**.

Step 7 Run the following SQL statement in the SQL window to view the states of all database threads:

show full processlist

Figure 2-50 Execution result



```
show full processlist
```

The following is the execution result set of show full processlist.

Metadata information cannot be edited, separated into pages, or exported in SQL format.

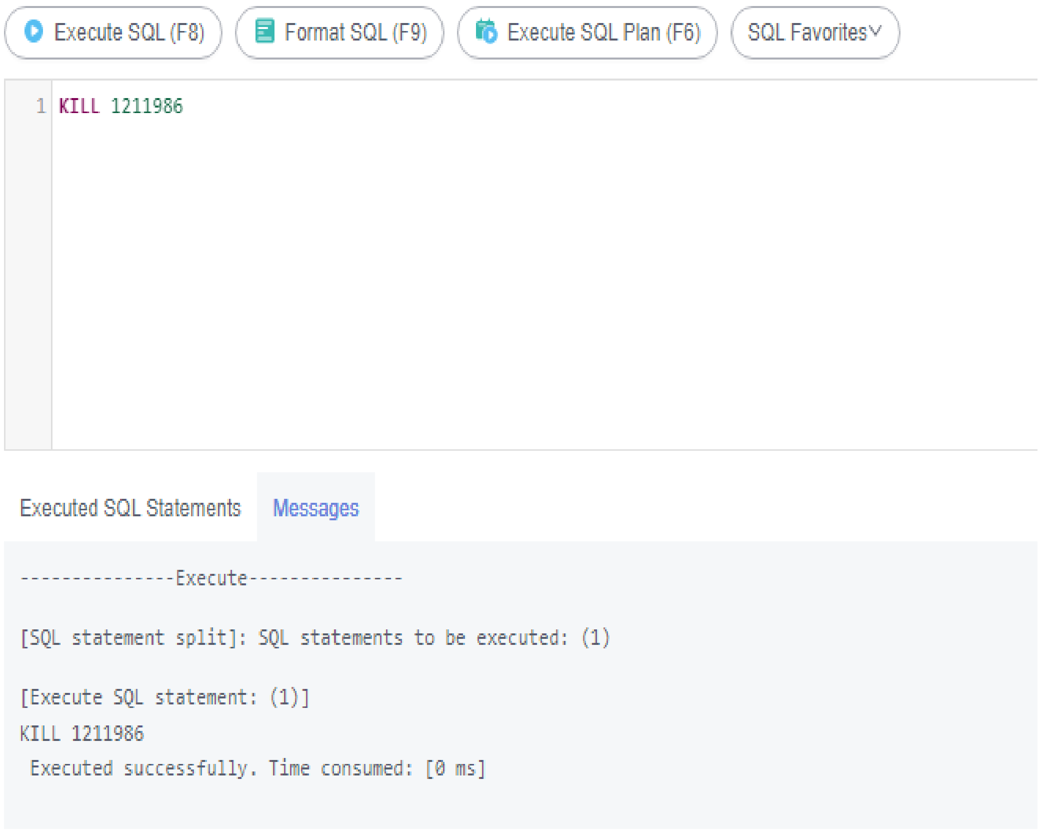
	Id	User	Host	db	Command	Time	State	Info
1	1211922	root	[redacted]		Sleep	68		
2	1211937	root	[redacted]		Sleep	68		
3	1211986	root	[redacted]		Sleep	15		
4	1211987	root	[redacted]		Sleep	5		
5	1211999	root	[redacted]		Sleep	5		
6	1212005	root	[redacted]	information_schema	Query	0	starting	show full

Step 8 Check whether a large number of **Waiting for table metadata lock** are displayed in the **State** column, which would indicate that SQL statements are being blocked. Locate the sessions in the table operations in the **Info** column and record the values in the **Id** column.

Step 9 Run the following command in the SQL window to unlock the metadata lock:

kill *Id*

Figure 2-51 Execution result



```
KILL 1211986
```

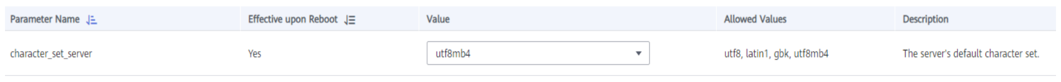
```
---------------Execute---------------

[SQL statement split]: SQL statements to be executed: (1)

[Execute SQL statement: (1)]
KILL 1211986
 Executed successfully. Time consumed: [0 ms]
```

----End

2.9 How Do I Use the utf8mb4 Character Set to Store Emojis in an RDS for MySQL DB Instance?

To store emojis in an RDS for MySQL DB instance, ensure that:

- The client outputs the utf8mb4 character set.
- The connection supports the utf8mb4 character set. If you want to use a JDBC connection, download MySQL Connector/J 5.1.13 or a later version and leave **characterEncoding** undefined for the JDBC connection string.
- Configure the RDS DB instance as follows:
  - Setting **character_set_server** to **utf8mb4**



Parameter Name	Effective upon Reboot	Value	Allowed Values	Description
character_set_server	Yes	utf8mb4	utf8, latin1, gbk, utf8mb4	The server's default character set.

    i. **Log in to the management console**.
    ii. Click in the upper left corner and select a region and a project.
    iii. Click in the upper left corner of the page and choose **Databases** > **Relational Database Service**.

iv. On the **Instances** page, click the instance name.

v. In the navigation pane on the left, choose **Parameters**. On the **Parameters** tab page, locate **character_set_server** and change its value to **utf8mb4**.

vi. Click **Save**. In the displayed dialog box, click **Yes**.

– Selecting **utf8mb4** for database character set

i. Log in to the management console.

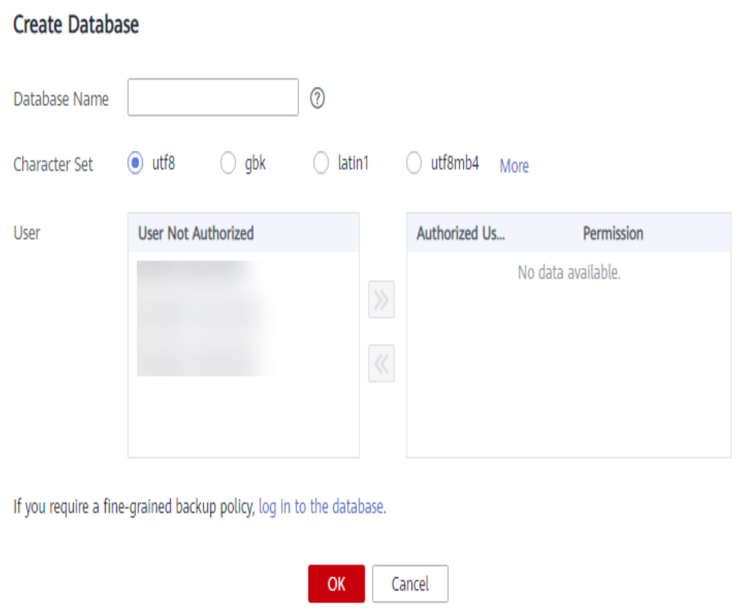
ii. Click in the upper left corner and select a region and a project.

iii. Click in the upper left corner of the page and choose **Databases** > **Relational Database Service**.

iv. On the **Instances** page, click the instance name.

v. On the **Databases** page, click **Create Database**. In the displayed dialog box, enter a database name and remarks, select the character set **utf8mb4**, and authorize permissions for users. Then, click **OK**.

Figure 2-52 Creating a database



Create Database

Database Name:

Character Set: [✓ utf8] gbk latin1 utf8mb4 More

User

User Not Authorized

[redacted]

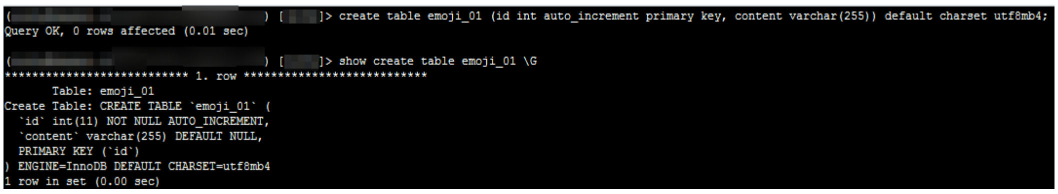
Authorized Us... Permission

No data available.

If you require a fine-grained backup policy, log in to the database.

OK Cancel

– Setting the character set of the table to **utf8mb4**



```
([redacted]) [[redacted]]> create table emoji_01 (id int auto_increment primary key, content varchar(255)) default charset utf8mb4;
Query OK, 0 rows affected (0.01 sec)

([redacted]) [[redacted]]> show create table emoji_01 \G
************************** 1. row **************************
       Table: emoji_01
Create Table: CREATE TABLE `emoji_01` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `content` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
1 row in set (0.00 sec)
```

FAQs

If you have set **characterEncoding** to **utf8** for the JDBC connection string, or the emoji data cannot be inserted properly after you have performed the above operations, you are advised to set the connection character set to **utf8mb4** as follows:

```
String query = "set names utf8mb4";
stat.execute(query);
```

2.10 Handling RDS for MySQL Long Transactions

Potential Impacts of Long Transactions

1. Long transactions lock resources and usually increase metadata locks and row locks. As a result, other transactions cannot access these resources, reducing the database concurrency.
2. Long transactions may occupy a large amount of memory.
3. Long transactions may cause too large log files and high storage usage.

Identifying Long Transactions

- Connect to your DB instance and check long transactions and their session IDs.

  After connecting to the DB instance, run the following command to view the ID of any transaction that has been executing for more than 3,000s, the executed SQL statement, and the corresponding session ID.

  mysql> SELECT trx_id, trx_state, trx_started, trx_mysql_thread_id, trx_query, trx_rows_modified FROM information_schema.innodb_trx WHERE TIME_TO_SEC(timediff(now(),trx_started)) >3000;

  Table 2-7 Parameter description

Parameter	Description
trx_id	Transaction ID.
trx_state	Transaction status, which can be **RUNNING**, **LOCK WAIT**, or **ROLLING BACK**.
trx_started	Time when the transaction was started.
trx_mysql_thread_id	ID of the MySQL session to which the transaction belongs.
trx_query	SQL statement executed by the transaction.
trx_rows_modified	Number of rows modified by the transaction.

- Check monitoring metrics for long transactions.
  a. **Log in to the management console**.
  b. Click ☰ in the upper left corner of the page and choose **Databases** > **Relational Database Service**.

c. On the **Instances** page, locate the target DB instance and click **View Metrics** in the **Operation** column.

d. Check the long transaction metric **rds_long_transaction**. If the metric increases linearly to a large value, there are long transactions.

Killing Long Transactions

1. Obtain the thread IDs corresponding to long transactions.

   Run the SQL statement in **Connect to your DB instance to check long transactions and their session IDs** to obtain the session ID of the transaction whose execution time exceeds a certain period (for example, 3,000s).

   mysql> SELECT trx_mysql_thread_id FROM information_schema.innodb_trx WHERE TIME_TO_SEC(timediff(now(),trx_started)) >3000;

2. After obtaining the session ID, run the **kill** command to kill the transaction.

   mysql> kill *trx_mysql_thread_id*

NOTICE

Killing a long transaction will cause the transaction to roll back. Evaluate the impact before running this command.

Configuring Long Transaction Alarms

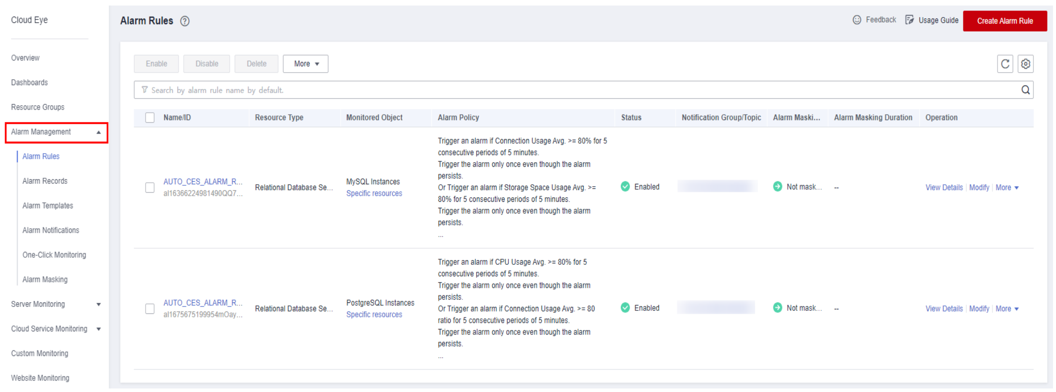
1. View the configured alarms.

   a. **Log in to the management console**.

   b. Click ☰ in the upper left corner of the page and choose **Management & Governance** > **Cloud Eye**.

   c. Choose **Alarm Management** > **Alarm Rules**.

Figure 2-53 Viewing alarm rules



Name/ID	Resource Type	Monitored Object	Alarm Policy	Status	Notification Group/Topic	Alarm Maski...	Alarm Masking Duration	Operation
AUTO_CES_ALARM_R... al16366224981490QQ7...	Relational Database Se...	MySQL Instances Specific resources	Trigger an alarm if Connection Usage Avg. >= 80% for 5 consecutive periods of 5 minutes. Trigger the alarm only once even though the alarm persists. Or Trigger an alarm if Storage Space Usage Avg. >= 80% for 5 consecutive periods of 5 minutes. Trigger the alarm only once even though the alarm persists. ...	Enabled	[redacted]	Not mask...	--	View Details | Modify | More
AUTO_CES_ALARM_R... al1675675199954mOay...	Relational Database Se...	PostgreSQL Instances Specific resources	Trigger an alarm if CPU Usage Avg. >= 80% for 5 consecutive periods of 5 minutes. Trigger the alarm only once even though the alarm persists. Or Trigger an alarm if Connection Usage Avg. >= 80 ratio for 5 consecutive periods of 5 minutes. Trigger the alarm only once even though the alarm persists. ...	Enabled	[redacted]	Not mask...	--	View Details | Modify | More

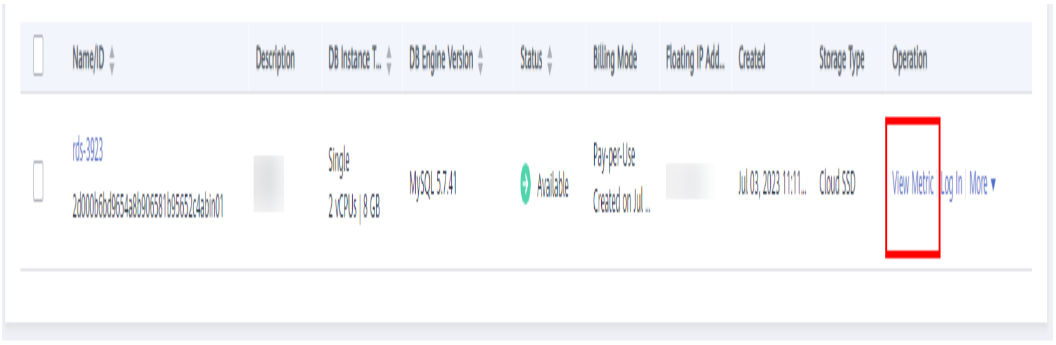
2. Configure long transaction alarms.

   a. Click ☰ in the upper left corner of the page and choose **Databases** > **Relational Database Service**.

b. On the **Instances** page, locate the target DB instance and click **View Metrics** in the **Operation** column.

c. View the **Long Transaction** metric.

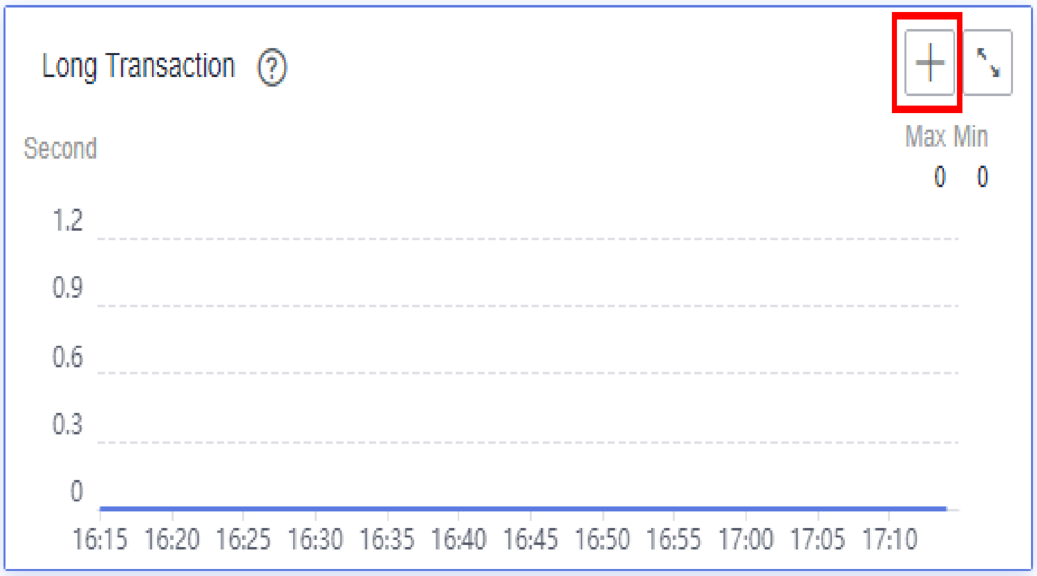
Figure 2-54 Viewing metrics



Name/ID	Description	DB Instance T...	DB Engine Version	Status	Billing Mode	Floating IP Add...	Created	Storage Type	Operation
rds-3923 2d000b6bd9654a8b906581b95652c4abin01		Single 2 vCPUs | 8 GB	MySQL 5.7.41	Available	Pay-per-Use Created on Jul ...		Jul 03, 2023 11:11...	Cloud SSD	View Metric | Log In | More ▾

d. Click + in the upper right corner of the **Long Transaction** metric.

Figure 2-55 Long Transaction



e. On the displayed page, set parameters as required. For details about the parameters, see **Creating an Alarm Rule**.

2.11 Performance Tuning

2.11.1 Identifying Why CPU Usage of RDS for MySQL DB Instances Is High and Providing Solutions

If the CPU usage of your RDS for MySQL instance is high or close to 100%, database performance deteriorates. For example, data read/write becomes slow, connecting to the instance takes a longer time, or errors are reported when you are trying to delete data.

NOTICE

The following functions of Data Admin Service (DAS) are only available to the accounts that are using them, from November 25, 2021, 00:00 GMT+08:00: SQL tuning, table structure comparison and synchronization, data tracking and rollback, data generator, and DBA intelligent O&M in Development Tool, as well as space, intelligent parameter recommendation, historical transaction, and binlog parsing functions in Intelligent O&M.

- For DB instances created after November 25, 2021, 00:00 GMT+08:00, **Solution 1** is recommended.
- For DB instances created before November 25, 2021, 00:00 GMT+08:00, **Solution 2** is recommended.

You are advised to **enable SQL audit** in advance so that you can view SQL execution records in audit logs to locate the fault when the CPU usage is high.

Solution 1

Analyze slow SQL logs and CPU usage to locate slow queries and then optimize them.

1. View the slow SQL logs to check for slowly executed SQL queries and view their performance characteristics (if any) to locate the cause.

   For details on viewing RDS for MySQL logs, see **Slow Query Log**.
2. View the CPU usage metric of your DB instance.

   For details, see **Performance**.
3. Create read replicas to reduce read pressure from primary DB instances.
4. Add indexes for associated fields in multi-table association queries.
5. Do not use the SELECT statement to scan all tables. You can specify fields or add the WHERE condition.

Solution 2

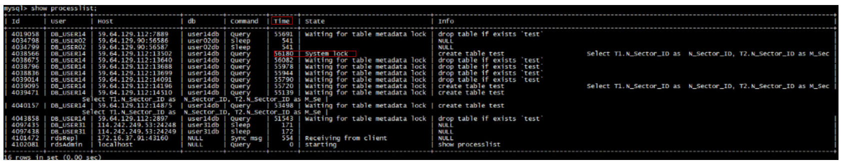
You can identify slow query statements and optimize them according to the suggestions provided by DAS to reduce the CPU usage.

Step 1 Connect to the RDS for MySQL DB instance.

You can connect to an instance through a private or public network. For details, see the *Relational Database Service Getting Started.*

Step 2 Run the following command to show the running threads and locate the queries that are slowly executed:

show full processlist



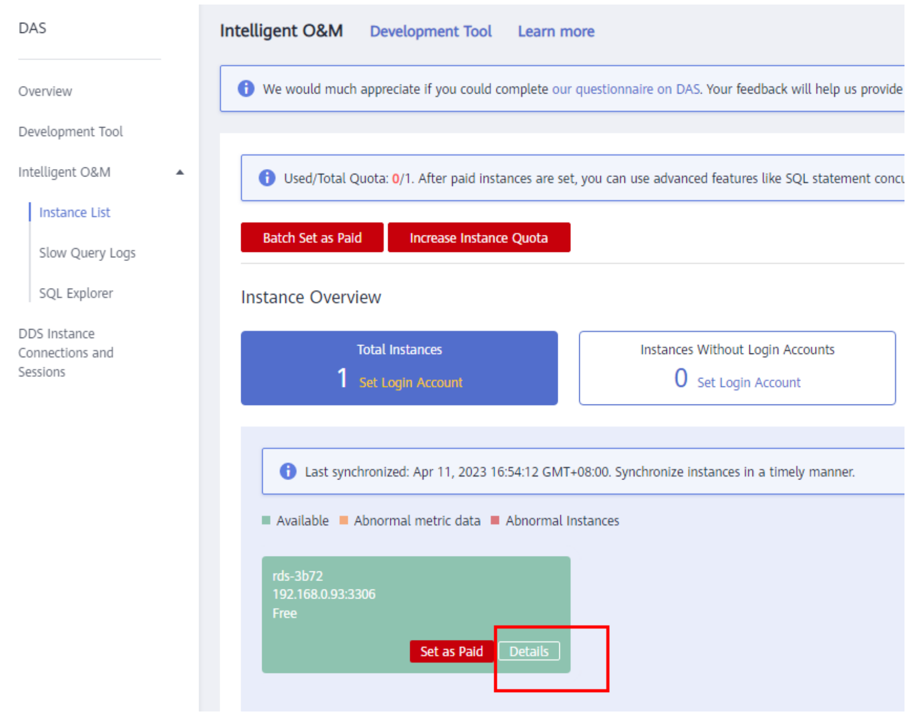
```
mysql> show processlist;
+---------+-----------+----------------------+-----------+----------+-------+---------------------------------+-----------------------------------------------------------------------------------------------------+
| Id      | User      | Host                 | db        | Command  | Time  | State                           | Info                                                                                                |
+---------+-----------+----------------------+-----------+----------+-------+---------------------------------+-----------------------------------------------------------------------------------------------------+
| 4019058 | DB_USER14 | 59.64.129.112:7889   | user14db  | Query    | 55691 | Waiting for table metadata lock | drop table if exists `test`                                                                         |
| 4034798 | DB_USER02 | 59.64.129.90:56586   | user02db  | Sleep    |   541 |                                 | NULL                                                                                                |
| 4034799 | DB_USER02 | 59.64.129.90:56587   | user02db  | Sleep    |   541 |                                 | NULL                                                                                                |
| 4038566 | DB_USER14 | 59.64.129.112:13502  | user14db  | Query    | 56180 | System lock                     | create table test                  Select T1.N_Sector_ID as  N_Sector_ID, T2.N_Sector_ID as M_Sec |
| 4038675 | DB_USER14 | 59.64.129.112:13640  | user14db  | Query    | 56082 | Waiting for table metadata lock | drop table if exists `test`                                                                         |
| 4038796 | DB_USER14 | 59.64.129.112:13688  | user14db  | Query    | 55978 | Waiting for table metadata lock | drop table if exists `test`                                                                         |
| 4038836 | DB_USER14 | 59.64.129.112:13699  | user14db  | Query    | 55944 | Waiting for table metadata lock | drop table if exists `test`                                                                         |
| 4039014 | DB_USER14 | 59.64.129.112:14091  | user14db  | Query    | 55790 | Waiting for table metadata lock | drop table if exists `test`                                                                         |
| 4039095 | DB_USER14 | 59.64.129.112:14196  | user14db  | Query    | 55720 | Waiting for table metadata lock | create table test                  Select T1.N_Sector_ID as  N_Sector_ID, T2.N_Sector_ID as M_Sec |
| 4039471 | DB_USER14 | 59.64.129.112:14510  | user14db  | Query    | 55139 | Waiting for table metadata lock | create table test
                 Select T1.N_Sector_ID as  N_Sector_ID, T2.N_Sector_ID as M_Se |
| 4040157 | DB_USER14 | 59.64.129.112:14875  | user14db  | Query    | 53498 | Waiting for table metadata lock | create table test
                 Select T1.N_Sector_ID as  N_Sector_ID, T2.N_Sector_ID as M_Se |
| 4043858 | DB_USER14 | 59.64.129.112:2897   | user14db  | Query    | 51543 | Waiting for table metadata lock | drop table if exists `test`                                                                         |
| 4097435 | DB_USER31 | 114.242.249.53:24248 | user31db  | Sleep    |   171 |                                 | NULL                                                                                                |
| 4097438 | DB_USER31 | 114.242.249.53:24249 | user31db  | Sleep    |   172 |                                 | NULL                                                                                                |
| 4101472 | rdsRepl   | 172.16.37.91:43160   | NULL      | Sync msg |   554 | Receiving from client           | NULL                                                                                                |
| 4102081 | rdsAdmin  | localhost            | NULL      | Query    |     0 | starting                        | show processlist                                                                                    |
+---------+-----------+----------------------+-----------+----------+-------+---------------------------------+-----------------------------------------------------------------------------------------------------+
16 rows in set (0.00 sec)
```

Check the values in the **Time** and **State** columns. As shown in the preceding figure, the long-running transaction ID is **4038566**.

Step 3 Use SQL diagnosis of DAS to identify SQL statements that are executed frequently, consume a large amount of resources, or take a long time to execute. You can optimize the statements according to the diagnosis suggestions to ensure the stability of the database performance.

1. Log in to the DAS console.
2. In the navigation pane, choose **Intelligent O&M** > **Instance List**.
3. Click **Details** on the instance.

Figure 2-56 Instance list



We would much appreciate if you could complete our questionnaire on DAS. Your feedback will help us provide

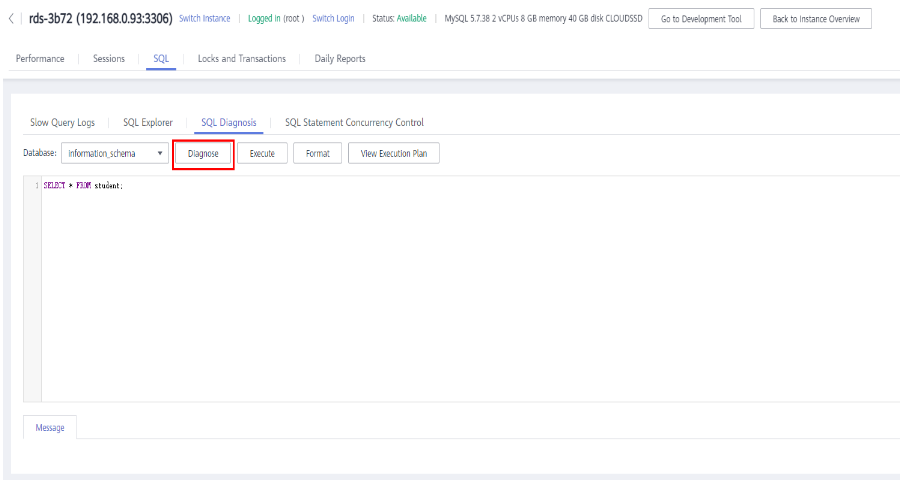
Used/Total Quota: 0/1. After paid instances are set, you can use advanced features like SQL statement concu

Last synchronized: Apr 11, 2023 16:54:12 GMT+08:00. Synchronize instances in a timely manner.

rds-3b72
192.168.0.93:3306
Free

4. Choose **SQL** > **SQL Diagnosis**.
5. Select a database, enter an SQL statement, and click **Diagnose**.

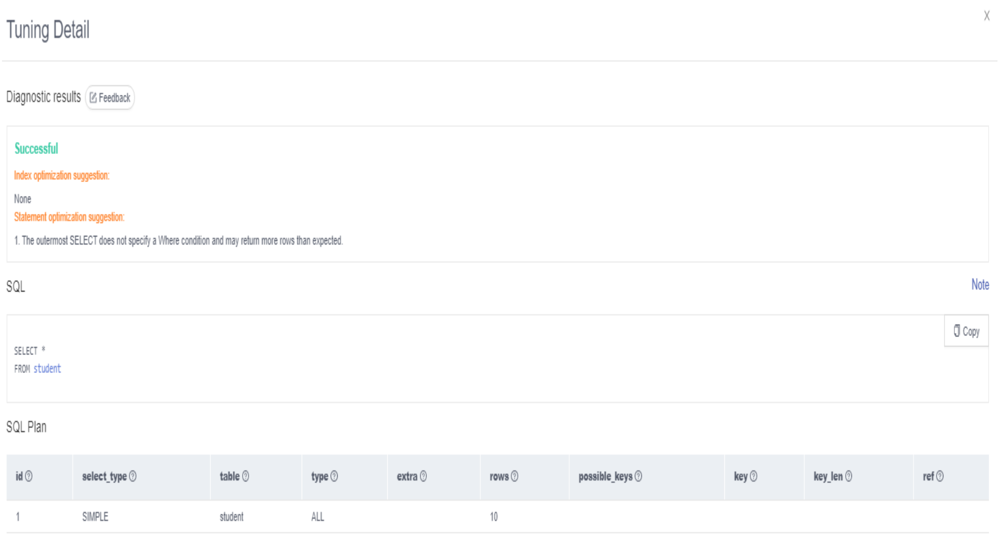
Figure 2-57 SQL diagnosis



```
SELECT * FROM student;
```

6. View diagnosis details and obtain statement optimization suggestions.

Figure 2-58 Tuning details



Successful

Index optimization suggestion:

None

Statement optimization suggestion:

1. The outermost SELECT does not specify a Where condition and may return more rows than expected.

```
SELECT *
FROM student
```

id	select_type	table	type	extra	rows	possible_keys	key	key_len	ref
1	SIMPLE	student	ALL		10				

NOTE

- Only the diagnosis of SELECT, INSERT, UPDATE, and DELETE statements is supported. An INSERT statement must contain a SELECT clause.
- SQL statements for querying system databases like **information_schema**, **test**, and **mysql** are not supported.
- SQL statements that use views are not supported.
- Using SQL diagnosis can obtain table structure and data distribution information (non-original). The obtained information is only used for logic diagnosis, but not stored on the DAS server.
- Obtaining table structure and data distribution information may cause additional load on the DB instance, but has little impact on its performance.
- Only the SQL diagnosis history is stored on the DAS server. You can delete it from the server permanently.

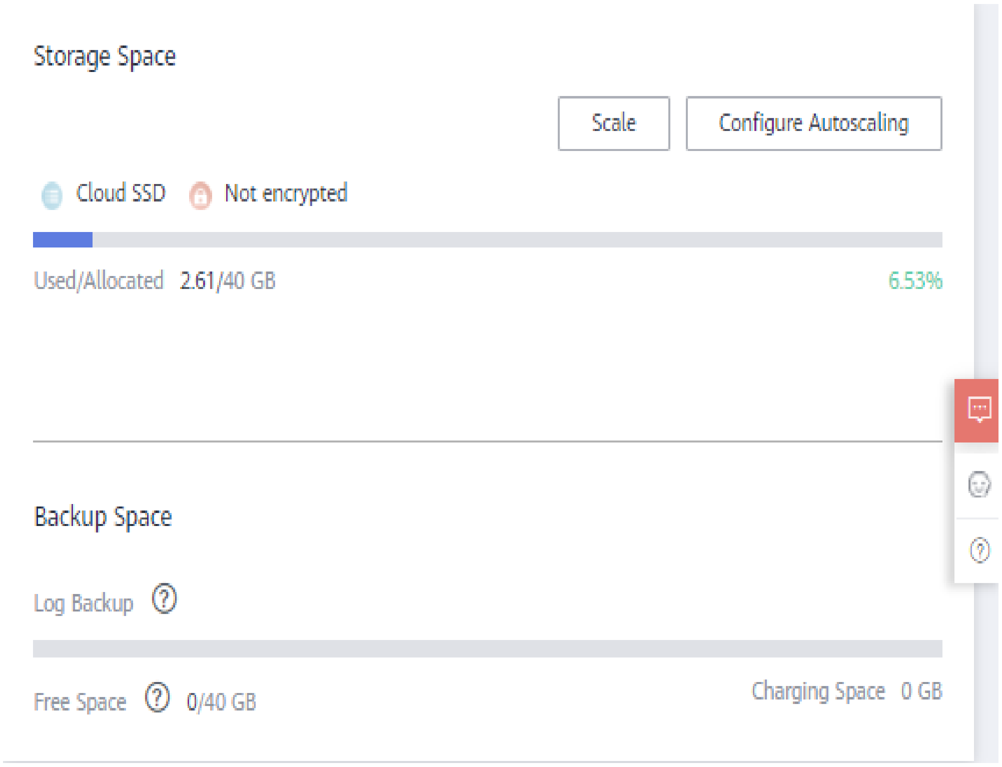
----End

2.11.2 RDS for MySQL Performance Tuning – Resolving Insufficient Storage Issues

Storage usage is an important metric for measuring the performance of your DB instances. If the available storage space is insufficient, your DB instance may encounter serious issues. For example, data cannot be backed up or written into databases, or scaling up storage takes an extended period of time.

Viewing Storage Space Usage

- On the **Basic Information** page, you can see how much of your instance storage and backup space has been used. However, this page does not provide any details about what different types of data are being stored.



Storage Space

Scale | Configure Autoscaling

Cloud SSD Not encrypted

Used/Allocated 2.61/40 GB 6.53%

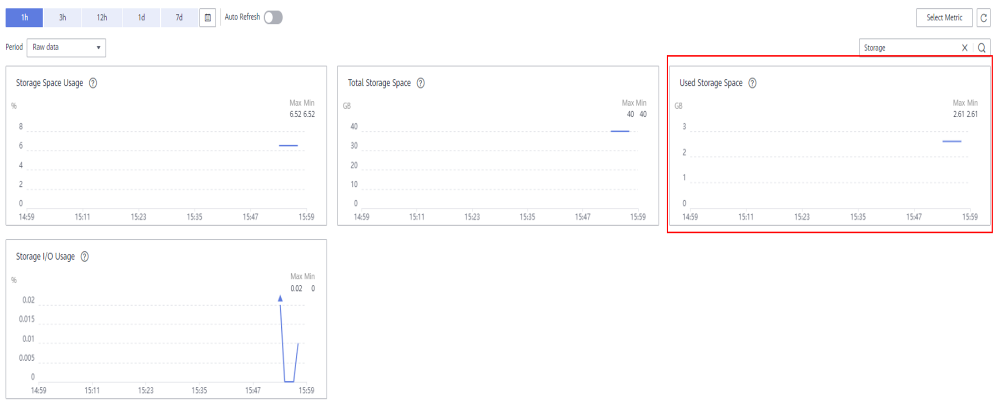
Backup Space

Log Backup

Free Space 0/40 GB

Charging Space 0 GB

- To view the historical usage and the changes over time, click **View Metrics** on the **Basic Information** page.



Insufficient Storage Caused by Excessive Indexes

- Cause and symptom

  In most cases, a table contains primary key indexes and secondary indexes. More secondary indexes mean that the table takes up more space.

- Solution

  Optimize the data structure of the table to reduce the number of secondary indexes.

Insufficient Storage Caused by Large Fields

- Cause and symptom

  If large fields of the binary large object (BLOB), TEXT, or VARCHAR data type are defined in the schema of a table, the table takes up a lot of space.

- Solution

  Compress data before you insert the data into the table.

Insufficient Storage Caused by Excessive Idle Tablespaces

- Cause and symptom

  If the fragmentation ratio of an InnoDB table is high, there will be an excessive number of idle tablespaces. InnoDB manages tablespaces by page. If some records of a full page are deleted and no new records are inserted into the positions these records were deleted from, a large number of tablespaces will be idle.

- Solution

  Run the **show table status like '<***Name of the table***>';** command to query idle tablespaces. If there are too many tablespaces, run the **optimize table '<***Name of the table***>';** command to manage the tablespaces.

Insufficient Storage Caused by Excessively Large Temporary Tables

- Cause and symptom

  When you perform a semi-join, distinct, or sort operation without using an index on a table, a temporary table is created. If the temporary table contains an excessive amount of data, the storage usage for the temporary table may be excessively high.

  When you execute data definition language (DDL) statements to rebuild tablespaces that are used to store the data of a large table, the temporary table that is generated from an index-based sort operation will also be large. If your DB instance runs MySQL 5.6 or MySQL 5.7, you cannot immediately add fields, and some DDL statements can be executed only on new tables. If you send requests to execute these DDL statements on a table, RDS creates a new table and executes these DDL statements on the new table. The new table is a replica of the original table. In this situation, you end up with two copies of the files when these DDL statements are being executed. After these DDL statements are executed, the original table is deleted.

- Solution

  View the plans that the DDL statements were based on. Check whether the **Using Temporary** field was specified.

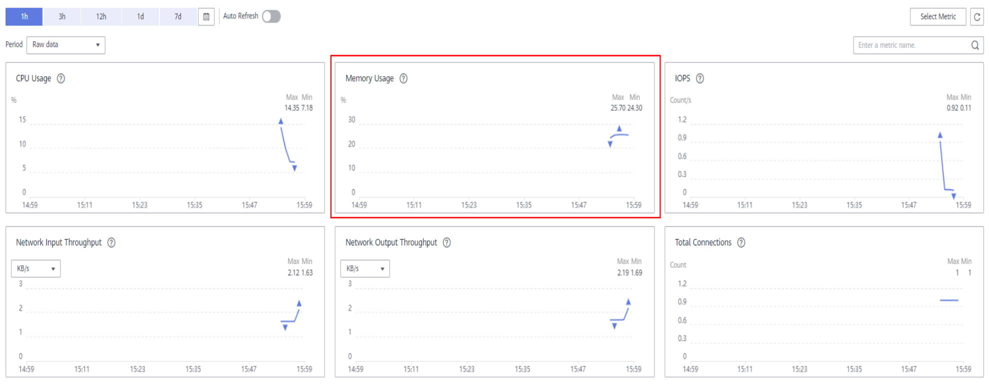
  Before you execute DDL statements on large tables, check whether your DB instance provides sufficient storage space. If the available storage space is insufficient, **scale up the storage space** of your DB instance before executing the statements.

2.11.3 RDS for MySQL Performance Tuning – Resolving High Memory Usage Issues

The memory usage and the buffer pool hit ratio are two important metrics for measuring the overall memory use of your DB instance. If the memory usage is excessively high, the risk of memory exhaustion arises. If the buffer pool hit ratio is low, a large number of data pages that are requested cannot be hit in the buffer pool. As a result, RDS needs to read data from disks. This increases I/O operations and query latencies.

Viewing the Memory Usage

- View the memory usage of instances using DBA Assistant. For details, see **Performance Metrics**.

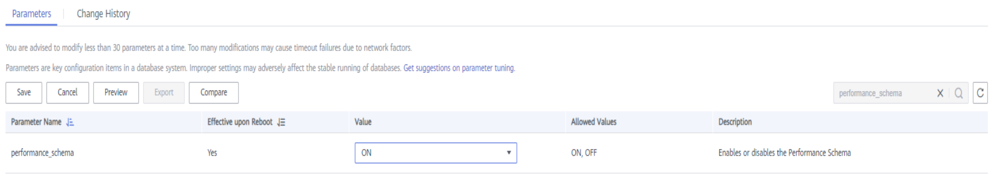


- You can also use performance_schema to configure memory instruments. This allows you to aggregate memory usage statistics into memory summary tables. For more information, see **official MySQL documentation**.

NOTICE

MySQL 5.6 does not support performance_schema for memory usage monitoring.

1. Set **performance_schema** to **ON** by referring to **Modifying RDS for MySQL Instance Parameters**.



You are advised to modify less than 30 parameters at a time. Too many modifications may cause timeout failures due to network factors.

Parameters are key configuration items in a database system. Improper settings may adversely affect the stable running of databases. Get suggestions on parameter tuning.

Parameter Name	Effective upon Reboot	Value	Allowed Values	Description
performance_schema	Yes	ON	ON, OFF	Enables or disables the Performance Schema

2. Reboot your DB instance for the modification to take effect.
3. Query the sys.memory_global_total view for the total memory usage.

   select * from sys.memory_global_total;
4. Query the sys.session view.
   a. Check the **current_memory** field for memory usage of each session.

      select thd_id,conn_id, current_memory from sys.session;
   b. Check the memory usage details of the session thread with high memory usage in performance_schema.

      select * from memory_summary_by_thread_by_event_name where thread_id= *"ID of the thread with high memory usage"* **order by CURRENT_NUMBER_OF_BYTES_USED;**
5. Query the sys.memory_by_thread_by_current_bytes view.
   a. Check the **current_allocated** field for memory usage of each background thread.

select thread_id, user, current_allocated from memory_by_thread_by_current_bytes;

b. Check the memory usage details of the background thread with high memory usage in performance_schema.

select * from memory_summary_by_thread_by_event_name where thread_id= *"ID of the thread with high memory usage"* **order by CURRENT_NUMBER_OF_BYTES_USED**;

6. Query the memory_global_by_current_bytes view for memory usage statistics by allocation type.

   select event_name,current_alloc from sys.memory_global_by_current_bytes where event_name not like 'memory/performance_schema%' ;

7. Analyze the cause of high memory usage based on the query results.

Common Causes for Excessively High Memory Usage

Generally, the InnoDB buffer pool consumes the most memory. The maximum memory that can be consumed by a buffer pool depends on the parameter settings of the pool. In addition, most of the memory is allocated dynamically and then adjusted as the requests are processed. The memory usage includes the memory that is used by in-memory temporary tables, prefetch caches, table caches, hash indexes, and row lock objects. For more information about the memory usage and parameter limits, see the **official MySQL documentation**.

- Multiple Statements in One Query

  You can include multiple SQL statements in a single query and separate them using semicolons (;). When receiving the multi-statement query, RDS for MySQL processes the SQL statements one by one. However, some memory can be released only after all these SQL statements are executed.

  If too many SQL statements are sent in a single batch, the memory that is used by various objects to batch execute these SQL statements increases by up to a few hundred MB. This can exhaust the available memory for the MySQL process.

  A multi-statement query causes a sharp increase of network traffic. You can detect a sudden increase through network traffic monitoring or SQL Explorer. Therefore, multi-statement queries are not recommended.

- Buffer pool issues

  The data pages of all tables are stored in the buffer pool. If the requested data pages are hit in the buffer pool, RDS does not perform physical I/O operations. In this case, RDS executes SQL statements very quickly. In addition, the buffer pool uses the least recently used (LRU) caching algorithm to manage the data pages. This algorithm allows the buffer pool to store all of the dirty pages in the flush list.

  The InnoDB buffer pool accounts for the largest proportion of the memory that is provided by your DB instance.

  The following issues are related to the buffer pool:

  – If data pages are not sufficiently pre-warmed, query latency increases. This can be an issue if you restart your DB instance, read cold data, or if there is a low buffer pool hit ratio. Before you upgrade instance specifications or launch a sales promotion, we recommend that you sufficiently pre-warm data pages.

  – There may be too many dirty pages. If a dirty page is not updated for a long period of time, for instance, if the difference between the earliest and current log sequence numbers (LSNs) of the dirty page exceeds 76%, a user thread is triggered to synchronously update the page. This significantly decreases the performance of your DB instance. To fix this issue, you can balance the write loads, prevent excessively high throughput for write operations, reconfigure the parameters that specify how to update dirty pages, or upgrade instance specifications.

- Temporary table issues

  The in-memory temporary table size is limited by the **tmp_table_size** and **max_heap_table_size** parameters. If the size of an in-memory temporary table exceeds the configured limit, the in-memory temporary table is converted into an on-disk temporary table. If too many temporary tables are created over a number of connections, the memory usage of your DB instance suddenly increases. MySQL 8.0 provides a new TempTable engine. With this engine, the total size of the in-memory temporary tables that are created by all threads must be smaller than the value of **temptable_max_ram**. The default value of this parameter is 1 GB. If the total size exceeds the value of this parameter, earlier in-memory temporary tables are converted into on-disk temporary tables.

- Other issues

  If too many tables are created on your DB instance or if the QPS is high, the table cache may consume a certain amount of memory. We recommend that you do not create too many tables or set **table_open_cache** to a too large value.

  The default memory usage for adaptive hash indexes is 1/64 of the buffer pool size. If you query or write large fields of the BLOB data type, memory is dynamically allocated to these large fields. This also increases the memory usage of your DB instance.

  If the memory usage increases abnormally or the memory runs out altogether, you can locate the cause by referring to **official MySQL documentation**.

2.11.4 RDS for MySQL Performance Tuning – Resolving High I/O Issues

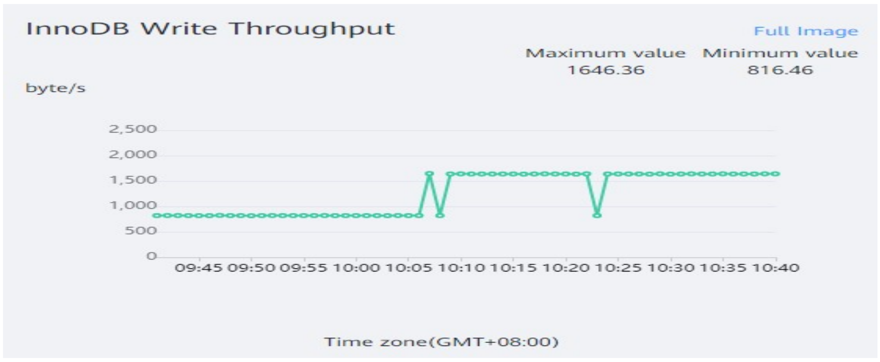
This section describes how to troubleshoot high I/O issues for RDS for MySQL DB instances. The I/O performance of your instance depends on three factors: the storage media, the database engine architecture, and the SQL statements that are executed to scan or modify a specific amount of data.

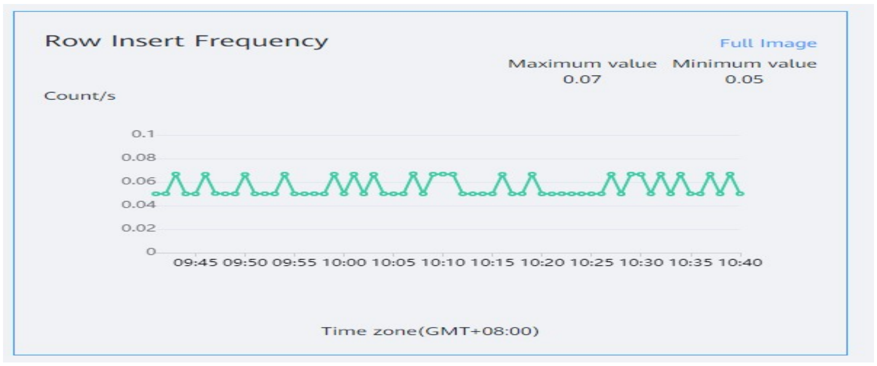
High I/O Caused by High Throughput

- Symptom

  If your application frequently initiates requests to update, delete, and insert data on the tables containing many indexes or large fields, the I/O of your instance significantly increases due to the data reads and the flushes of dirty pages.

To view read/write performance, click **View Metrics** in the **Operation** column.





- Solution

  Modify instance parameters to reduce the read/write frequency, upgrade instance specifications, or tune the settings of dirty page flushing parameters. The dirty page flushing parameters are explained as follows:

  innodb_max_dirty_pages_pct: the percentage of dirty pages allowed in the buffer pool. Default value: **75**.

  innodb_io_capacity: the maximum number of I/O operations allowed by InnoDB per second for each background task. The value of this parameter affects the speed at which RDS flushes dirty pages to the disk and the speed at which RDS writes data to the buffer pool. The default value varies depending on the disk type.

  innodb_io_capacity_max: the maximum number of I/O operations allowed by InnoDB per second for each background task when the flushing activity falls behind. The value of this parameter is greater than the value of **innodb_io_capacity**. The default value of **innodb_io_capacity_max** is **40000**.

High I/O Caused by Temporary Tables

- Symptom

  If the temporary directory size is too big, RDS may have created large temporary tables due to operations such as the sorting and deduplication of slow SQL statements. This increases the I/O of your instance. Data writes to temporary tables also increase the I/O of your instance.

  To view the temporary table creation, click **View Metrics** in the **Operation** column.

- Solution

  On the **Slow Query Logs** page of the RDS console, download and view the SQL statements that are executed slowly. Analyze the information such as execution duration of the slow SQL statements and optimize the SQL statements based on the analysis result.

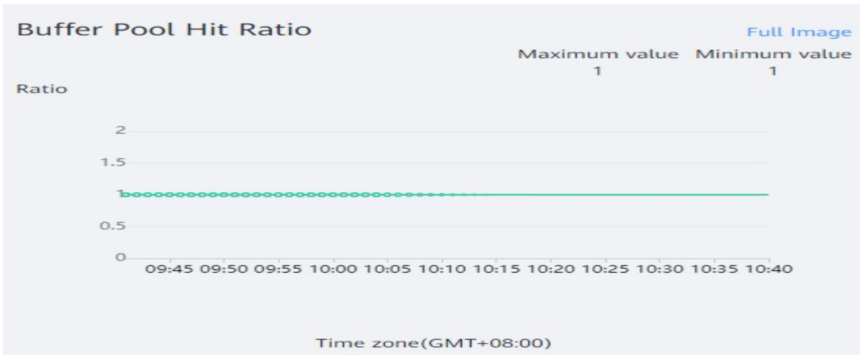
- For details about how to download slow query logs, see **Viewing and Downloading Slow Query Logs**.
- For details about how to optimize slow SQL statements, see **Troubleshooting Slow SQL Issues for RDS for MySQL DB Instances**.

High I/O Caused by Cold Data Reads

- Symptom

  If the data that is queried or modified by using SQL statements cannot be hit in the buffer pool, RDS needs to read the data from disks. This may significantly increase the I/O of your instance.

  To view the buffer pool hit ratio, click **View Metrics** in the **Operation** column.



- Solution

  Redesign the cache policy based on your service scenario or **upgrade instance specifications**.

High I/O Caused by Binlog Writes from Large Transactions

- Symptom

  A transaction only writes log records into binlog files when it is committed. If your application runs a large transaction, the transaction may write a few dozen GB of data into binlog files, for example, if the transaction contains a DELETE statement that is used to delete a large number of rows. When these binlog files are flushed to the disk, the I/O of your instance significantly increases.

- Solution

  You are advised to split large transactions. This allows you to reduce the flushes of dirty pages to the disk.

Introduction to the InnoDB I/O System

InnoDB uses an independent I/O system to read and write data pages. If a data page that is requested by an SQL statement cannot be hit in the buffer pool, physical I/O operations are performed to read and write data to the disk.

- Operations to read data pages

  The underlying read interface is called based on synchronous I/O to read data pages.

- Operations to write data pages

  Take the flushes of dirty pages for example. Background I/O threads are called based on asynchronous I/O to asynchronously flush dirty pages to the disk.

  In addition to I/O operations on common data files, a number of other operations may also significantly increase the I/O of your instance. These operations include the operations to write redo logs, undo logs, and binlogs, the operations to sort temporary tables, and the operations to rebuild tablespaces due to DDL statements.

2.11.5 Troubleshooting Slow SQL Issues for RDS for MySQL DB Instances

This section describes how to troubleshoot slow SQL statements on MySQL DB instances. For any given service scenario, query efficiency depends on the architecture and on the database table and index design. Poorly designed architecture and indexes will cause many slow SQL statements.

Slow SQL Statements Caused by SQL Exceptions

- Causes and symptoms

  There are many causes for SQL exceptions, for example, unsuitable database table structure design, missing indexes, or too many rows that need to be scanned.

  On the slow query logs page of the management console, you can download slow query logs to identify the slow SQL statements and see how long they took to execute. For details, see **Viewing and Downloading Slow Query Logs**.
- Solution

  Optimize the SQL statements that you need to execute.

Slow SQL Statements Caused by DB Instance Limits

- Causes and symptoms

  DB instance performance can be limited because:

  – Your workloads have been increasing but the storage has not been scaled up accordingly.
  – The performance of your DB instance has been deteriorating as the physical server of the instance ages.
  – The amount of data has been increasing, and the data structure has been changing.

  You can view the resource usage of the DB instance on the console. If the values of all resource usage metrics are close to 100%, your DB instance may reach its maximum performance. For details, see **Viewing Performing Metrics**.
- Solution

  Upgrade the instance class. For details, see **Changing a DB Instance Class**.

Slow SQL Statements Caused by Version Upgrades

- Causes and symptoms

  Upgrading your DB instance may change the SQL execution plan. The join types determined in the execution plan are, in descending order of efficiency:

  system > const > eq_ref > ref > fulltext > ref_or_null > index_merge > unique_subquery > index_subquery > range > index > all

  For more information, see **official MySQL documentation**.

  If your application frequently resends query requests that specify range and index joins but RDS processes these query requests slowly, a number of SQL statements are parallelized. In this case, your application is slow to release threads. As a result, the connections in the connection pool get depleted, affecting all the workloads on your DB instance.

  You can log in to the console to see how many current connections your DB instance has established. For details, see **Viewing Performance Metrics**.

- Solution

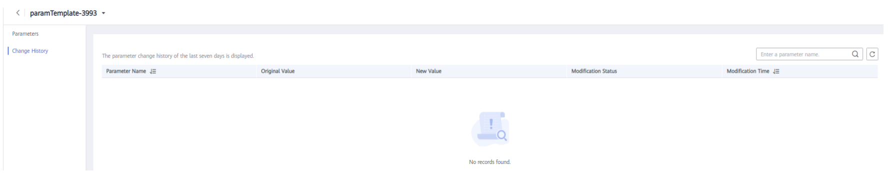
  Analyze the index usage and the number of rows to scan, estimate the query efficiency, reconstruct SQL statements, and adjust indexes. For details, see **Executing SQL Plan**.

Slow SQL Statements Caused by Inappropriate Parameter Settings

- Causes and symptoms

  Inappropriate settings of some parameters (such as **innodb_spin_wait_delay**) can impact performance.

  You can view parameter modifications on the console. For details, see **Viewing Parameter Change History**.

  paramTemplate-3993

  The parameter change history of the last seven days is displayed.

Parameter Name	Original Value	New Value	Modification Status	Modification Time

  No records found.

- Solution

  Modify related parameters based on your specific service scenario.

Slow SQL Statements Caused by Batch Operations

- Causes and symptoms

  A large number of operations are performed to import, delete, and query data.

  You can view **Total Storage Space**, **Storage Space Usage**, and **IOPS** on the console. For details, see **Viewing Performance Metrics**.

- Solution

  Perform batch operations during off-peak hours, or split them.

Slow SQL Statements Caused by Scheduled Tasks

- Causes and symptoms

  If the load of your DB instance changes regularly over time, there may be scheduled tasks causing this.

You can view **DELETE Statements per Second**, **INSERT Statements per Second**, **INSERT_SELECT Statements per Second**, **REPLACE Statements per Second**, **REPLACE_SELECTION Statements per Second**, **SELECT Statements per Second**, and **UPDATE Statements per Second** on the console to determine whether the load has been changing regularly. For details, see **Viewing Monitoring Metrics**.

- Solution

  Adjust the time when scheduled tasks are run. You are advised to run scheduled tasks during off-peak hours and change the maintenance window to off-peak hours. For details, see **Changing the Maintenance Window**.

2.12 Security Best Practices

Security is a shared responsibility between Huawei Cloud and you. Huawei Cloud is responsible for the security of cloud services to provide a secure cloud. As a tenant, you should properly use the security capabilities provided by cloud services to protect data, and securely use the cloud. For details, see **Shared Responsibilities**.

This section provides actionable guidance for enhancing the overall security of using RDS for MySQL. You can continuously evaluate the security status of your RDS for MySQL DB instances and enhance their overall security defense by combining different security capabilities provided by RDS for MySQL. By doing this, data stored in RDS for MySQL DB instances can be protected from leakage and tampering both at rest and in transit.

Make security configurations from the following dimensions to meet your service needs.

- **Optimizing Database Connection Configurations to Reduce Network Attack Risks**
- **Properly Managing Database Accounts and Passwords to Reduce Data Leakage Risks**
- **Strengthening Permissions Management to Reduce Related Risks**
- **Enabling Database Audit for Post-Event Backtracking**
- **Configuring Data Backup to Ensure Data Reliability**
- **Encrypting Data Before Storage**
- **Hardening Sensitive Parameters**
- **Using the Latest Database Version for Better Experience and Security**
- **Using Other Cloud Services for Additional Data Security**

Optimizing Database Connection Configurations to Reduce Network Attack Risks

1. **Do not bind an EIP to your RDS for MySQL instance for access over the Internet.**

   Do not deploy your instance on the Internet or in a demilitarized zone (DMZ). Instead, deploy it on an intranet and use routers or firewalls to protect it. Do not bind an EIP to your instance for access over the Internet to prevent unauthorized access and DDoS attacks. If you have bound an EIP to your instance, you are advised to **unbind it**. If you do need an EIP, **configure security group rules** to restrict the source IP addresses that can access your instance.

2. **Do not use the default port number.**

   RDS for MySQL instances use port 3306 by default, which is more vulnerable to malicious attacks. **Change the port number** for your DB instance.

3. **Restrict available resources of a database user.**

   If the resources available to a database user are not restricted, the system may be overloaded when the user is attacked, causing DoS. Restricting available resources of a database user can prevent excessive resource consumption caused by excessive resource occupation. To prevent service availability from being affected in heavy-load scenarios, you need to configure available resources for a database user based on the service model using the following SQL statements:

   ```
   alter user  '<user>'@'<hostname>' with max_queries_per_hour <queries_num>;
   alter user  '<user>'@'<hostname>' with max_user_connections <connections_num>;
   alter user  '<user>'@'<hostname>' with max_updates_per_hour <updates_num>;
   alter user  '<user>'@'<hostname>' with max_connections_per_hour
   <connections_per_hour>;
   ```

   – *<user>* indicates the name of the user you want to configure resources for.
   – *<hostname>* indicates the name of the host.
   – *<queries_num>* indicates the maximum number of queries allowed for the user per hour.
   – *<connections_num>* indicates the maximum number of connections allowed for the user.
   – *<updates_num>* indicates the maximum number of updates allowed for the user per hour.
   – *<connections_per_hour>* indicates the maximum number of connections allowed for the user per hour.

4. **Do not use the wildcard % for the host name.**

   The host name specifies which host is allowed to connect to your database. The host name corresponds to the **host** field in the **user** table. If the host name is set to the wildcard %, the user accepts connections from any IP address, increasing risks of attacks to the database. To minimize the attack risks, you are advised to **set the host IP address** to a specific network segment or IP address.

5. **Limit the waiting time of idle database connections.**

   Each connection to an RDS for MySQL server consumes memory, and the maximum number of connections supported is limited. If the RDS for MySQL server has a large number of idle connections, too much memory is consumed and the maximum number of connections can be reached. In this case, the error message "too many connections" will be reported if a new connection is established. You need to set the waiting time for idle connections to ensure that idle connections are cleared in time. Change the values of **wait_timeout** and **interactive_timeout** by referring to **Modifying Parameters of an RDS for MySQL Instance**.

6. **Ensure that SSL is enabled by default.**

If SSL is not configured, data transmitted between the MySQL client and server is in plaintext, which is vulnerable to eavesdropping, tampering, and man-in-the-middle attacks. To improve data transmission security, you are advised to add the **REQUIRE SSL** attribute and **configure SSL** for database users.

You can use the following SQL statements:

```
create user '<user>'@'<hostname>' REQUIRE SSL;
alter user '<user>'@'<hostname>' REQUIRE SSL;
```

Properly Managing Database Accounts and Passwords to Reduce Data Leakage Risks

1. **Periodically change the password of the administrator.**

   The default database administrator account **root** has high permissions. You are advised to periodically change the password of user **root** by referring to **Resetting the Administrator Password to Restore Root Access**.

2. **Configure password complexity.**

   As a collector of information, the database system is easy to be the target of attacks. You need to keep your database account and password secure to prevent disclosure. In addition, you are advised to configure the complexity of your password to avoid weak passwords. For details, see "Setting Password Complexity" in **Database Account Security**.

3. **Configure a password expiration policy.**

   Using the same password too long makes it easier for hackers to crack or guess your password. You are advised to **configure a password expiration policy** to restrict the time when a password can be used.

Strengthening Permissions Management to Reduce Related Risks

1. **Do not create stored procedures or functions as the administrator.**

   Stored procedures and functions are run as creators by default. If you create stored procedures and functions as the administrator, regular users can run them through privilege escalation, so do not use the administrator to create stored procedures or functions.

2. **Review and harden permission configurations.**

   Check whether the following permission configurations meet security requirements. If they do not meet security requirements, harden the configurations.

   – Ensure that only the administrator can perform operations on the **mysql.user** table.
   – Ensure that the **Process_priv** permission can be granted only to the administrator.
   – Ensure that the **Create_user_priv** permission can be granted only to the administrator.
   – Ensure that the **Grant_priv** permission can be granted only to the administrator.
   – Ensure that the **Reload_priv** permission can be granted only to the administrator.

- Ensure that the replication account has only the **replication slave** permission.
- Ensure that the database metric monitoring user has only the **replication client** permission.

Example: If a non-administrator user has the **Process** permission, run the following SQL statement to revoke the **Process** permission:

```
revoke process on *.* from <your_account>;
```

In the preceding command, *<your_account>* indicates the name of the user whose **Process** permission needs to be revoked.

Enabling Database Audit for Post-Event Backtracking

The database audit function records all user operations on the database in real time. By recording, analyzing, and reporting user access to the database, database audit helps generate compliance reports and trace accidents, improving data asset security. For details, see **Enabling SQL Audit**.

Configuring Data Backup to Ensure Data Reliability

1. **Enable data backup.**

   RDS for MySQL supports automated and manual backups. You can periodically back up databases. If a database is faulty or data is damaged, you can restore the database using backups to ensure data reliability. For details, see **Data Backups**.

2. **Configure a binlog clearing policy.**

   Binlogs continuously increase as services run. You need to configure a clearing policy to prevent disk expansion. Configure binlog retention period by referring to **Setting a Local Retention Period for RDS for MySQL Binlogs**.

Encrypting Data Before Storage

To improve user data security, you are advised to **enable server-side encryption**. After server-side encryption is enabled, data will be encrypted on the server before being stored when you create a DB instance or scale up storage space, reducing data leakage risks.

Hardening Sensitive Parameters

1. **Set local_infile to 0**.

   If **local_infile** is set to **1**, the database client can use the **load data local** syntax to load local files to database tables. For example, when a web server functions as a database client to connect to a database, if the web server has an SQL injection vulnerability, an attacker can use the **load data local** command to load sensitive files on the web server to the database, causing information leakage. You are advised to set **local_infile** to **0** by referring to **Modifying Parameters of an RDS for MySQL Instance**.

2. **Set the sql_mode parameter to a value containing STRICT_ALL_TABLES.**

   When attempting to launch an attack, an attacker may enter various parameters in a trial-and-error manner. If the server adapts to incorrect statements, database data may be leaked. Therefore, **STRICT_ALL_TABLES** is recommended. Even if an error occurs in other rows than the first row, the statement will be discarded once an invalid data value is found. This method maximally ensures that database information is not disclosed. You are advised to set **sql_mode** to a value containing **STRICT_ALL_TABLES** by referring to **Modifying Parameters of an RDS for MySQL Instance**.

Using the Latest Database Version for Better Experience and Security

The MySQL community irregularly discloses newly discovered vulnerabilities. RDS for MySQL evaluates the actual risks of database kernel versions and release new database kernel versions accordingly. To improve the usability and security of the database system, you are advised to use **the latest database version**.

Using Other Cloud Services for Additional Data Security

To obtain extended data security capabilities, you are advised to use **Database Security Service (DBSS)**.

3 RDS for PostgreSQL

3.1 Creating Databases

This section describes how to create databases using Data Admin Service (DAS) or CLIs after you purchase an RDS instance.

3.1.1 Database Naming Rules

- The keyword length of RDS for PostgreSQL is limited to 63 bytes. Therefore, it is recommended that the length of a database name be no more than 30 characters.
- A database name can contain only lowercase letters, underscores (_), and digits. Do not use reserved keywords in database names or begin a database name with pg, digits, or underscores (_). For details about reserved keywords, see the **official documentation**.

3.1.2 Creating a Database Through DAS

This section describes how to log in a DB instance and create databases in it using Data Admin Service (DAS).

Procedure

Step 1 **Log in to the management console**.

Step 2 Click in the upper left corner and select a region and a project.

Step 3 Click in the upper left corner of the page and choose **Databases** > **Relational Database Service**.

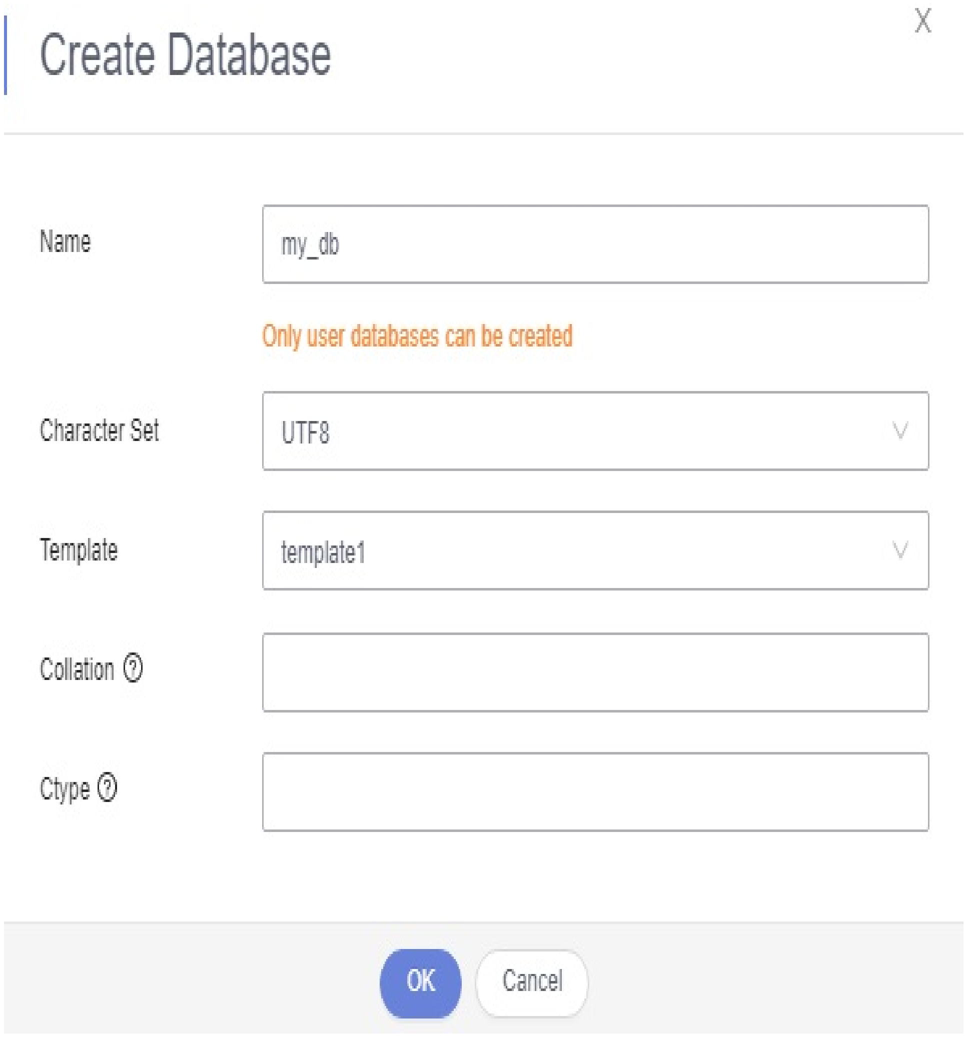
Step 4 On the **Instances** page, locate the target DB instance and click **Log In** in the **Operation** column.

Alternatively, click the target DB instance on the **Instances** page. On the displayed **Basic Information** page, click **Log In** in the upper right corner.

Step 5 On the displayed login page, enter the username and password and click **Log In**.

Step 6 Click **Create Database**. In the displayed dialog box, set the required parameters. For details, see **Parameter Description**.

Step 7 Confirm the information and click **OK**.



Create Database

Name: my_db

Only user databases can be created

Character Set: UTF8

Template: template1

Collation:

Ctype:

----End

3.1.3 CREATE DATABASE Options

When creating databases, you can specify a template database and set different character sets and collations for each database.

This section describes how to use the **CREATE DATABASE** command to configure the template database, character set, LC_COLLATE, and LC_CTYPE. For details about other parameters, see the **official documentation**.

```
CREATE DATABASE name
[ [ WITH ] [ OWNER [=] user_name ]
       [ TEMPLATE [=] template ]
       [ ENCODING [=] encoding ]
       [ LC_COLLATE [=] lc_collate ]
       [ LC_CTYPE [=] lc_ctype ]
```

```
    [ TABLESPACE [=] tablespace_name ]
    [ ALLOW_CONNECTIONS [=] allowconn ]
    [ CONNECTION LIMIT [=] connlimit ]
    [ IS_TEMPLATE [=] istemplate ] ]
```

3.1.3.1 TEMPLATE Option

RDS for PostgreSQL provides two default templates for you to create a database: template0 and template1. Their differences are as follows:

- When template1 is used, the character set or collation defined in this template cannot be changed. For details, see **Configuring the Collation of a Database in a Locale**.
  ```
  CREATE DATABASE my_db WITH TEMPLATE template1 ;
  ```
- When template0 is used, you can change the character set and collation. For details, see **Configuring the Collation of a Database in a Locale**.
  ```
  CREATE DATABASE my_db WITH ENCODING = 'UTF8' LC_COLLATE ='zh_CN.utf8'
  LC_CTYPE ='zh_CN.utf8' TEMPLATE = template0 ;
  ```
- If no template is specified during database creation, template1 is used by default. You can also specify a custom template to create a database.
  ```
  CREATE DATABASE my_db WITH TEMPLATE = mytemplate;
  ```

3.1.3.2 LC_COLLATE and LC_CTYPE Options

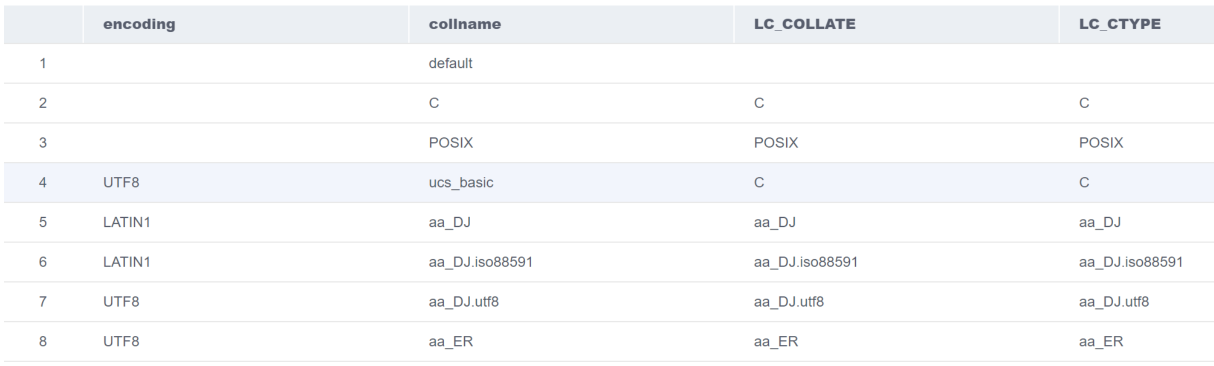
This section describes how to specify LC_COLLATE (string sort order) and LC_CTYPE (character classification) using **CREATE DATABASE**.

Querying Character Sets Supported by LC_COLLATE and LC_CTYPE

Run the following SQL statement to query character sets (also called encodings) supported by LC_COLLATE and LC_CTYPE:

```
SELECT pg_encoding_to_char(collencoding) AS encoding,collname,collcollate AS
"LC_COLLATE",collctype AS "LC_CTYPE" FROM pg_collation;
```

If **encoding** is empty, LC_COLLATE supports all character sets.



	encoding	collname	LC_COLLATE	LC_CTYPE
1		default		
2		C	C	C
3		POSIX	POSIX	POSIX
4	UTF8	ucs_basic	C	C
5	LATIN1	aa_DJ	aa_DJ	aa_DJ
6	LATIN1	aa_DJ.iso88591	aa_DJ.iso88591	aa_DJ.iso88591
7	UTF8	aa_DJ.utf8	aa_DJ.utf8	aa_DJ.utf8
8	UTF8	aa_ER	aa_ER	aa_ER

Configuring the Collation of a Database in a Locale

Collation information includes LC_COLLATE and LC_CTYPE. For details, see the **official documentation**.

- LC_COLLATE

The default value is **en_US.utf8**.

Comparison of the same string in different collations may have different results.

For example, after you execute **SELECT 'a'>'A';**, the result is **false** if this parameter is set to **en_US.utf8** and the result is **true** if this parameter is set to **C**. If you need to migrate a database from Oracle to RDS for PostgreSQL, set LC_COLLATE to **C**. You can query the supported collations from the **pg_collation** table.

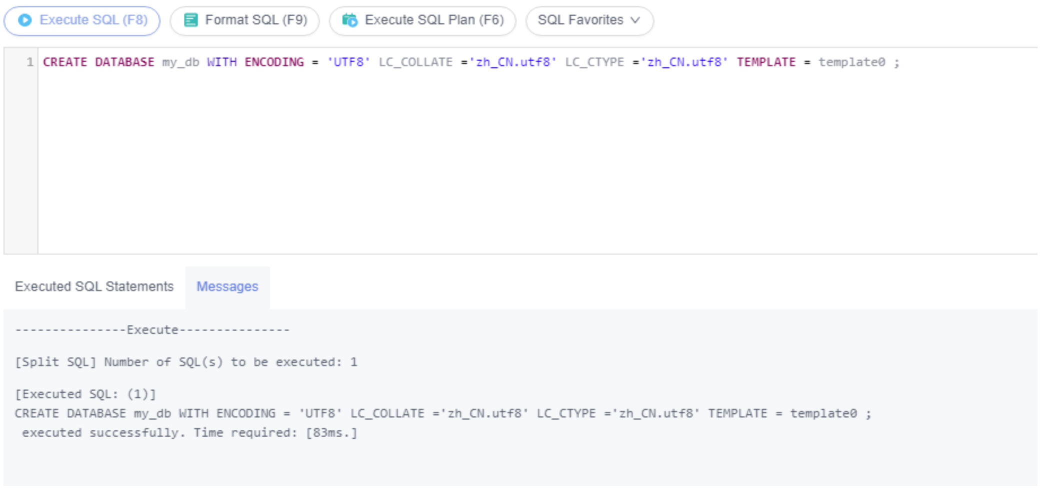
- LC_CTYPE

  It is used to classify if a character is a digit, uppercase letter, lowercase letter, and so on. You can query the supported character classifications from the **pg_collation** table.

How to Use

Run the following command to create a database with LC_COLLATE and LC_CTYPE set to **zh_CN.utf8**:

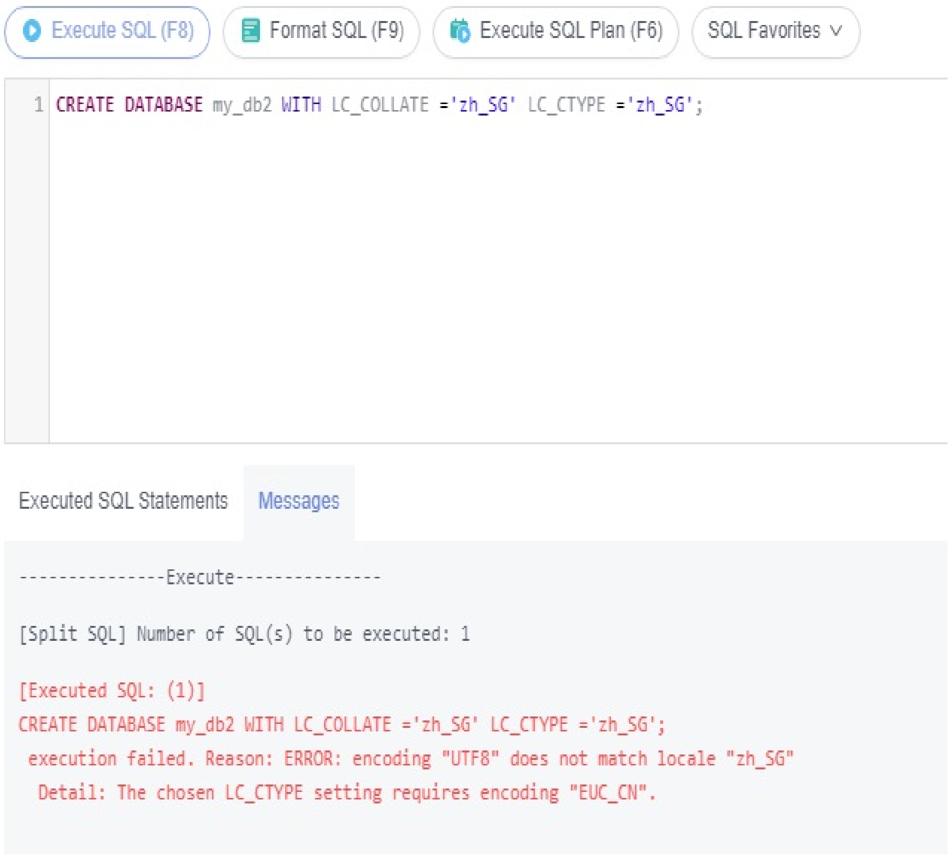
```
CREATE DATABASE my_db WITH ENCODING = 'UTF8' LC_COLLATE ='zh_CN.utf8' LC_CTYPE ='zh_CN.utf8' TEMPLATE = template0 ;
```



```
CREATE DATABASE my_db WITH ENCODING = 'UTF8' LC_COLLATE ='zh_CN.utf8' LC_CTYPE ='zh_CN.utf8' TEMPLATE = template0 ;
```

```
--------------Execute--------------
[Split SQL] Number of SQL(s) to be executed: 1
[Executed SQL: (1)]
CREATE DATABASE my_db WITH ENCODING = 'UTF8' LC_COLLATE ='zh_CN.utf8' LC_CTYPE ='zh_CN.utf8' TEMPLATE = template0 ;
 executed successfully. Time required: [83ms.]
```

If the specified LC_COLLATE is incompatible with the character set, error information similar to the following is displayed:



```
CREATE DATABASE my_db2 WITH LC_COLLATE ='zh_SG' LC_CTYPE ='zh_SG';
```

```
---------------Execute---------------

[Split SQL] Number of SQL(s) to be executed: 1

[Executed SQL: (1)]
CREATE DATABASE my_db2 WITH LC_COLLATE ='zh_SG' LC_CTYPE ='zh_SG';
 execution failed. Reason: ERROR: encoding "UTF8" does not match locale "zh_SG"
  Detail: The chosen LC_CTYPE setting requires encoding "EUC_CN".
```

NOTE

1. The specified LC_COLLATE and LC_CTYPE must be compatible with the target character set. Otherwise, an error is reported.
2. The LC_COLLATE and LC_CTYPE settings of an existing database cannot be changed by running the **ALTER DATABASE** statement. You can change them while creating a new database and then import your data to the new database.

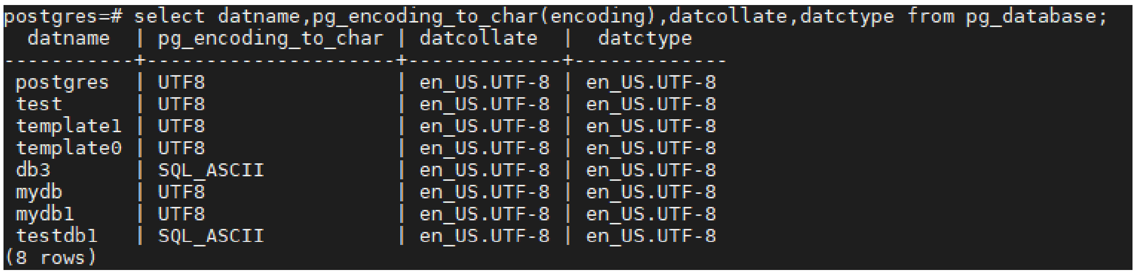
3.1.4 FAQ

3.1.4.1 How Do I View the Created Databases and the Character Sets, LC_COLLATE, and LC_CTYPE Information of the Databases?

- To view the created databases, run the psql meta-command **\ l**.

```
postgres=# \l
                                  List of databases
   Name    |  Owner   | Encoding  |  Collate    |    Ctype    |   Access privileges
-----------+----------+-----------+-------------+-------------+-----------------------
 db3       | postgres | SQL_ASCII | en_US.UTF-8 | en_US.UTF-8 |
 mydb      | postgres | UTF8      | en_US.UTF-8 | en_US.UTF-8 |
 mydb1     | postgres | UTF8      | en_US.UTF-8 | en_US.UTF-8 |
 postgres  | postgres | UTF8      | en_US.UTF-8 | en_US.UTF-8 |
 template0 | postgres | UTF8      | en_US.UTF-8 | en_US.UTF-8 | =c/postgres          +
           |          |           |             |             | postgres=CTc/postgres
 template1 | postgres | UTF8      | en_US.UTF-8 | en_US.UTF-8 | =c/postgres          +
           |          |           |             |             | postgres=CTc/postgres
```

- To view the character sets, LC_COLLATE, and LC_CTYPE information, query the **pg_database** system catalog.

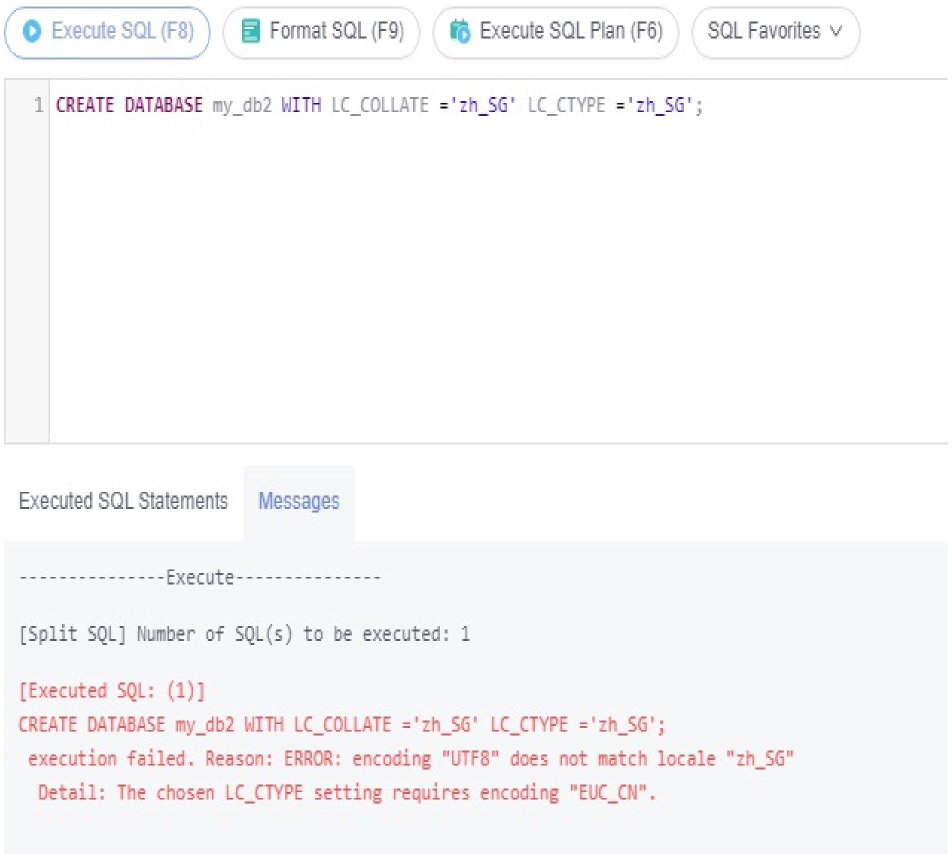


```
postgres=# select datname,pg_encoding_to_char(encoding),datcollate,datctype from pg_database;
  datname   | pg_encoding_to_char | datcollate  |  datctype
-----------+---------------------+-------------+-------------
 postgres   | UTF8                | en_US.UTF-8 | en_US.UTF-8
 test       | UTF8                | en_US.UTF-8 | en_US.UTF-8
 template1  | UTF8                | en_US.UTF-8 | en_US.UTF-8
 template0  | UTF8                | en_US.UTF-8 | en_US.UTF-8
 db3        | SQL_ASCII           | en_US.UTF-8 | en_US.UTF-8
 mydb       | UTF8                | en_US.UTF-8 | en_US.UTF-8
 mydb1      | UTF8                | en_US.UTF-8 | en_US.UTF-8
 testdb1    | SQL_ASCII           | en_US.UTF-8 | en_US.UTF-8
(8 rows)
```

3.1.4.2 What Do I Do If the Character Set Does Not Match the Locale During Database Creation?

If the **LC_COLLATE** you specified does not match the character set during database creation, the following error message is displayed:

```
CREATE DATABASE my_db2 WITH LC_COLLATE ='zh_SG' LC_CTYPE ='zh_SG' ;
```



```
CREATE DATABASE my_db2 WITH LC_COLLATE ='zh_SG' LC_CTYPE ='zh_SG';
```

```
--------------Execute--------------

[Split SQL] Number of SQL(s) to be executed: 1

[Executed SQL: (1)]
CREATE DATABASE my_db2 WITH LC_COLLATE ='zh_SG' LC_CTYPE ='zh_SG';
 execution failed. Reason: ERROR: encoding "UTF8" does not match locale "zh_SG"
  Detail: The chosen LC_CTYPE setting requires encoding "EUC_CN".
```

Solution:

1. Query the character set supported by the template database. For details, see **How Do I View the Created Databases and the Character Sets, LC_COLLATE, and LC_CTYPE Information of the Databases?**. The default template database is template1.
2. Query the **LC_COLLATE** value supported by the character set. For details, see **Configuring the Collation of a Database in a Locale**.
3. Change the value of **LC_COLLATE** to match the character set and create the database again.

3.1.4.3 How Do I Determine a Collation When Migrating Databases from Oracle to RDS for PostgreSQL?

RDS for PostgreSQL servers use en_US.utf8 as the default collation. Chinese Internal Code (GBK) is not supported. When migrating databases from Oracle to RDS for PostgreSQL, you need to set the collation to **C** to meet your expectations.

3.1.4.4 How Do I Create a Database Using a Command-Line Interface (CLI)?

When creating databases, you can specify a template database and set different character sets and collations for each database.

You can connect to your instance using the DAS query window, psql, or pgadmin. This section takes the DAS query window as an example to describe how to create a database using the **CREATE DATABASE** command.

NOTE

Collation information includes LC_COLLATE and LC_CTYPE. For details, see the **official documentation**.

Syntax

```
CREATE DATABASE name
[ [ WITH ] [ OWNER [=] user_name ]
      [ TEMPLATE [=] template ]
      [ ENCODING [=] encoding ]
      [ LC_COLLATE [=] lc_collate ]
      [ LC_CTYPE [=] lc_ctype ]
      [ TABLESPACE [=] tablespace_name ]
      [ ALLOW_CONNECTIONS [=] allowconn ]
      [ CONNECTION LIMIT [=] connlimit ]
      [ IS_TEMPLATE [=] istemplate ] ]
```

Procedure

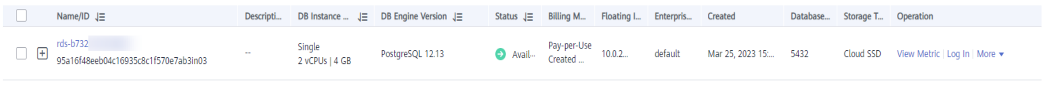
Step 1 **Log in to the management console**.

Step 2 Click in the upper left corner and select a region and a project.

Step 3 Click in the upper left corner of the page and choose **Databases** > **Relational Database Service**.

Step 4 On the **Instances** page, locate the target DB instance and click **Log In** in the **Operation** column.

Figure 3-1 Logging in to an instance



Name/ID	Descripti...	DB Instance ...	DB Engine Version	Status	Billing M...	Floating I...	Enterpris...	Created	Database...	Storage T...	Operation
rds-b732[redacted] 95a16f48eeb04c16935c8c1f570e7ab3in03	--	Single 2 vCPUs | 4 GB	PostgreSQL 12.13	Avail...	Pay-per-Use Created ...	10.0.2...	default	Mar 25, 2023 15:...	5432	Cloud SSD	View Metric | Log In | More ▾

Step 5 On the displayed page, enter the username and password and click **Log In**.

Step 6 Choose **SQL Operations** > **SQL Window**.

Step 7 On the displayed DAS console, choose **SQL Operations** > **SQL Query**. Run the following command to create a database:

```
create database Database name;
```

----End

Parameter Description

- TEMPLATE

  RDS for PostgreSQL has two database templates: **template0** and **template1**. The default template is **template1**. When you use **template1** to create a database, do not specify a new character set for the database. Otherwise, an error will be reported. You can also specify a custom template to create a database.

- ENCODING

  When creating a database, you can specify a character set using **WITH ENCODING**. For details about the supported character sets, see the **official documentation**.

- LC_COLLATE

  String sort order. The default value is **en_US.utf8**.

  Comparison of the same string in different collations may have different results.

  For example, after you execute **SELECT 'a'>'A';**, the result is **false** if this parameter is set to **en_US.utf8** and the result is **true** if this parameter is set to **C**. If you need to migrate a database from Oracle to RDS for PostgreSQL, set **LC_COLLATE** to **C**. You can query the supported collations from the **pg_collation** table.

- LC_CTYPE

  It is used to classify if a character is a digit, uppercase letter, lowercase letter, and so on. You can query the supported character classifications from the **pg_collation** table.

- For details about other parameters, see the **official documentation**.

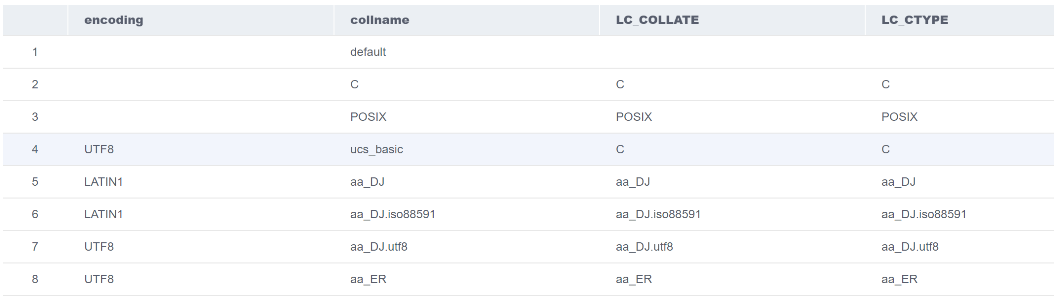
How to Use These Parameters

- Using **TEMPLATE** to specify a database template
  - When template1 is used, the character set or collation defined in this template cannot be changed. For details about collations, see **Configuring the Collation of a Database in a Locale**.
    ```
    CREATE DATABASE my_db WITH TEMPLATE template1 ;
    ```
  - When template0 is used, you can change the character set and collation. For details, see **Configuring the Collation of a Database in a Locale**.
    ```
    CREATE DATABASE my_db WITH ENCODING = 'UTF8' LC_COLLATE ='zh_CN.utf8'
    LC_CTYPE ='zh_CN.utf8' TEMPLATE = template0 ;
    ```
  - If no template is specified during database creation, template1 is used by default. You can also specify a custom template to create a database.
    ```
    CREATE DATABASE my_db WITH TEMPLATE = mytemplate;
    ```
- Using **WITH ENCODING** to specify a character set
  ```
  CREATE DATABASE my_db WITH ENCODING 'UTF8';
  ```

- **LC_COLLATE** and **LC_CTYPE**
  - Querying character sets (encodings) supported by LC_COLLATE and LC_CTYPE

```
SELECT pg_encoding_to_char(collencoding) AS encoding,collname,collcollate AS
"LC_COLLATE",collctype AS "LC_CTYPE" FROM pg_collation;
```

If **encoding** is empty, **LC_COLLATE** supports all character sets.

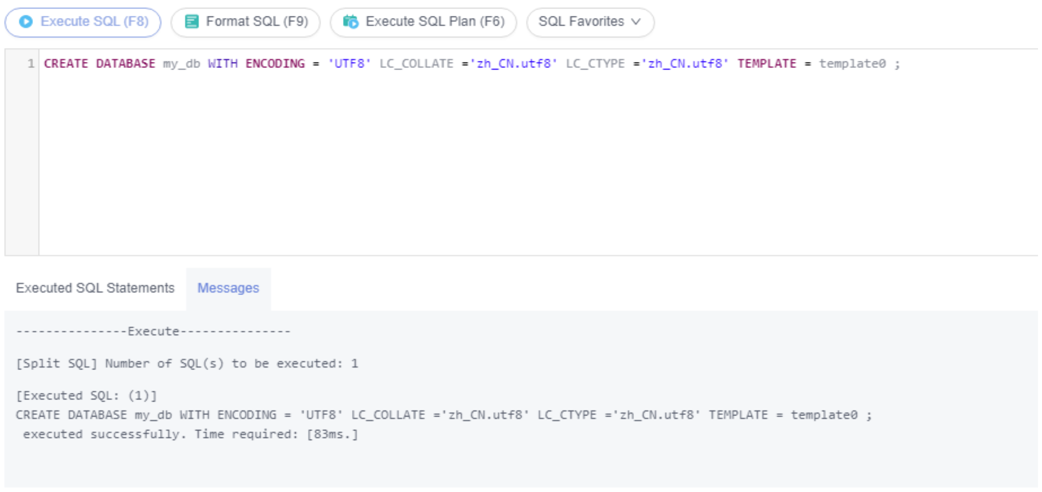


	encoding	collname	LC_COLLATE	LC_CTYPE
1		default		
2		C	C	C
3		POSIX	POSIX	POSIX
4	UTF8	ucs_basic	C	C
5	LATIN1	aa_DJ	aa_DJ	aa_DJ
6	LATIN1	aa_DJ.iso88591	aa_DJ.iso88591	aa_DJ.iso88591
7	UTF8	aa_DJ.utf8	aa_DJ.utf8	aa_DJ.utf8
8	UTF8	aa_ER	aa_ER	aa_ER

  - Configuring the collation of a database in a locale

Run the following command to create a database with **LC_COLLATE** and **LC_CTYPE** set to **zh_CN.utf8**:

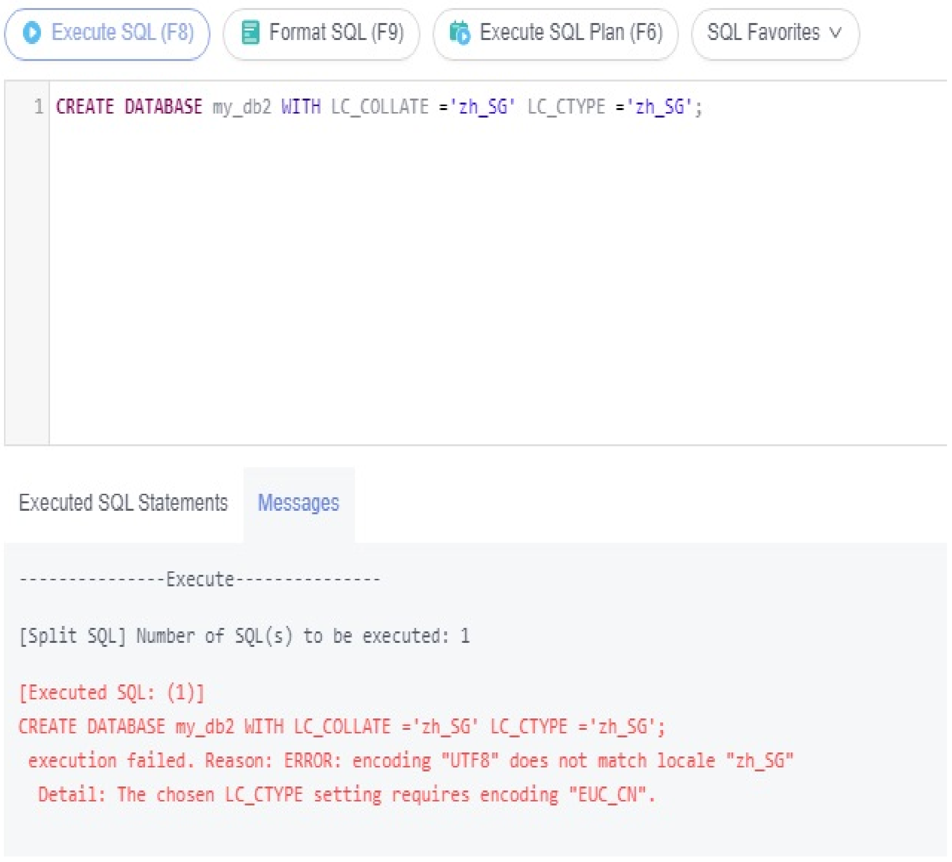
```
CREATE DATABASE my_db WITH ENCODING = 'UTF8' LC_COLLATE ='zh_CN.utf8'
LC_CTYPE ='zh_CN.utf8' TEMPLATE = template0 ;
```



```
CREATE DATABASE my_db WITH ENCODING = 'UTF8' LC_COLLATE ='zh_CN.utf8' LC_CTYPE ='zh_CN.utf8' TEMPLATE = template0 ;
```

```
--------------Execute--------------

[Split SQL] Number of SQL(s) to be executed: 1

[Executed SQL: (1)]
CREATE DATABASE my_db WITH ENCODING = 'UTF8' LC_COLLATE ='zh_CN.utf8' LC_CTYPE ='zh_CN.utf8' TEMPLATE = template0 ;
 executed successfully. Time required: [83ms.]
```

If the specified **LC_COLLATE** is incompatible with the character set, error information similar to the following is displayed:



```
CREATE DATABASE my_db2 WITH LC_COLLATE ='zh_SG' LC_CTYPE ='zh_SG';
```

```
---------------Execute---------------

[Split SQL] Number of SQL(s) to be executed: 1

[Executed SQL: (1)]
CREATE DATABASE my_db2 WITH LC_COLLATE ='zh_SG' LC_CTYPE ='zh_SG';
 execution failed. Reason: ERROR: encoding "UTF8" does not match locale "zh_SG"
  Detail: The chosen LC_CTYPE setting requires encoding "EUC_CN".
```

NOTE

1. The specified **LC_COLLATE** and **LC_CTYPE** must be compatible with the target character set. Otherwise, an error is reported. For details, see **Querying LC_COLLATE and LC_CTYPE Settings Supported by a Character Set**.
2. The **LC_COLLATE** and **LC_CTYPE** settings of an existing database cannot be changed by running the **ALTER DATABASE** statement. You can change them while creating a new database and then import your data to the new database.

3.1.4.5 What Are the Constraints on Character Sets?

When creating a database, you can specify a character set using **WITH ENCODING**. For details about the supported character sets, see the **official documentation**.

Constraints

When template1 is used to create a database, the character set defined for this template cannot be changed. Otherwise, an error is reported. For details, see **TEMPLATE Option**.

Example

```
CREATE DATABASE my_db WITH ENCODING 'UTF8';
```

3.2 Performing Basic Operations Using pgAdmin

3.2.1 Connecting to an RDS for PostgreSQL Instance

pgAdmin is client management software that designs, maintains, and manages RDS for PostgreSQL databases. It allows you to connect to specific databases, create tables, and run various simple and complex SQL statements. It supports Windows, Linux, macOS, and other operating systems. The latest version of pgAdmin is based on the browser/server (B/S) architecture.

This section describes how to use pgAdmin to connect to an RDS for PostgreSQL instance and create databases and tables. pgAdmin4-4.17 is used as an example in this section.

For more information, see the **pgAdmin documentation**.

Procedure

Step 1 Obtain the pgAdmin installation package.

Visit the **pgAdmin official website** and download the pgAdmin installation package for Windows.

Step 2 Double-click the installation package and complete the installation as instructed.

Step 3 Start the pgAdmin client after the installation.

Step 4 Connect pgAdmin to your RDS for PostgreSQL instance by referring to **Connecting to a DB Instance Through pgAdmin**.

----End

3.2.2 Creating a Database

Scenarios

This section describes how to create a database.

Prerequisites

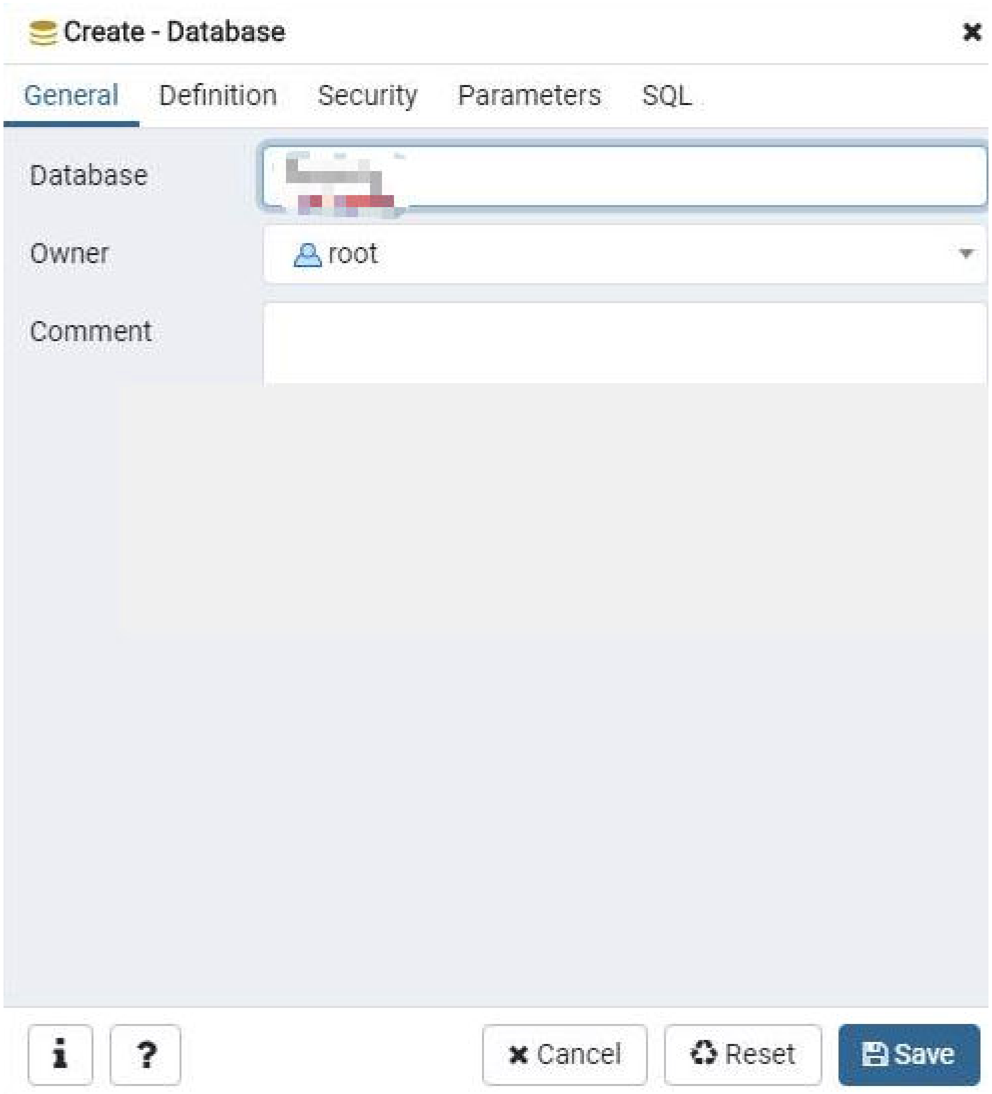
An RDS for PostgreSQL DB instance has been connected.

Procedure

Step 1 In the navigation pane on the left, right-click the target instance node and choose **Create** > **Database** from the shortcut menu.

Step 2 On the **General** tab, specify **Database** and click **Save**.

Figure 3-2 Creating a database



Database	
Owner	root
Comment	

----End

3.2.3 Creating a Table

Scenarios

This section describes how to create a table.

Prerequisites

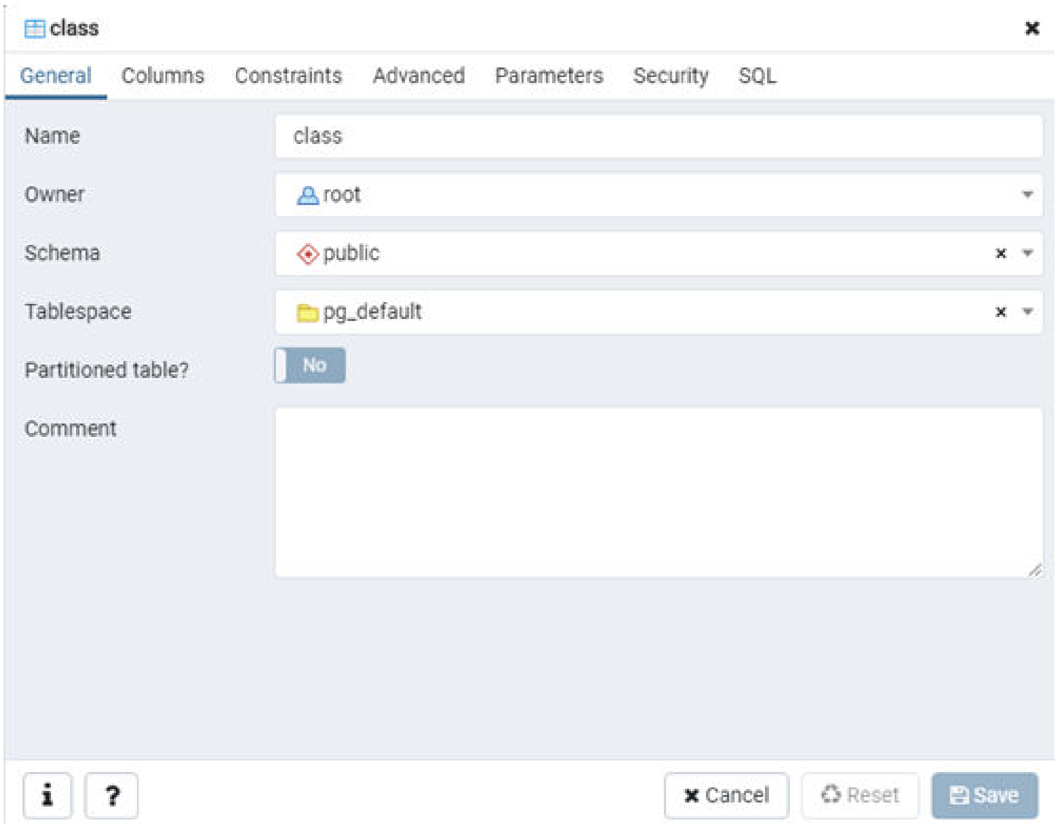
- An RDS for PostgreSQL DB instance has been connected.
- There are tables in the database and schema created by the current user.

Procedure

Step 1 In the navigation pane on the left, right-click the target table and choose **Create** > **Table** from the shortcut menu.

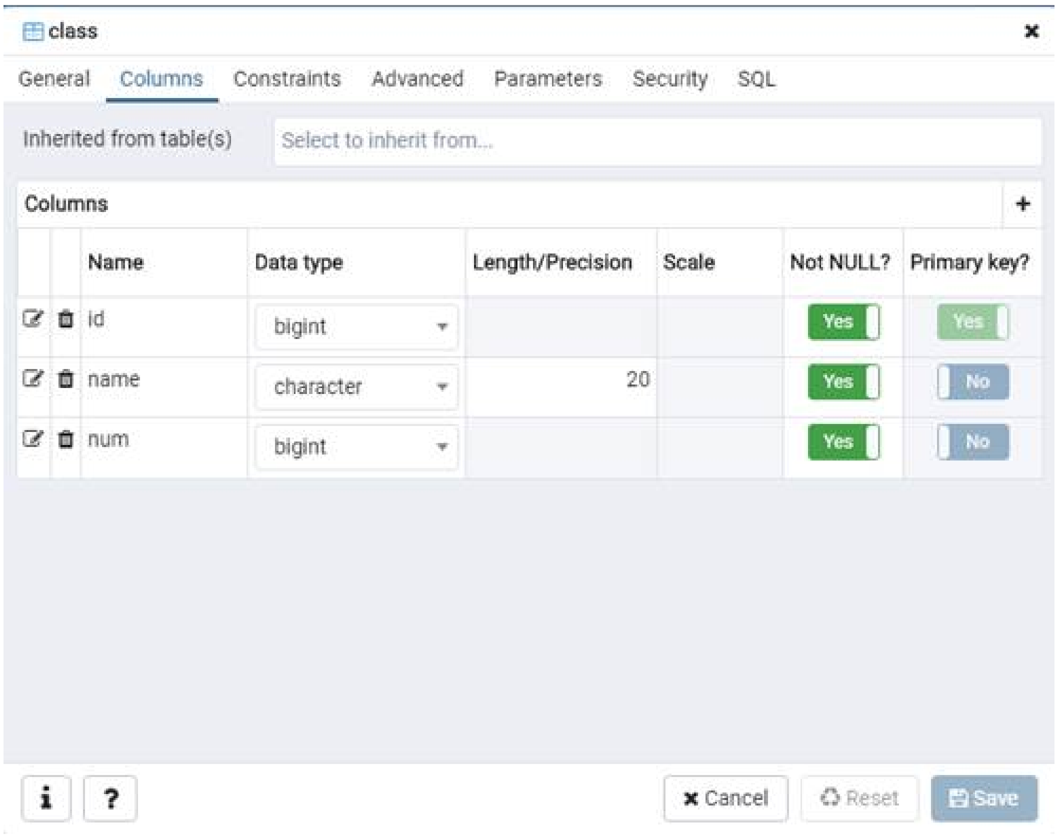
Step 2 On the **General** tab, enter required information and click **Save**.

Figure 3-3 Basic information



Name	class
Owner	root
Schema	public
Tablespace	pg_default
Partitioned table?	No
Comment	

Step 3 On the **Columns** tab, add table columns and click **Save**.



Inherited from table(s): Select to inherit from...

Columns

Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?
id	bigint			Yes	Yes
name	character	20		Yes	No
num	bigint			Yes	No

----End

3.2.4 Using the Query Tool to Run SQL Statements

Scenarios

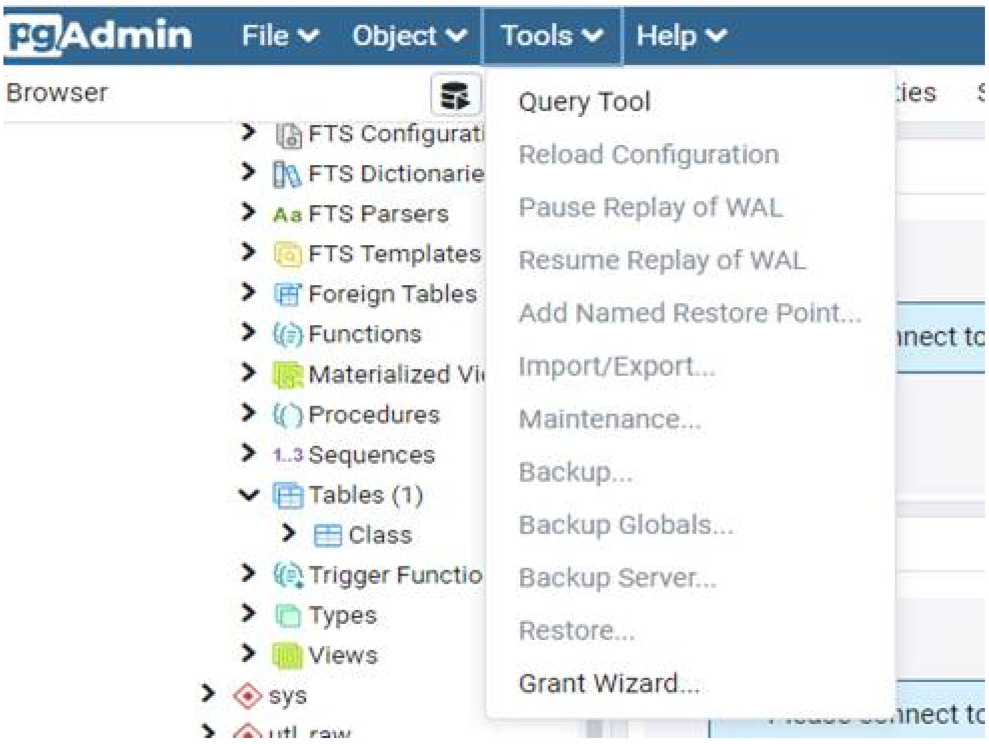
This topic describes how to use Query Tool to run SQL commands.

Prerequisites

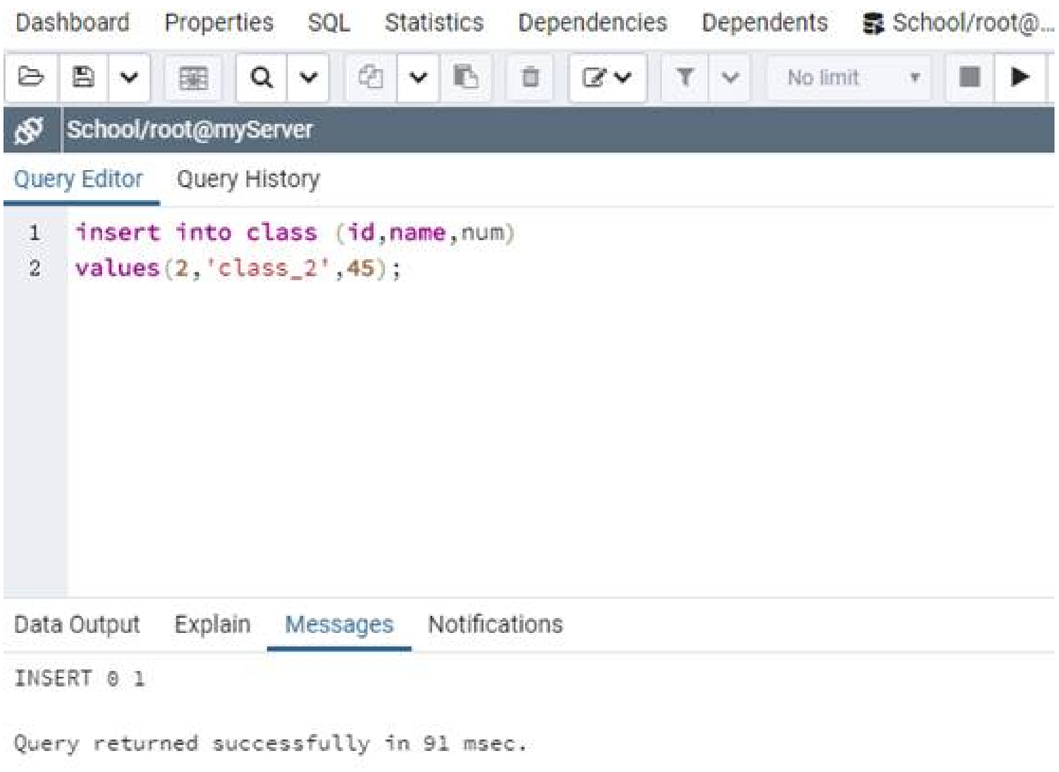
An RDS for PostgreSQL DB instance has been connected.

Inserting Data

Step 1 On the top menu bar, choose **Tools** > **Query Tool**. The SQL CLI is displayed.



Step 2 On the SQL CLI, enter an **INSERT** command and click **Execute** to insert data into the corresponding table.



```
insert into class (id,name,num)
values(2,'class_2',45);
```

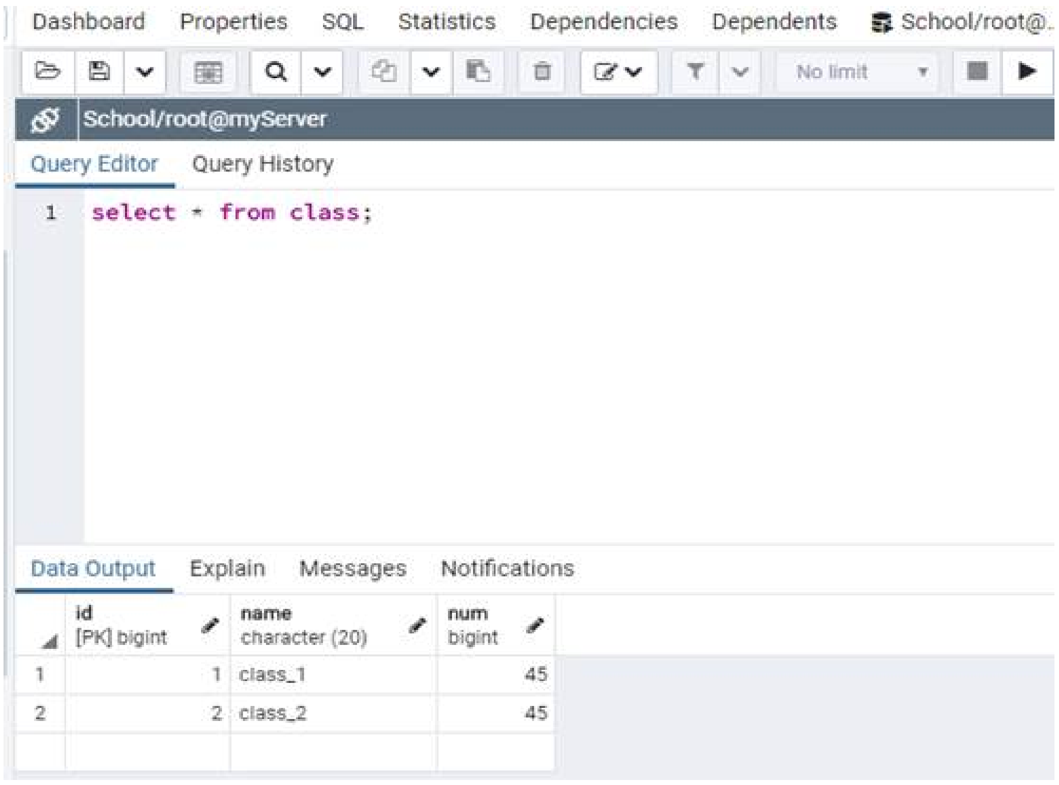
```
INSERT 0 1

Query returned successfully in 91 msec.
```

----End

Querying Data

Step 1 On the top menu bar, choose **Tools** > **Query Tool**. The SQL CLI is displayed.

Step 2 On the SQL CLI, enter an **SELECT** command and click **Execute** to query data in the corresponding table.



```
select * from class;
```

	id [PK] bigint	name character (20)	num bigint
1	1	class_1	45
2	2	class_2	45

----End

3.2.5 Monitoring Databases

Scenarios

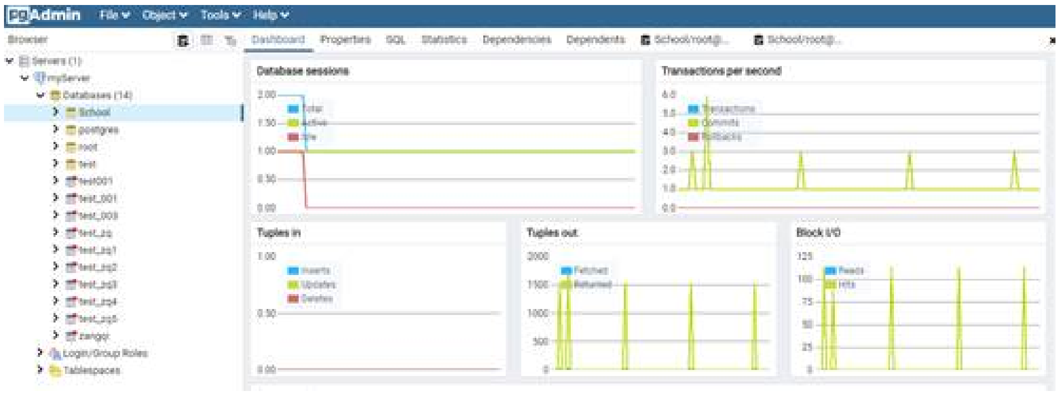
This section describes database monitoring.

Prerequisites

An RDS for PostgreSQL DB instance has been connected.

pgAdmin Monitoring

Step 1 In the navigation pane on the left, select a database and click the **Dashboard** tab on the right pane to monitor database metrics, including Database sessions, Transactions per second, Tuples in, Tuples out, and Block I/O.



----End

RDS Monitoring

The Cloud Eye service monitors operating statuses of RDS DB instances. You can view RDS database metrics on the management console. For more information, see **Viewing Monitoring Metrics**.

3.2.6 Backing Up and Restoring Data

Scenarios

This section describes how to back up and restore data.

Prerequisites

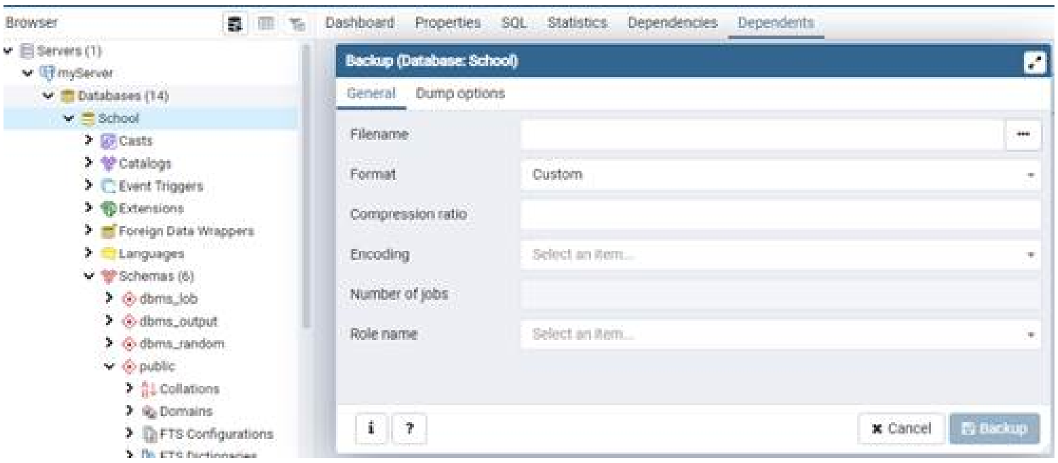
An RDS for PostgreSQL DB instance has been connected.

pgAdmin Backup and Restoration

Backup

Step 1 In the navigation pane on the left, right-click the database to be backed up and choose **Backup** from the shortcut menu.

Step 2 On the **General** tab, enter required information and click **Backup**.



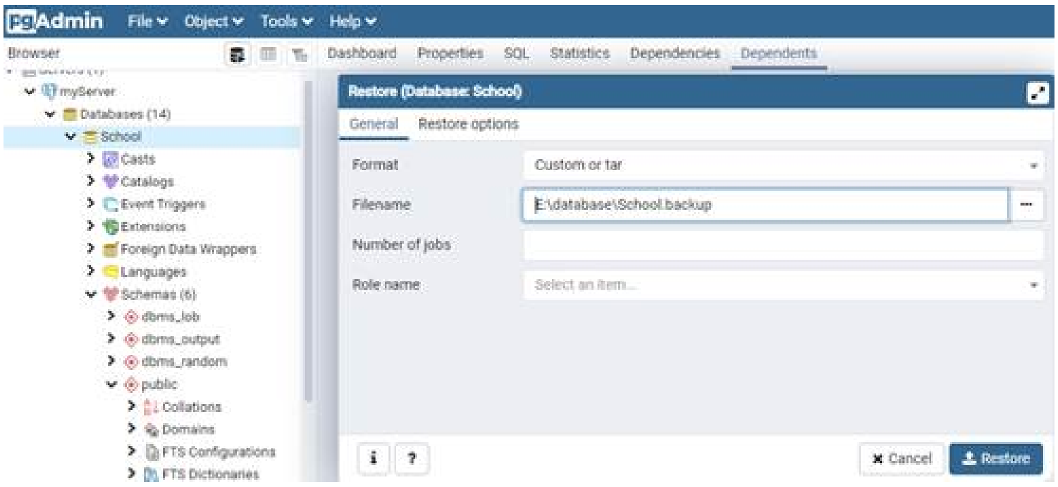
Filename:
Format: Custom
Compression ratio:
Encoding: Select an item...
Number of jobs:
Role name: Select an item...

----End

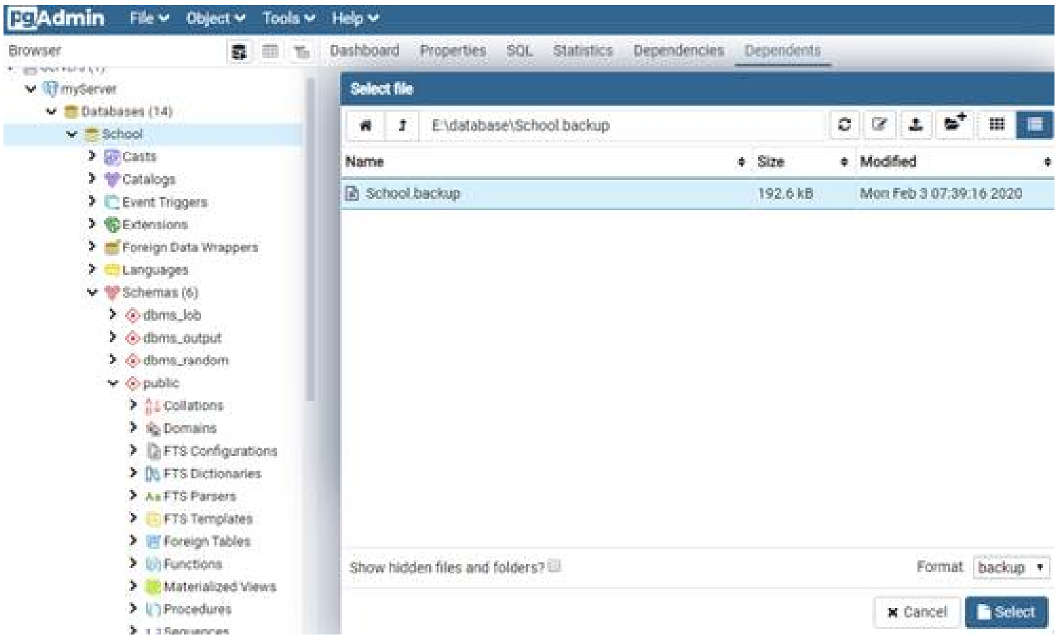
Restoration

Step 1 In the navigation pane on the left, right-click the database to be restored and choose **Restore** from the shortcut menu.

Step 2 On the displayed tab in the right pane, select a file name and click **Restore**.



Format: Custom or tar
Filename: E:\database\School.backup
Number of jobs:
Role name: Select an item...



E:\database\School.backup

Name	Size	Modified
School.backup	192.6 kB	Mon Feb 3 07:39:16 2020

Format: backup

----End

RDS Backup and Restoration

RDS supports DB instance backup and restoration to ensure data reliability. For more information, see **Data Backups**.

3.3 Using PoWA

3.3.1 Overview

PoWA is an open-source system used to monitor the performance of RDS for PostgreSQL databases. It consists of the PoWA-archivist, PoWA-collector, and PoWA-web components and obtains performance data through other plug-ins installed in the RDS for PostgreSQL databases. The key components are as follows:

- PoWA-archivist: the PostgreSQL plug-in for collecting performance data obtained by other plug-ins.
- PoWA-collector: the daemon that gathers performance metrics from remote PostgreSQL instances on a dedicated repository server.
- PoWA-web: the web-based user interface displaying performance metrics collected by the PoWA-collector.
- Other plug-ins: the sources of performance metric data. They are installed on the destination PostgreSQL database.
- PoWA: the system name.

Security Risk Warning

The following security risks may exist during PoWA deployment and configuration.

- (Remote mode) When configuring instance performance metric information to be collected in powa-repository, you need to enter the IP address, **root** username, and connection password of the destination instance. You can query related information in the **powa_servers** table. The connection password is displayed in plaintext.

- In the **PoWA-collector** configuration file, the powa-repository connection information does not contain the connection password. It means that the powa-repository connection configuration item for PoWA-collector must be trust.
- In the **PoWA-web** configuration file, the **root** user and connection password of powa-repository (remote mode) or DB instance (local mode) are optional and stored in plaintext.

Before using the PoWA, you need to be aware of the preceding security risks. For details about how to harden security, see **PoWA official document**.

3.3.2 Performance Metrics

Using PoWA has a minor negative impact on PostgreSQL server performance, and it is difficult to accurately assess this impact because it may come from different parts.

You need to activate at least the pg_stat_statements plug-in and other supported Stats plug-ins (if any). These plug-ins may slow down your instances and you are advised to choose an appropriate method to configure them.

If you do not use the remote mode, the data is periodically stored locally. Overhead varies with snapshot frequency and the disk usage affects backups.

3.3.2.1 Database Performance Metrics

General Overview

Figure 3-4 General Overview

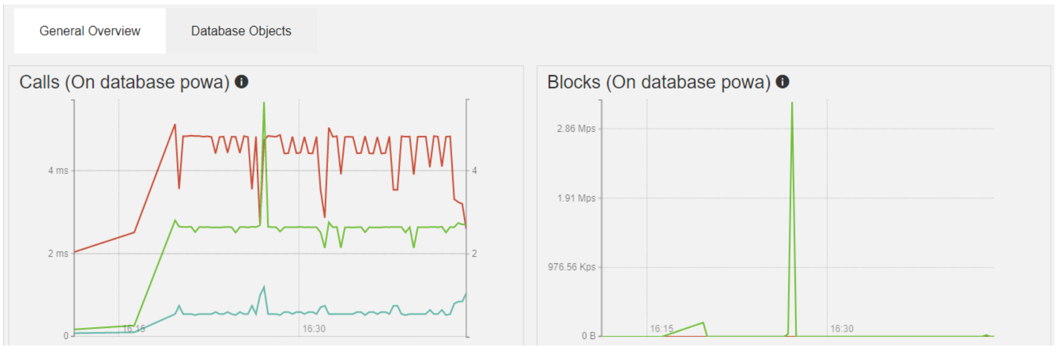


Table 3-1 Calls field description

Field	Description
Queries per sec	Number of queries executed per second
Runtime per sec	Total duration of queries executed per second
Avg runtime	Average query duration

Table 3-2 Blocks field description

Field	Description
Total shared buffers hit	Amount of data found in shared buffers
Total shared buffers miss	Amount of data found in OS cache or read from disk

Database Objects

Figure 3-5 Database Objects

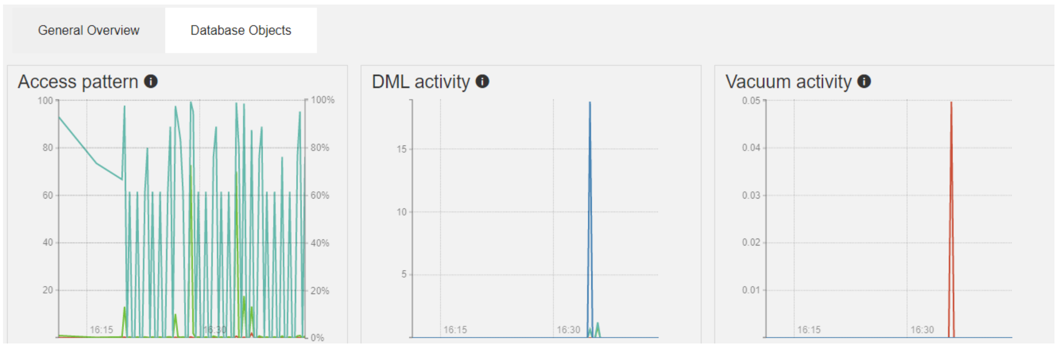


Table 3-3 Access pattern field description

Field	Description
Index scans ratio	Ratio of index scans to sequential scans
Index scans	Number of index scans per second
Sequential scans	Number of sequential scans per second

Table 3-4 DML activity field description

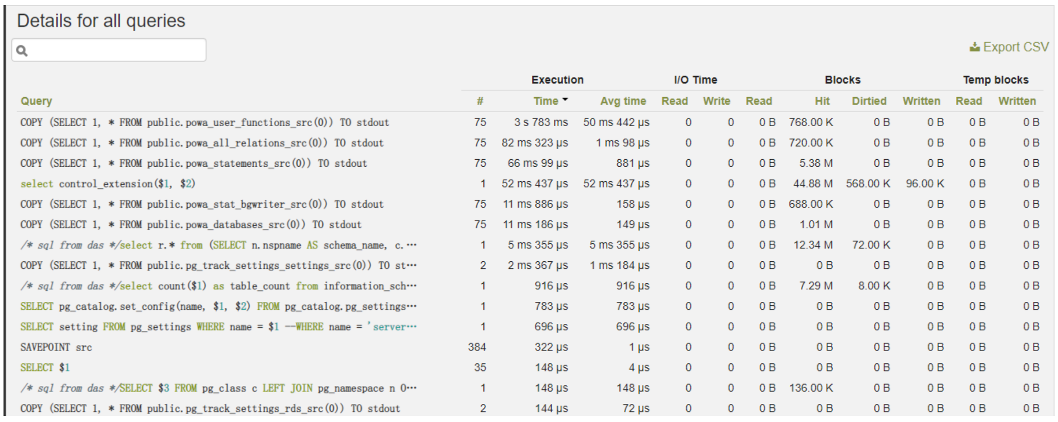
Field	Description
Tuples inserted	Number of tuples inserted per second
Tuples updated	Number of tuples updated per second
Tuples HOT updated	Number of heap-only tuples (HOT) updated per second
Tuples deleted	Number of tuples deleted per second

Table 3-5 Vacuum activity field description

Field	Description
# Vacuum	Number of vacuums per second
# Autovacuum	Number of autovacuums per second
# Analyze	Number of analyses per second
# Autoanalyze	Number of autoanalyses per second

Details for all databases

Figure 3-6 Details for all databases


Details for all queries

	Execution			I/O Time		Blocks				Temp blocks	
Query	#	Time	Avg time	Read	Write	Read	Hit	Dirtied	Written	Read	Written
COPY (SELECT 1, * FROM public.powa_user_functions_src(0)) TO stdout	75	3 s 783 ms	50 ms 442 µs	0	0	0 B	768.00 K	0 B	0 B	0 B	0 B
COPY (SELECT 1, * FROM public.powa_all_relations_src(0)) TO stdout	75	82 ms 323 µs	1 ms 98 µs	0	0	0 B	720.00 K	0 B	0 B	0 B	0 B
COPY (SELECT 1, * FROM public.powa_statements_src(0)) TO stdout	75	66 ms 99 µs	881 µs	0	0	0 B	5.38 M	0 B	0 B	0 B	0 B
select control_extension($1, $2)	1	52 ms 437 µs	52 ms 437 µs	0	0	0 B	44.88 M	568.00 K	96.00 K	0 B	0 B
COPY (SELECT 1, * FROM public.powa_stat_bgwriter_src(0)) TO stdout	75	11 ms 886 µs	158 µs	0	0	0 B	688.00 K	0 B	0 B	0 B	0 B
COPY (SELECT 1, * FROM public.powa_databases_src(0)) TO stdout	75	11 ms 186 µs	149 µs	0	0	0 B	1.01 M	0 B	0 B	0 B	0 B
/* sql from das */select r.* from (SELECT n.nspname AS schema_name, c.···	1	5 ms 355 µs	5 ms 355 µs	0	0	0 B	12.34 M	72.00 K	0 B	0 B	0 B
COPY (SELECT 1, * FROM public.pg_track_settings_settings_src(0)) TO st···	2	2 ms 367 µs	1 ms 184 µs	0	0	0 B	0 B	0 B	0 B	0 B	0 B
/* sql from das */select count($1) as table_count from information_sch···	1	916 µs	916 µs	0	0	0 B	7.29 M	8.00 K	0 B	0 B	0 B
SELECT pg_catalog.set_config(name, $1, $2) FROM pg_catalog.pg_settings···	1	783 µs	783 µs	0	0	0 B	0 B	0 B	0 B	0 B	0 B
SELECT setting FROM pg_settings WHERE name = $1 --WHERE name = 'server···	1	696 µs	696 µs	0	0	0 B	0 B	0 B	0 B	0 B	0 B
SAVEPOINT src	384	322 µs	1 µs	0	0	0 B	0 B	0 B	0 B	0 B	0 B
SELECT $1	35	148 µs	4 µs	0	0	0 B	0 B	0 B	0 B	0 B	0 B
/* sql from das */SELECT $3 FROM pg_class c LEFT JOIN pg_namespace n O···	1	148 µs	148 µs	0	0	0 B	136.00 K	0 B	0 B	0 B	0 B
COPY (SELECT 1, * FROM public.pg_track_settings_rds_src(0)) TO stdout	2	144 µs	72 µs	0	0	0 B	0 B	0 B	0 B	0 B	0 B

Table 3-6 Details for all databases field description

Field	Description
Query	SQL statement to be executed
(Execution) #	Number of times that the SQL statement is executed
(Execution) Time	Total execution time of the SQL statement
(Execution) Avg time	Average time for executing the SQL statement
(I/O Time) Read	Read I/O wait time
(I/O Time) Write	Write I/O wait time
(Blocks) Read	Number of disk read pages
(Blocks) Hit	Number of hit pages in the shared buffer
(Blocks) Dirtied	Number of dirty pages
(Blocks) Written	Number of disk write pages
(Temp blocks) Read	Number of disk temporary read pages

Field	Description
(Temp blocks) Write	Number of disk temporary write pages

3.3.2.2 Instance Performance Metrics

General Overview

Figure 3-7 General Overview metrics

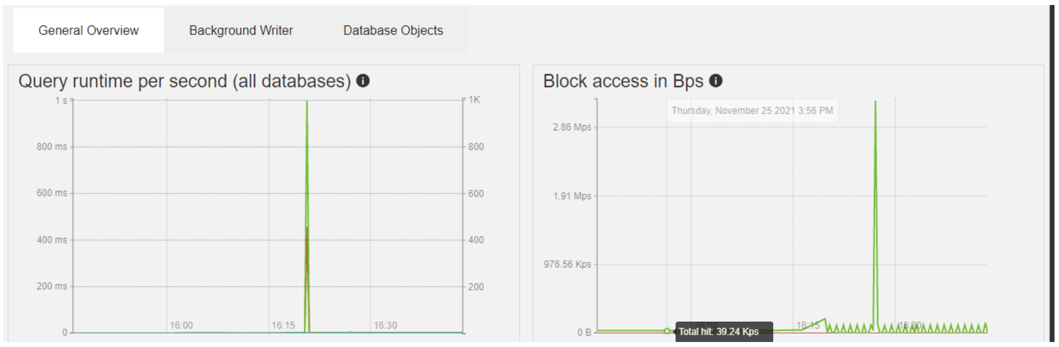


Table 3-7 Query runtime per second (all databases) field description

Field	Description
Queries per sec	Number of queries executed per second
Runtime per sec	Total duration of queries executed per second
Avg runtime	Average query duration

Table 3-8 Block access in Bps field description

Field	Description
Total hit	Amount of data found in shared buffers
Total read	Amount of data found in OS cache or read from disk

Background Writer

Figure 3-8 Background Writer metrics

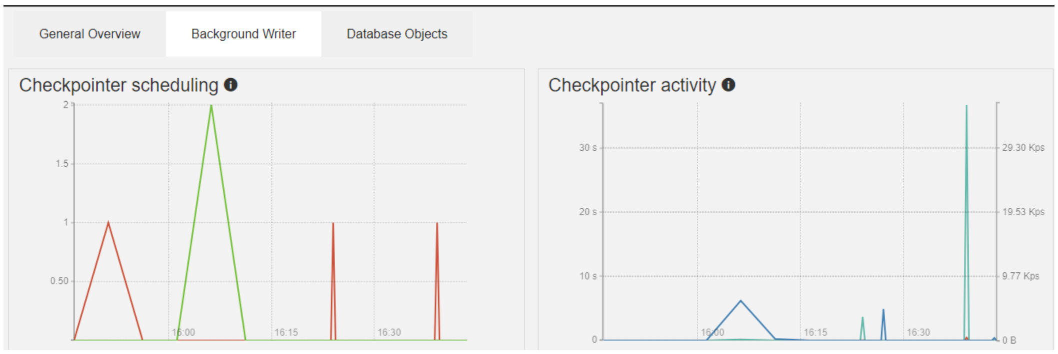


Table 3-9 Checkpointer scheduling field description

Field	Description
of requested checkpoints	Number of requested checkpoints that have been performed
of scheduled checkpoints	Number of scheduled checkpoints that have been performed

Table 3-10 Checkpointer activity field description

Field	Description
Buffers alloc	Number of buffers allocated
Sync time	Total amount of time that has been spent in the portion of checkpoint processing where files are synchronized to disk, in milliseconds
Write time	Total amount of time that has been spent in the portion of checkpoint processing where files are written to disk, in milliseconds

Table 3-11 Background writer field description

Field	Description
Maxwritten clean	Number of times the background writer stopped a cleaning scan because it had written too many buffers
Buffers clean	Number of buffers written by the background writer

Table 3-12 Backends field description

Field	Description
Buffers backend fsync	Number of times a backend had to execute its own fsync call (normally the background writer handles those even when the backend does its own write)
Buffers backend	Number of buffers written directly by a backend

Database Objects

Figure 3-9 Database Objects metrics

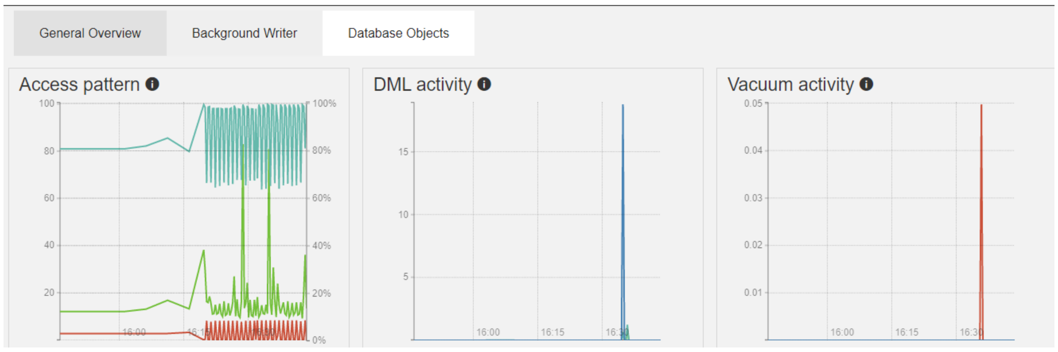


Table 3-13 Access pattern field description

Field	Description
Index scans ratio	Ratio of index scans to sequential scans
Index scans	Number of index scans per second
Sequential scans	Number of sequential scans per second

Table 3-14 DML activity field description

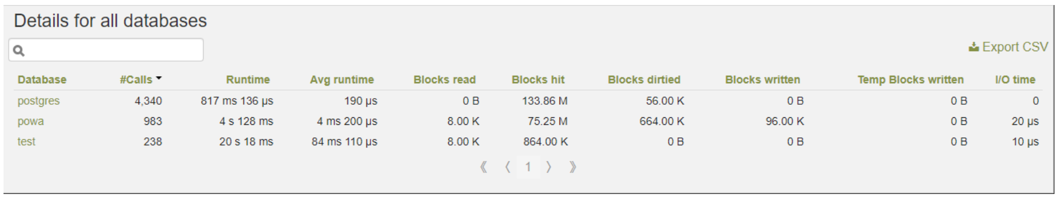
Field	Description
Tuples inserted	Number of tuples inserted per second
Tuples updated	Number of tuples updated per second
Tuples HOT updated	Number of heap-only tuples (HOT) updated per second
Tuples deleted	Number of tuples deleted per second

Table 3-15 Vacuum activity field description

Field	Description
# Vacuum	Number of vacuums per second
# Autovacuum	Number of autovacuums per second
# Analyze	Number of analyses per second
# Autoanalyze	Number of autoanalyses per second

Details for all databases

Figure 3-10 Details for all databases metrics



Details for all databases

Database	#Calls	Runtime	Avg runtime	Blocks read	Blocks hit	Blocks dirtied	Blocks written	Temp Blocks written	I/O time
postgres	4,340	817 ms 136 μs	190 μs	0 B	133.86 M	56.00 K	0 B	0 B	0
powa	983	4 s 128 ms	4 ms 200 μs	8.00 K	75.25 M	664.00 K	96.00 K	0 B	20 μs
test	238	20 s 18 ms	84 ms 110 μs	8.00 K	864.00 K	0 B	0 B	0 B	10 μs

Table 3-16

Field	Description
Database	Database name
#Calls	Total number of executed SQL statements
Runtime	Total runtime of the SQL statement
Avg runtime	Average runtime of the SQL statement
Blocks read	Number of pages read from the disk
Blocks hit	Number of hit pages in the shared buffer
Blocks dirtied	Number of dirty pages
Blocks written	Number of disk write pages
Temp Blocks written	Number of disk temporary write pages
I/O time	I/O wait time

3.3.3 PoWA Deployment Models

PoWA supports local and remote deployment. For details, see **Local Deployment** and **Remote Deployment**. For security purposes, RDS for PostgreSQL supports only remote deployment of PoWA.

Local Deployment

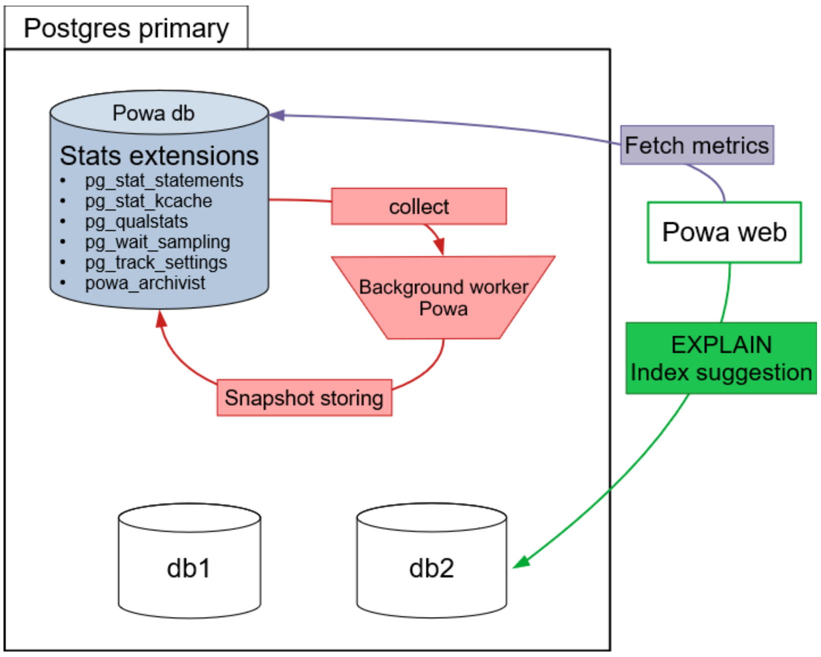
The architecture is simple and does not require the participation of the PoWA-collector. It consists of only the PoWA-archivist, PoWA-web, and other plug-ins. However, this architecture has the following disadvantages:

- It increases performance costs when collecting data and using the user interface.
- Collected performance metric data is stored on the local host, which increases the disk space usage.
- Data cannot be collected on the hot standby server.

NOTICE

For security purposes, RDS for PostgreSQL does not support local deployment.

Figure 3-11 Local deployment architecture



Remote Deployment

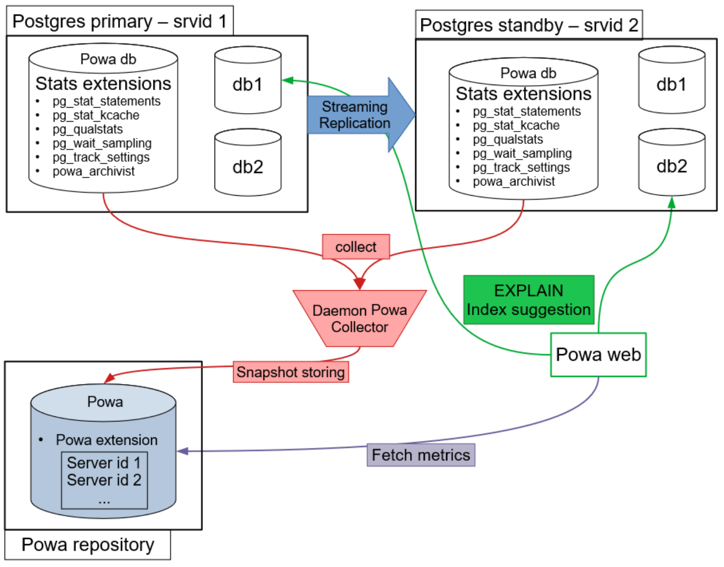
PoWA 4.0 allows you to store the data of one or multiples servers on an external PostgreSQL database. Currently, RDS for PostgreSQL supports PoWA 4.1.2.

Compared with local deployment, remote deployment has a dedicated **Powa repository** database for storing performance metric data collected by PoWA-collector.

RDS for PostgreSQL supports remote deployment.

The remote deployment architecture is as follows.

Figure 3-12 Remote deployment architecture



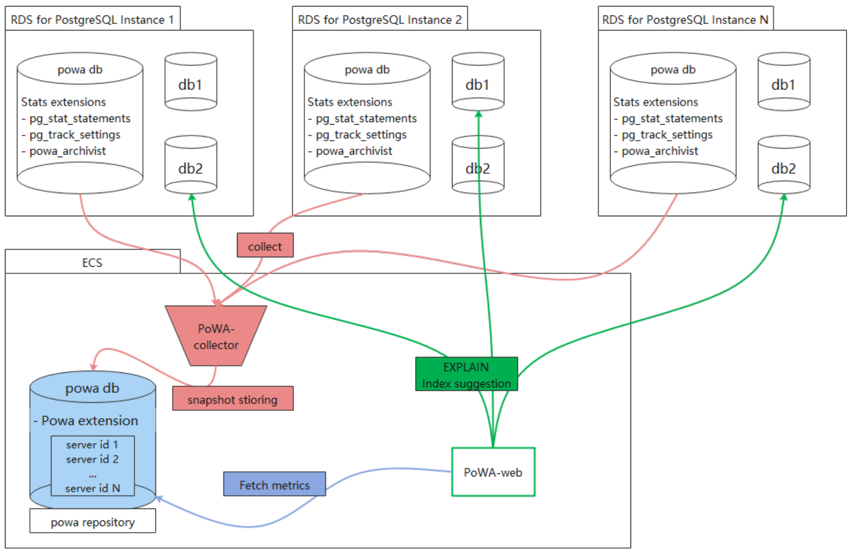
3.3.3.1 Introducing Remote Deployment

An ECS is required for remote deployment on Huawei Cloud. The RDS for PostgreSQL database, PoWA-archivist, PoWA-collector, and PoWA-web need to be installed on the ECS.

This section describes how to install the PoWA-archivist, PoWA-collector, and PoWA-web.

The following figure shows the remote deployment architecture.

Figure 3-13 Remote deployment architecture



Preparing an ECS

Create an ECS on Huawei Cloud and configure an EIP for it. The ECS configured with the CentOS 8.2 64-bit image is used as an example. PostgreSQL 12.6 has been installed on the ECS.

Installing Python3

PoWA-collector and PoWA-web must be installed in a Python3 environment. You can use pip3 to install them to facilitate the installation. In the example, Python 3.6.8 is installed on the ECS by default. The latest PoWA version fails to be installed. For details about how to install the latest version, see **Installing Python 3.9.9**.

Installing PoWA-archivist

1. After PostgreSQL 12.6 is installed, run the **wget** command to obtain the PoWA-archivist source code.

   wget https://github.com/powa-team/powa-archivist/archive/refs/tags/REL_4_1_2.tar.gz

2. Decompress the downloaded **REL_4_1_2.tar.gz** package.
3. Install PoWA-archivist to the decompressed directory.

   make && make install

Installing PoWA-collector and PoWA-web

1. Switch to the RDS for PostgreSQL database installation user. Take user **postgres** as an example.

   su - postgres

2. Install PoWA-collector and PoWA-web. psycopg2 is mandatory for installing them.

   pip install psycopg2

   pip install powa-collector

   pip install powa-web

After the installation is complete, check the following path tree. If the following information is displayed, the PoWA-collector and PoWA-web have been installed.

```
/home/postgres/.local/bin
├── powa-collector.py
├── powa-web
└── __pycache__
```

FAQs

Q: What should I do if the error message "python setup.py build_ext --pg-config /path/to/pg_config build" is displayed when the **pip install psycopg2** command is executed?

A: You need to add the **bin** and **lib** paths of RDS for PostgreSQL to environment variables and run the **pip install psycopg2** command.

3.3.3.2 Configuring Remote Deployment

Configuring a DB Instance for Performance Metric Collection

Step 1 Log in to the powa database of the target DB instance as user **root**. (If the powa database does not exist, create it first.)

Step 2 Create the powa plug-in in the powa database.

```
select control_extension('create', 'pg_stat_statements');
select control_extension('create', 'btree_gist');
select control_extension('create', 'powa');
```

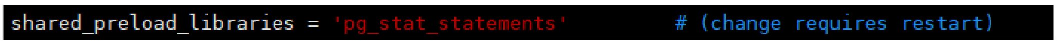
----End

Configuring Local PostgreSQL

Take the PostgreSQL (powa-repository) on the ECS as an example.

- Version: PostgreSQL 12.6
- superuser: postgres
- Data path: /**home/postgres/data**

Step 1 Add **pg_stat_statements** to **shared_preload_libraries** in the **/home/postgres/data/postgresql.conf** file.



```
shared_preload_libraries = 'pg_stat_statements'          # (change requires restart)
```

Step 2 Restart the database.

pg_ctl restart -D /home/postgres/data/

Step 3 Log in to the database as the **super** user, create database **powa**, and install related plug-ins.

NOTICE

The created database must be named **powa**. Otherwise, an error is reported and certain functions do not take effect while PoWA is running.

```
[postgres@ecs-ad4d ~]$ psql -U postgres -d postgres
psql (12.6)
Type "help" for help.
postgres=# create database powa;
CREATE DATABASE
postgres=# \c powa
You are now connected to database "powa" as user "postgres".
powa=# create extension pg_stat_statements ;
CREATE EXTENSION
powa=# create extension btree_gist ;
CREATE EXTENSION
powa=# create extension powa;
CREATE EXTENSION
```

Step 4 Configure the instance whose performance metrics need to be collected.

1. Add the instance information.

```
powa=# select powa_register_server(
    hostname => '192.168.0.1',
    alias => 'myInstance',
    port => 5432,
    username => 'user1',
    password => '**********',
    frequency => 300);
     powa_register_server
    ----------------------
     t
    (1 row)
```

2. View the **powa_servers** table to obtain the performance metric information of the instance.

```
powa=# select * from powa_servers;
id | hostname | alias | port | username |  password  | dbname | frequency | powa_coalesce | retention |
allow_ui_connection |version
----+---------------+------------+------+----------+------------+--------+-----------+---------------+-----------
+------------------
0 |           | <local>|  0 |          |            |        |        -1 |   100        | 00:00:00  | t           |
1 | 192.168.0.1 | myInstance | user1 | 5432     | ********** | powa    |       300 |           100 | 1 day       |
t             |
(2 rows)
```

NOTICE

Information about the destination instance information includes important privacy information, such as its IP address, root user account, and plaintext password.

Assess the plug-in security risks before you decide to use them.

----End

Configuring PoWA-collector

When PoWA-collector starts, it searches for configuration files in the following sequence:

1. /etc/powa-collector.conf
2. ~/.config/powa-collector.conf
3. ~/.powa-collector.conf
4. ./powa-collector.conf

The configuration files must contain the following options:

- repository.dsn: URL. It is used to notify the powa-collector of how to connect to the dedicated storage database (powa-repository).
- debug: Boolean type. It specifies whether to enable the powa-collector in debugging mode.

Take the configuration file **./powa-collector.conf** as an example.

```
{
    "repository": {
    "dsn": "postgresql://postgres@localhost:5432/powa"
    },
    "debug": true
}
```

No password is configured in the PoWA-collector configuration. Therefore, you need to set the connection policy in the **pg_hba.conf** file of the **powa-repository** database to **trust** (password-free connection).

Start the powa-collector.

cd /home/postgres/.local/bin

./powa-collector.py &

Configuring PoWA-Web

When PoWA-collector starts, it searches for configuration files in the following sequence:

1. /etc/powa-web.conf
2. ~/.config/powa-web.conf
3. ~/.powa-web.conf
4. ./powa-web.conf

Take the configuration file **./powa-web.conf** as an example.

```
# cd /home/postgres/.local/bin
# vim ./powa-web.conf
# Write the configuration information and save it.
servers={
      'main': {
        'host': 'localhost',
        'port': '5432',
        'database': 'powa',
        'username': 'postgres',
        'query': {'client_encoding': 'utf8'}
      }
}
cookie_secret="SECRET_STRING"
```

In this example, the connection policy in the **pg_hab.conf** file of the powa-repository database is set to **trust** (password-free connection). Therefore, the password is not configured.

Start the powa-web.

cd /home/postgres/.local/bin

./powa-web &

3.3.4 Accessing PoWA

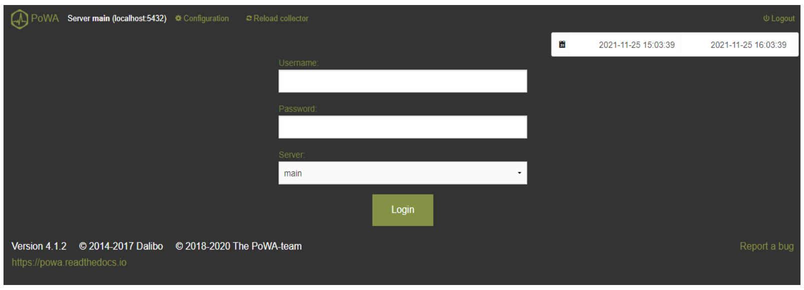
After the remote deployment is configured, and PoWA-collector and PoWA-web are started, you can view the monitoring metrics of the instance through a browser. In the example,

# Port is not configured in the **powa-web.conf** file. The default value **8888** is used.

# Browser access website: **http://***ECS_IP_address***:8888/**

Step 1 Use a browser to access PoWA.

Figure 3-14 Access PoWA.



Server main (localhost:5432)

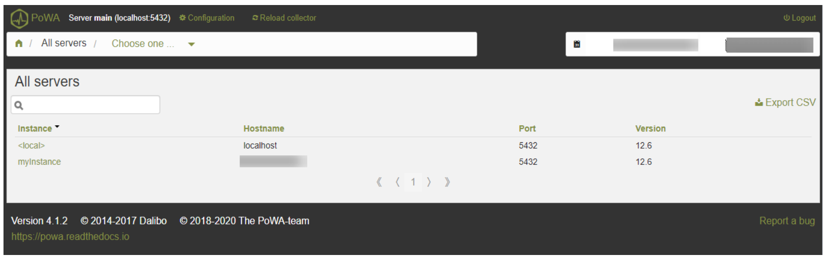
2021-11-25 15:03:39 2021-11-25 16:03:39

Server: main

Version 4.1.2 © 2014-2017 Dalibo © 2018-2020 The PoWA-team
https://powa.readthedocs.io

Step 2 Enter the username and password, and click **Login**.

Figure 3-15 PoWA home page



Server main (localhost:5432)

[redacted]

All servers

Instance	Hostname	Port	Version
<local>	localhost	5432	12.6
myInstance	[redacted]	5432	12.6

Version 4.1.2 © 2014-2017 Dalibo © 2018-2020 The PoWA-team
https://powa.readthedocs.io

The PoWA collects information about two PostgreSQL instances.

- **<local>**: PostgreSQL database built on the ECS, which is used as the powa-repository role.
- **myinstance**: RDS for PostgreSQL instance, which is used as the target for performance data collection. (**myinstance** is the instance alias registered in powa-repository.)

Step 3 Click an instance to view its performance metrics.

----End

3.3.5 Example for Using PoWA

PoWA can collect and display various performance metrics. This example describes how to view a slow SQL statement.

Step 1 Log in to the management console.

Step 2 Click in the upper left corner and select a region and a project.

Step 3 Click in the upper left corner of the page and choose **Databases** > **Relational Database Service**.

Step 4 On the **Instances** page, locate the target DB instance and click **Log In** in the **Operation** column.

Alternatively, click the instance name on the **Instances** page. On the displayed **Basic Information** page, click **Log In** in the upper right corner of the page.

Step 5 On the displayed login page, enter the correct username and password and click **Log In**.

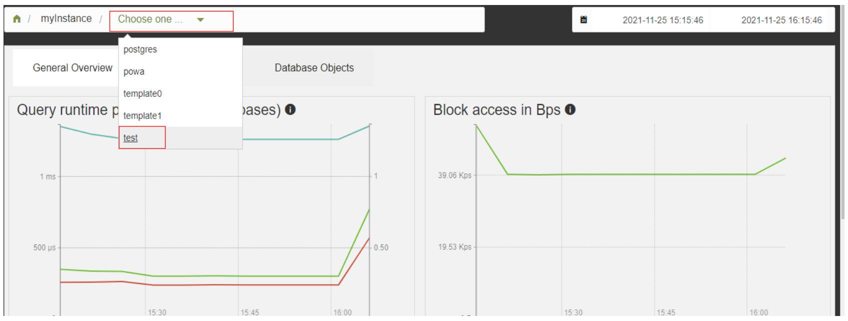
Step 6 In the database list of the **Home** page, click **Create Database**.

Step 7 On the displayed page, enter a database name **test** and select a character set.

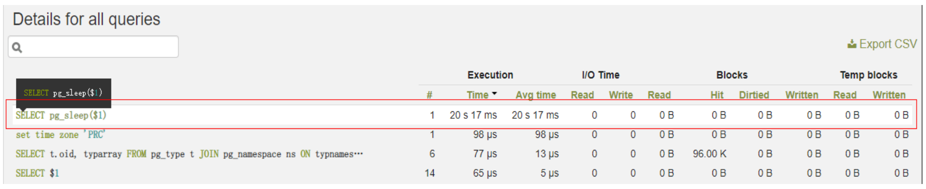
Step 8 Choose **SQL Operations** > **SQL Query** to execute a slow SQL statement, for example, SELECT pg_sleep($1), in the **test** database.

Step 9 After about 5 minutes, on the PoWA home page, select the target DB instance and select the **test** database.

Figure 3-16 PoWA home page



In **Details for all queries**, view that the execution time of the SELECT pg_sleep($1) statement is 20s.



Details for all queries

Export CSV

		Execution		I/O Time		Blocks				Temp blocks	
	#	Time	Avg time	Read	Write	Read	Hit	Dirtied	Written	Read	Written
SELECT pg_sleep($1)	1	20 s 17 ms	20 s 17 ms	0	0	0 B	0 B	0 B	0 B	0 B	0 B
set time zone 'PRC'	1	98 µs	98 µs	0	0	0 B	0 B	0 B	0 B	0 B	0 B
SELECT t.oid, typarray FROM pg_type t JOIN pg_namespace ns ON typnames···	6	77 µs	13 µs	0	0	0 B	96.00 K	0 B	0 B	0 B	0 B
SELECT $1	14	65 µs	5 µs	0	0	0 B	0 B	0 B	0 B	0 B	0 B

----End

3.3.6 Other Extension Plug-Ins

In addition to mandatory plug-ins pg_stat_statements, btree_gist, and PoWA, the following plug-ins are used for collecting new performance indicators:

- pg_qualstats
- pg_stat_kcache
- pg_wait_sampling
- pg_track_settings
- hypopg

Each of the plug-ins can extend different performance metrics.

Currently, only pg_track_settings is supported on Huawei Cloud.

pg_track_settings Plug-In Extension

Step 1 **Log in to the management console**.

Step 2 Click in the upper left corner and select a region and a project.

Step 3 Click ☰ in the upper left corner of the page and choose **Databases** > **Relational Database Service**.

Step 4 On the **Instances** page, locate the target DB instance and click **Log In** in the **Operation** column.

Alternatively, click the instance name on the **Instances** page. On the displayed **Basic Information** page, click **Log In** in the upper right corner of the page.

Step 5 On the displayed login page, enter the correct username and password and click **Log In**.

Step 6 Select the powa database and run the SQL command to create pg_track_settings.

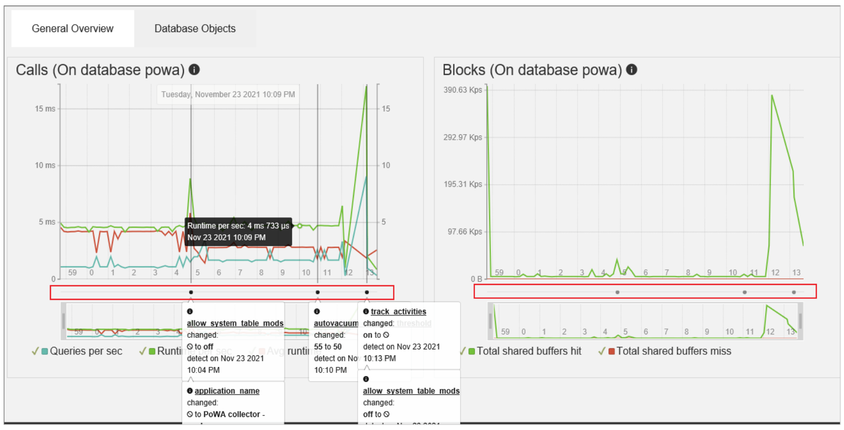
```
select control_extension('create', 'pg_track_settings');
```

Step 7 Create a PostgreSQL database (powa-repository) on the ECS, and install and activate pg_track_settings to collect performance metrics.

```
# pg_track_settings
cd /home/postgres/env
wget https://github.com/rjuju/pg_track_settings/archive/refs/tags/2.0.1.tar.gz
mv 2.0.1.tar.gz pg_track_settings.2.0.1.tar.gz
tar -xzvf pg_track_settings.2.0.1.tar.gz
cd pg_track_settings-2.0.1
make && make install
# powa-repository
psql -d powa
powa=# create extension pg_track_settings ;
CREATE EXTENSION
# Activate the pg_track_settings collection function for the target instance.
dbpowa=# select powa_activate_extension(1, 'pg_track_settings');
powa_activate_extension
-------------------------
t
(1 row)
```

Step 8 Verify the pg_track_settings plug-in extension.

Change the value of the **autovacuum_analyze_threshold** parameter on the target instance to **55**. The default value is **50**. After about 5 minutes, you can view the modification record on the PoWA, as shown in the following figure.



allow_system_table_mods
changed:
⃠ to off
detect on Nov 23 2021
10:04 PM

application_name
changed:
⃠ to PoWA collector -

autovacuum
changed:
55 to 50
detect on Nov
10:10 PM

track_activities
changed:
on to ⃠
detect on Nov 23 2021
10:13 PM

allow_system_table_mods
changed:
off to ⃠

The contents in the three boxes in the preceding figure are as follows:

- The time when pg_track_settings is activated and the database parameter value at that time.

- The time when the **autovacuum_analyze_threshold** parameter is modified, its original value, and changed value.
- The time when pg_track_settings is canceled and the database parameter value at that time.

----End

3.3.7 Operation References

3.3.7.1 Installing Python 3.9.9

Preparing the Environment

Perform the following operations in sequence. Otherwise, Python 3.9.9 may fail to be installed (SSL component dependency fails). As a result, PoWA-collector and PoWA-web fail to be installed.

```
yum install readline* -y
yum install zlib* -y
yum install gcc-c++ -y
yum install sqlite* -y
yum install openssl-* -y
yum install libffi* -y
```

Installing Python 3.9.9

1. Run the following command as user **root**:
   ```
   mkdir env
   cd env
   wget https://www.python.org/ftp/python/3.9.9/Python-3.9.9.tgz
   tar -xzvf Python-3.9.9.tgz
   cd Python-3.9.9
   ./configure --prefix=/usr/local/python3.9.9
   make && make install
   ```
2. Create a soft link.
   ```
   ln -s /usr/local/python3.9.9/bin/python3.9 /usr/bin/python
   ln -s /usr/local/python3.9.9/bin/pip3.9 /usr/bin/pip
   ```

Verify the Installation.

1. Verify the installation, especially the SSL function.
   ```
   [root@ecs-ad4d Python-3.9.9]# python
   Python 3.9.9 (main, Nov 25 2021, 12:36:32)
   [GCC 8.4.1 20200928 (Red Hat 8.4.1-1)] on linux
   Type "help", "copyright", "credits" or "license" for more information.
   ```

import ssl

import urllib.request

context = ssl._create_unverified_context()

urllib.request.urlopen('https://www.*example*.com/',context=context).read()

2. If any command output is displayed, the installation is successful. Run the following command to exit.

   quit()

3.4 RDS for PostgreSQL Publications and Subscriptions

Logical Definition

A publication can be defined on any primary physical replication server. The node where a publication is defined is called the publisher. A publication is a set of changes generated from a table or a group of tables, and might also be described as a change set or replication set. Each publication exists in only one database.

A subscription is the downstream side of logical replication. The node where a subscription is defined is called the subscriber. A subscription defines the connection to another database and the set of publications (one or more) to which it wants to subscribe. The subscriber database behaves in the same way as any other RDS for PostgreSQL instance (primary) and can be used as a publisher for other databases by defining its own publications.

Required Permissions

- To create a publication, the publisher must have the replication permission.
- When creating a publication for ALL TABLES, ensure that the publisher uses the **root** user of the initial or later versions for privilege escalation.
- When creating or deleting a subscription, ensure that the subscriber uses the **root** user of the initial or later versions for privilege escalation.

For details about **root** privilege escalation of each version, see **Privileges of the root User**.

Restrictions on Publications

- Publications may currently only contain tables (indexes, sequence numbers, and materialized views cannot be published). Each table can be added to multiple publications if needed.
- One publication can have multiple subscribers.
- ALL TABLES can be used to publish all tables.
- Multiple publications can be created in a given database, but each publication must have a unique name. The created publications can be obtained by querying pg_publication.
- Publications can choose to limit the changes they produce to any combination of INSERT, UPDATE, DELETE, and TRUNCATE, similar to how triggers are fired by particular event types. By default, all operation types are replicated.

  Example: To publish the UPDATE and DELETE operations on the **t1** table:

```
CREATE PUBLICATION update_delete_only FOR TABLE t1
    WITH (publish = 'update, delete') ;
```

- Replica identity: A published table must have a replica identity configured in order to be able to replicate UPDATE and DELETE operations. If **nothing** is set for the replica identity, subsequent UPDATE or DELETE operations will cause an error on the publisher.

  You can obtain the replica identity of a table from **pg_class.relreplident**.

  relreplident is of character type and identifies columns used to form "replica identity" for rows: d = default, f = all columns, i = index, and n = nothing.

  To check whether a table has an index constraint that can be used as a replica identity, run the following:

```
SELECT quote_ident(nspname) || '.' || quote_ident(relname) AS name, con.ri AS keys,
       CASE relreplident WHEN 'd' THEN 'default' WHEN 'n' THEN 'nothing' WHEN 'f' THEN
'full' WHEN 'i' THEN 'index' END AS replica_identity
FROM pg_class c JOIN pg_namespace n ON c.relnamespace = n.oid, LATERAL (SELECT
array_agg(contype) AS ri FROM pg_constraint WHERE conrelid = c.oid) con
WHERE relkind = 'r' AND nspname NOT IN ('pg_catalog', 'information_schema', 'monitor',
'repack', 'pg_toast')
ORDER BY 2,3;
```

- Command for changing a replica identity

  The replica identity of a table can be changed using **ALTER TABLE**.

```
ALTER TABLE table_name REPLICA IDENTITY
{ DEFAULT | USING INDEX index_name | FULL | NOTHING };
-- There are four forms:
ALTER TABLE t_normal REPLICA IDENTITY DEFAULT;                    -- The primary key is
used as the replica identity. If there is no primary key, the replica identity is set to FULL.
ALTER TABLE t_normal REPLICA IDENTITY FULL;                       -- The entire row is used
as the replica identity.
ALTER TABLE t_normal REPLICA IDENTITY USING INDEX t_normal_v_key; -- A unique index
is used as the replica identity.
ALTER TABLE t_normal REPLICA IDENTITY NOTHING;                    -- No replica identity is
set.
```

- Precautions for using replica identities
  - If a table has a primary key, the replica identity can be set to DEFAULT.
  - If a table does not have a primary key but has a non-null unique index, the replica identity can be set to INDEX.
  - If a table does not have a primary key or a non-null unique index, the replica identity can be set to FULL. This, however, is very inefficient and should only be used as a fallback if no other solution is possible.
  - In all cases other than those mentioned above, logical replication cannot be implemented. The output information is insufficient, and an error may be reported.
  - **If a table with replica identity "nothing" is added to logical replication, deleting or updating the table will cause an error on the publisher.**

Restrictions on Subscriptions

- To ensure that failover slots are used, failover slots must be created on the publisher and associated with the existing replication slots using **create_slot = false**.

CREATE SUBSCRIPTION sub1 CONNECTION 'host=192.168.0.1 port=5432 user=user1 dbname=db1' PUBLICATION pub_name with (create_slot = false,slot_name = FailoverSlot_name);

- Logical replication does not replicate DDL changes, so the tables in the publication set must already exist on the subscriber.
- Multiple subscriptions can be created in a given database. These subscriptions can come from one or more publishers.
- A given table of a subscriber cannot accept multiple publications from the same source.
- When creating a subscription or altering a subscription, you can use **enable** to enable the subscription or **disable** to suspend the subscription.
- To delete a subscription, use **DROP SUBSCRIPTION**. Note that after a subscription is deleted, the local table and data are not deleted, but upstream information of the subscription is no longer received.

NOTICE

If a subscription is associated with a replication slot, **DROP SUBSCRIPTION** cannot be executed inside a transaction block. You can use **ALTER SUBSCRIPTION** to disassociate the subscription from the replication slot.

To completely delete a subscription, perform the following steps:

a. Query the replication slot associated with the subscription on the subscriber.

   select subname,subconninfo,subslotname from pg_subscription where subname = 'sub2';

   - **subname** indicates the subscriber name.
   - **subconninfo** indicates information about the connected remote host.
   - **subslotname** indicates the replication slot name of the remote host.

b. On the subscriber, disassociate the subscription from the replication slot and delete the subscription.

   ALTER SUBSCRIPTION subname SET (slot_name = NONE);

   DROP SUBSCRIPTION subname;

c. Delete the associated replication slot at the publisher.

   select pg_drop_replication_slot(' slot_name);

Syntax Reference

- Publications

  CREATE PUBLICATION is used to create a publication, **DROP PUBLICATION** is used to delete a publication, and **ALTER PUBLICATION** is used to modify a publication.

  After a publication is created, tables can be added or removed dynamically using **ALTER PUBLICATION**. Such operations are all transactional.

- Subscriptions

  CREATE SUBSCRIPTION is used to create a subscription, **DROP SUBSCRIPTION** is used to delete a subscription, and **ALTER SUBSCRIPTION** is used to modify a subscription.

  After creating a subscription, you can use **ALTER SUBSCRIPTION** to suspend or resume the subscription at any time. Deleting and recreating a subscription results in the loss of synchronized information, which means that related data needs to be synchronized again.

For details, see the official documentation. PostgreSQL 13 is used as an example.

- Creating a publication: **https://www.postgresql.org/docs/13/sql-createpublication.html**
- Deleting a publication: **https://www.postgresql.org/docs/13/sql-droppublication.html**
- Modifying a publication: **https://www.postgresql.org/docs/13/sql-alterpublication.html**

3.5 Viewing Slow Query Logs of RDS for PostgreSQL DB Instances

Scenarios

Slow query logs record statements that exceed the **log_min_duration_statement** value (1 second by default). You can view log details and statistics to identify statements that are slowly executed and optimize the statements. RDS for PostgreSQL supports the following statement types:

- SELECT
- INSERT
- UPDATE
- DELETE
- CREATE
- DROP
- ALTER
- DO
- CALL
- COPY

Parameter Description

Table 3-17 Parameters related to RDS for PostgreSQL slow queries

Parameter	Description
log_min_duration_statement	Specifies the minimum execution time. The statements whose execution time is greater than or equal to the value of this parameter are recorded. If this parameter is set to a smaller value, the number of log records increases, which increases the disk I/O and deteriorates the SQL performance.
log_statement	Specifies the statement type. The value can be **none**, **ddl**, **mod**, or **all**. The default value is **none**. If you change the value to **all**: ● The database disk I/O increases, and the SQL performance deteriorates. ● The log format changes, and you cannot view slow query logs on the console.
log_statement_stats	Specifies whether to output performance statistics to server logs. The default value is **off**. If you change the value to **on**: ● The database disk I/O increases, and the SQL performance deteriorates. ● The log format changes, and you cannot view slow query logs on the console.

Procedure

Step 1 Log in to the management console.

Step 2 Click in the upper left corner and select a region and a project.

Step 3 Click in the upper left corner of the page and choose **Databases** > **Relational Database Service**.

Step 4 On the **Instances** page, click the target DB instance.

Step 5 In the navigation pane on the left, choose **Logs**. On the **Slow Query Logs** page, click **Log Details**.

You can view the slow query log records of a specified statement type in a specified time period.

----End

3.6 Using Client Drivers to Implement Failover and Read/Write Splitting

Since PostgreSQL 10 (libpq.so.5.10), libpq has been supporting failover and read/write splitting, and Java Database Connectivity (JDBC) has been supporting read/write splitting, failover, and load balancing.

PostgreSQL client drivers are backward compatible. Even RDS for PostgreSQL 9.5 and 9.6 instances can be connected through the libpq driver of the latest version to implement failover.

NOTE

In this section, failover refers to the failover of read-only workloads.

- libpq is a C application programming interface (API) to PostgreSQL. libpq is a set of library functions that allow client programs to pass queries to the PostgreSQL backend server and to receive the results of these queries.
- JDBC is an API used in Java to define how client programs access databases. In PostgreSQL, JDBC supports failover and load balancing.

Table 3-18 Functions supported by libpq and JDBC

Driver	Read/Write Splitting	Load Balancing	Failover
libpq	√	×	√
JDBC	√	√	√

Using libpq for Failover and Read/Write Splitting

You can use libpq functions to connect to multiple databases. If one database fails, workloads are automatically switched to another available database.

postgresql://[user[:password]@][netloc][:port][,...][/dbname][?param1=value1&...]

Example: Connect to one primary RDS for PostgreSQL instance and two read replicas. Read requests will not fail as long as there is at least one available instance.

postgres://10.6.10.194:5432,10.6.2.98:5432,10.6.9.43:5432/postgres?target_session_attrs=any

Table 3-19 Parameter description

Parameter	Description	Example Value
target_session_attrs	Type of the database to be connected.	• **any** (default): libpq can connect to any database. If the connection is interrupted due to a fault in the database, libpq will attempt to connect to another database to implement failover. • **read-write**: libpq can only connect to a database that supports both read and write. libpq attempts a connection to the first database you specified. If this database supports only read or write operations, libpq disconnects from it and attempts to connect to the second one and so on until it connects to a database that supports both read and write. • **read-only**: libpq can only connect to a read-only database. libpq attempts a connection to the first database you specified. If this database is not a read-only database, libpq disconnects from it and attempts to connect to the second one and so on until it connects to a read-only database. This value is not supported in RDS for PostgreSQL 13 (libpq.so.5.13) or earlier versions.

For details about libpq and related parameters, see **Connection Strings**.

You can use the **pg_is_in_recovery()** function in your application to determine whether the connected database is a primary instance (indicated by **f**) or a read replica to implement read/write splitting.

The following is an example of Python code (psycopg2 a wrapper for libpq):

```
import psycopg2
conn = psycopg2.connect(database="postgres",host="10.6.10.194,10.6.2.98", user="root",
password="******", port="5432,5432", target_session_attrs="read-write")
cur = conn.cursor()
cur.execute("select pg_is_in_recovery()")
row = cur.fetchone()
print("recovery =", row[0])
```

Using JDBC for Failover and Read/Write Splitting

You can define multiple databases (hosts and ports) in the connection URL and separate them with commas (,). JDBC will attempt to connect to them in sequence until the connection is successful. If the connection fails, an error message is displayed.

jdbc:postgresql://node1,node2,node3/${database}?targetServerType=preferSecondary&loadBalanceHosts=true

Example:

jdbc:postgresql://10.6.10.194:5432,10.6.2.98:5432,10.6.9.43:5432/${database}?targetServerType=preferSecondary&loadBalanceHosts=true

For details about the Java code, see **Connecting to an RDS for PostgreSQL Instance Through JDBC**.

Table 3-20 Parameter description

Parameter	Description	Example Value
targetServerType	Type of the database to be connected.	• **any**: any database. • **primary**: primary database (writable and readable). For versions earlier than JDBC 42.2.0, use the parameter value **master**. • **secondary**: secondary database (readable). For versions earlier than JDBC 42.2.0, use the parameter value **slave**. • **preferSecondary**: The secondary database is preferred. If no secondary database is available, the primary database is connected. For versions earlier than JDBC 42.2.0, use the parameter value **preferSlave**.
loadBalanceHosts	Sequence of databases to be connected.	• **False** (default): Databases are connected in the sequence defined in the URL. • **True**: Databases are randomly connected.

NOTE

To distinguish between the primary and secondary databases, check whether data can be written to the database. If yes, it is a primary database. If no, it is a secondary database. You can use the **pg_is_in_recovery()** function to determine whether a database is a primary database. For details, see **Using libpq for Failover and Read/Write Splitting**.

To implement read/write splitting, you need to configure two data sources. For the first data source, set **targetServerType** to **primary** to process write requests. For the second data source:

- If there is only one read replica, set **targetServerType** to **preferSecondary** to process read requests. Assume that the IP addresses of the primary instance and read replica are 10.1.1.1 and 10.1.1.2, respectively.

  jdbc:postgresql://10.1.1.2:5432,10.1.1.1:5432/${database}?targetServerType=preferSecondary

- If there are two read replicas, set **targetServerType** to **any** to process read requests. Assume that the IP addresses of the read replicas are 10.1.1.2 and 10.1.1.3, respectively.

  jdbc:postgresql://10.1.1.2:5432,10.1.1.3:5432/${database}?targetServerType=any&loadBalanceHosts=true

3.7 User-Defined Data Type Conversion

Description

There are three data type conversion modes for PostgreSQL: implicit conversion, assignment conversion, and explicit conversion. They correspond to i (Implicit), a (Assignment), and e (Explicit) in the pg_cast system catalog.

- Implicit conversion: a conversion from low bytes to high bytes of the same data type, for example, from **int** to **bigint**
- Assignment conversion: a conversion from high bytes to low bytes of the same data type, for example, from **smallint** to **int**
- Explicit conversion: a conversion between different data types

How to Use

1. Before converting data types, you can run the following command to check whether RDS for PostgreSQL supports data type conversion:
```
select * from pg_catalog.pg_cast ;
oid  | castsource | casttarget | castfunc | castcontext | castmethod
-------+------------+------------+----------+-------------+------------
 11277 |         20 |         21 |      714 | a           | f
 11278 |         20 |         23 |      480 | a           | f
 11279 |         20 |        700 |      652 | i           | f
 11280 |         20 |        701 |      482 | i           | f
......
```
2. Run the following command to check whether **int4** can be converted to **text**:
```
select * from pg_catalog.pg_cast where castsource = 'int4'::regtype and casttarget = 'bool'::regtype;
  oid  | castsource | casttarget | castfunc | castcontext | castmethod
-------+------------+------------+----------+-------------+------------
 11311 |         23 |         16 |     2557 | e           | f
(1 row)
```
   The conversion is supported, and the conversion type is implicit conversion.

   If no built-in conversion functions are available, customize a conversion function to support the conversion. For details, see **User-Defined Data Type Conversion**.

User-Defined Data Type Conversion

- Use double colons (::) to perform a forcible conversion.
```
select '10'::int,'2023-10-05'::date;
 int4 |    date
------+------------
   10 | 2023-10-05
(1 row)
```
- Use the CAST function to convert the type.
```
select CAST('10' as int),CAST('2023-10-05' as date);
 int4 |    date
------+------------
   10 | 2023-10-05
(1 row)
```
- Customize a data type conversion.

  For details, see **https://www.postgresql.org/docs/14/sql-createcast.html**.

NOTICE

Adding a custom type conversion will affect the existing execution plans of RDS for PostgreSQL. Therefore, customizing type conversions are not recommended.

- Conversion between time and character types

  CREATE CAST(varchar as date) WITH INOUT AS IMPLICIT;
- Conversion between boolean types and numeric types

  create cast(boolean as numeric) with INOUT AS IMPLICIT;
- Conversion between numeric types and character types

  create cast(varchar as numeric) with INOUT AS IMPLICIT;

Example: Convert **text** to **date**.

```
create or replace function public.text_to_date(text) returns date as
$$
  select to_date($1,'yyyy-mm-dd');
$$
language sql strict;

create cast (text as date) with function public.text_to_date(text) as implicit;

select text '2023-09-09' + 1;
  ?column?
------------
 2023-09-10
(1 row)
```

3.8 Security Best Practices

PostgreSQL has earned a reputation for reliability, stability, and data consistency, and has become the preferred choice as an open-source relational database for many enterprises. RDS for PostgreSQL is a cloud-based web service that is reliable, scalable, easy to manage, and immediately ready for use.

Make security configurations from the following dimensions to meet your service needs.

- **Configuring the Maximum Number of Connections to the Database**
- **Configuring the Timeout for Client Authentication**
- **Configuring SSL and Encryption Algorithm**
- **Configuring Password Encryption**
- **Disabling the Backslash Quote**
- **Periodically Checking and Deleting Roles That Are No Longer Used**
- **Revoking All Permissions on the public Schema**
- **Setting a Proper Password Validity Period for a User Role**
- **Configuring the Log Level to Record SQL Statements That Cause Errors**
- **Providing Least Privileges for Database Accounts**
- **Enabling Data Backup**
- **Enabling Database Audit**

- **Avoiding Access to Your RDS for PostgreSQL Instance over the Internet**
- **Updating the Database Version to the Latest**

Configuring the Maximum Number of Connections to the Database

The **max_connections** parameter specifies the maximum concurrent connections allowed in a database. If the value of this parameter is large, the RDS for PostgreSQL database may request more System V shared memory or semaphore. As a result, the requested shared memory or semaphore may exceed the default value on the OS. Set **max_connections** based on service complexity. For details, see **Instance Usage Suggestions**.

Configuring the Timeout for Client Authentication

The **authentication_timeout** parameter specifies the maximum duration allowed for client authentication, in seconds. This parameter prevents the client from occupying the connection channel for a long time. The default value is 60s. If the authentication is not complete within the specified period, the connection is forcibly closed. This configuration can enhance the security of your RDS for PostgreSQL instance.

Configuring SSL and Encryption Algorithm

SSL is recommended for TCP/IP connections because SSL ensures that all communications between the client and server are encrypted, preventing data leakage and tampering and ensuring data integrity. When configuring SSL encryption, you need to configure a secure TLS protocol and encryption algorithm on the server. TLSv1.2 and EECDH+ECDSA+AESGCM:EECDH+aRSA+AESGCM:EDH+aRSA+AESGCM:EDH+aDSS+AESGCM:!aNULL:!eNULL:!LOW:!3DES:!MD5:!EXP:!SRP:!RC4 are recommended. For details, see **SSL Connection**.

To configure the TLS protocol and encryption algorithm, change the values of **ssl_min_protocol_version** and **ssl_ciphers**, respectively.

Configuring Password Encryption

Passwords must be encrypted. When you use **CREATE USER** or **ALTER ROLE** to change a password, the password is encrypted by default. The password encryption method **scram-sha-256** is recommended. To change the password encryption method, change the value of **password_encryption**.

The **MD5** option is used only for compatibility with earlier versions. New DB instances use **scram-sha-256** by default.

NOTICE

The modification of **password_encryption** takes effect only after the password is reset.

Disabling the Backslash Quote

The **backslash_quote** parameter specifies whether a single quotation mark (') in a string can be replaced by a backslash quote (\'). The preferred, SQL-standard way to represent a single quotation mark is by doubling it (''). If client-side code does escaping incorrectly then an SQL-injection attack is possible. You are advised to set **backslash_quote** to **safe_encoding** to reject queries in which a single quotation mark appears to be escaped by a backslash, preventing SQL injection risks.

Periodically Checking and Deleting Roles That Are No Longer Used

Check whether all roles are mandatory. Any unknown role must be reviewed to ensure that it is used properly. If any role is no longer used, delete it. To query roles, run the following command:

SELECT rolname FROM pg_roles;

Revoking All Permissions on the public Schema

The **public** schema is the default schema. All users can access objects in it, including tables, functions, and views, which may cause security vulnerabilities. You can run the following command as user **root** to revoke the permissions:

revoke all on schema public from public;

Setting a Proper Password Validity Period for a User Role

When creating a role, you can use the **VALID UNTIL** keyword to specify when the password of the role becomes invalid. If this keyword is ignored, the password will be valid permanently. You are advised to change the password periodically, for example, every three months. To configure a password validity period, run the following command:

CREATE ROLE name WITH PASSWORD *'password'* **VALID UNTIL 'timestamp';**

To check whether a password validity period is configured, run the following command:

SELECT rolname,rolvaliduntil FROM pg_roles WHERE rolsuper = false AND rolvaliduntil IS NULL;

Configuring the Log Level to Record SQL Statements That Cause Errors

The **log_min_error_statement** parameter specifies which SQL statements that cause errors can be recorded in server logs. The SQL statements of the specified level or higher are recorded in logs. Valid values include **debug5**, **debug4**, **debug3**, **debug2**, **debug1**, **info**, **notice**, **warning**, **error**, **log**, **fatal**, and **panic**. The value of **log_min_error_statement** must be at least **error**. For details, see **Log Reporting**.

Providing Least Privileges for Database Accounts

RDS for PostgreSQL allows you to grant role-based permissions to a database account for data and command access. You are advised to create **database accounts** and authorize the accounts based on the least privilege principle. If any account permission does not meet the role requirements, update the account permission or **delete** the account based on service requirements. RDS for PostgreSQL has some **built-in accounts**, which are used to provide comprehensive background O&M services for DB instances and cannot be used or deleted by users.

Enabling Data Backup

When you create an RDS DB instance, an automated backup policy is enabled by default with the retention period set to seven days. You can change the backup retention period as required. RDS for PostgreSQL DB instances support **automated backups** and **manual backups**. You can periodically back up your instance. If the instance fails or data is damaged, **restore it using backups** to ensure data reliability. For details, see **Data Backups**.

Enabling Database Audit

By using the PostgreSQL Audit Extension (pgAudit) with your RDS for PostgreSQL instance, you can capture detailed records that auditors usually need to meet compliance regulations. For example, you can use pgAudit to track changes made to specific databases and tables, as well as record users who make such changes and many other details. pgAudit is disabled by default. You can enable it based on service requirements. For details, see **Using pgAudit**.

Avoiding Access to Your RDS for PostgreSQL Instance over the Internet

Do not deploy your instance on the Internet or in a demilitarized zone (DMZ). Instead, deploy it on a Huawei Cloud private network and use routers or firewalls to protect it. Do not bind an EIP to your instance for access over the Internet to prevent unauthorized access and DDoS attacks. If you have bound an EIP to your instance, you are advised to unbind it. If you do need an EIP, **configure security group rules** to restrict the source IP addresses that can access your instance.

Updating the Database Version to the Latest

PostgreSQL 9.5, 9.6, and 10 have reached end of life (EOL) and are no longer maintained by the community. **An EOS notice** has been released for RDS for PostgreSQL 9.5 and 9.6. Using an earlier version may pose security risks. Running the software of the latest version can protect the system from certain attacks. You can upgrade **the minor version** or **the major version** of your DB instance as required.

4 RDS for SQL Server

4.1 Restoring Data from Backup Files to RDS Microsoft SQL Server DB Instances

This section describes how to use automated or manual backup files to restore a DB instance to the status when the backup was created.

Best Practices

- You can create a full backup file for your DB instance and use OBS and DRS to restore the backup file to an RDS Microsoft SQL Server DB instance.
- The restoration must be from an earlier database version to the same or a later version. The version of local backups must be earlier than or the same as the version of the destination DB instance to be restored.

  NOTE

  For example, if the local database version is Microsoft SQL Server 2012 Standard Edition, you can only restore the local backups to Standard or Enterprise Edition of Microsoft SQL Server 2014 or 2016. You cannot restore the local backups to any versions of Microsoft SQL Server 2008 or Web Editions of Microsoft SQL Server 2014 and 2016.
- For operation details on the RDS console, see **Restoring a DB Instance to a Point in Time** and **Restoring a DB Instance from a Backup**.

4.2 Migrating Data from a Self-Managed SQL Server Database on an ECS to an RDS for SQL Server DB Instance

Scenarios

- You have created a Microsoft SQL Server database on an ECS.
- The self-managed SQL Server database version on the ECS cannot be later than the version of the RDS for SQL Server DB instance.

- You have installed the SQL Server Management Studio (SSMS).

Procedure

Step 1 Create an ECS.

NOTE

The ECS and the RDS DB instance must be in the same region and VPC.

Step 2 Install Microsoft SQL Server 2008, 2012, or 2014 on the ECS.

NOTE

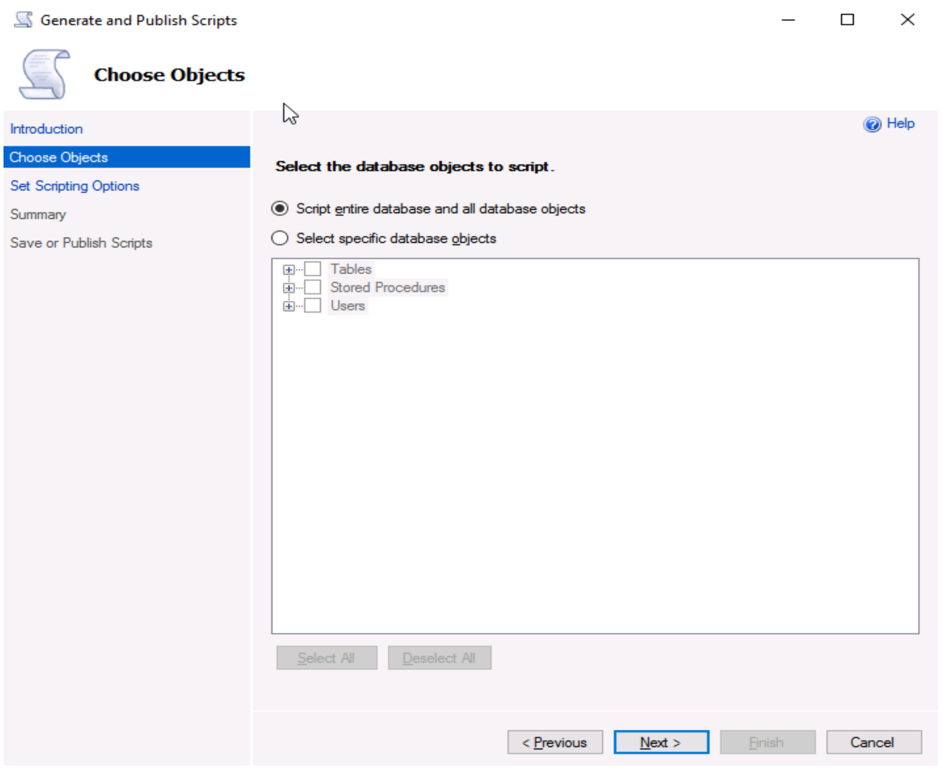
The Microsoft SQL Server installed on the ECS must be Standard or Enterprise Edition. It is recommended that the Microsoft SQL Server version be the same as the RDS DB instance version.

Step 3 Upload a local .bak file to the ECS and use Microsoft SQL Server to restore the local file to the RDS DB instance.

Step 4 Use the script generation tool provided by Microsoft SQL Server to generate a database structure script.

1. Right-click the database whose schema script needs to be generated and choose **Tasks** > **Generate Scripts**.
2. On the **Choose Objects** page, choose database objects to script, as shown in **Figure 4-1**. Then, click **Next**.

Figure 4-1 Choosing objects



Select the database objects to script.

(●) Script entire database and all database objects

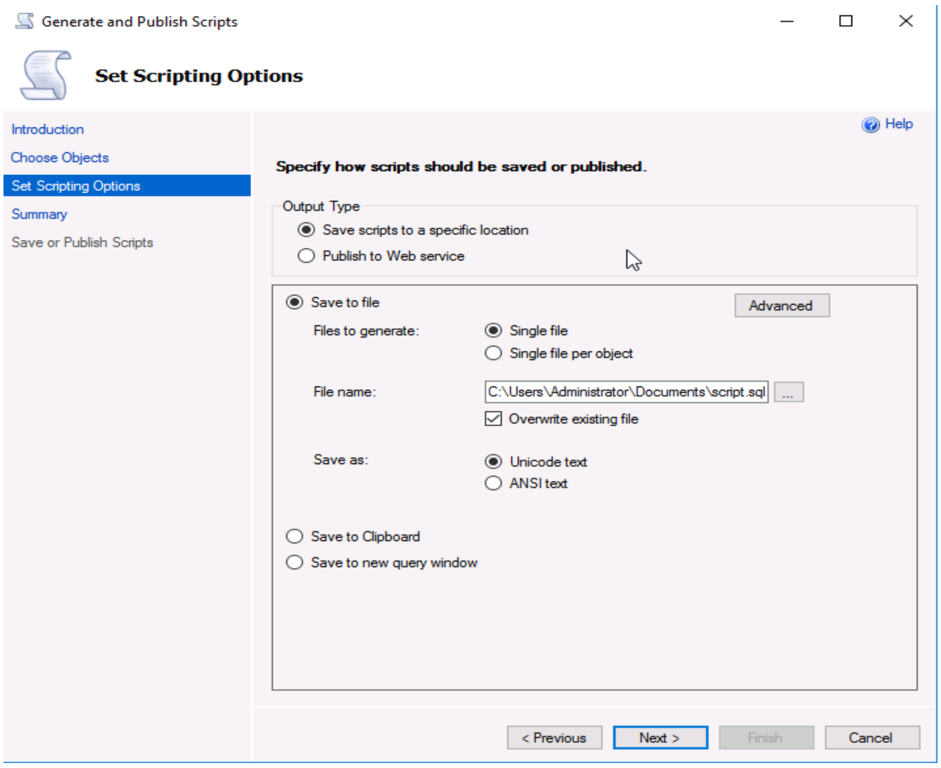
( ) Select specific database objects

3. On the **Set Scripting Options** page, specify a directory for saving the script.

NOTE

You are advised to save the script locally and generate an SQL script for execution.

Figure 4-2 Specifying a directory for saving the script



Generate and Publish Scripts

Set Scripting Options

Specify how scripts should be saved or published.

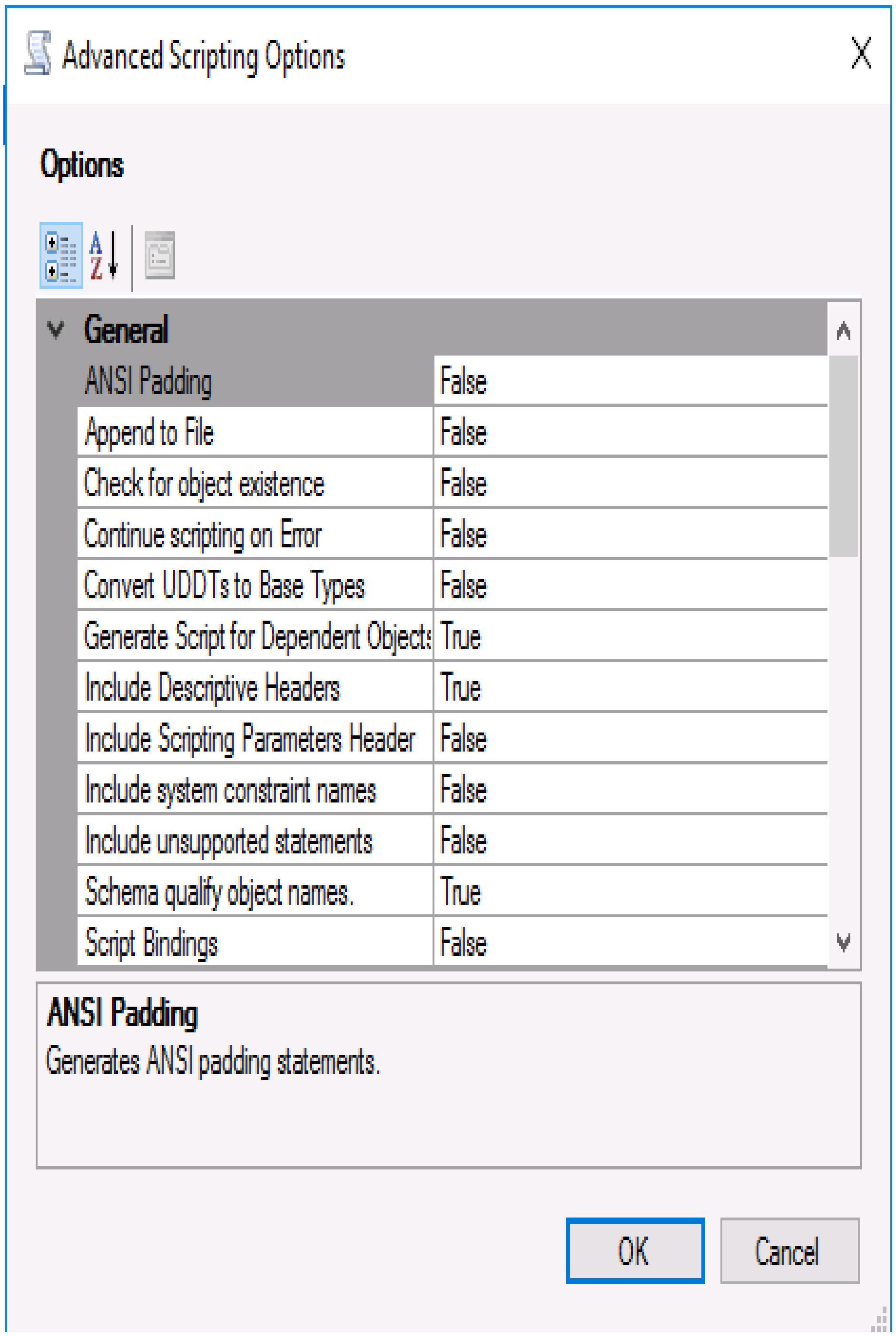
Output Type
[✓ Save scripts to a specific location]
Publish to Web service

[✓ Save to file]
Files to generate: [✓ Single file] Single file per object
File name: C:\Users\Administrator\Documents\script.sql
[✓ Overwrite existing file]
Save as: [✓ Unicode text] ANSI text

Save to Clipboard
Save to new query window

4. Click **Advanced**. In the displayed **Advanced Scripting Options** dialog box, specify scripting options for triggers, indexes, unique keys, the primary key, and server version. Then, click **OK**.

Figure 4-3 Specifying advanced scripting options



Advanced Scripting Options

Options

General	
ANSI Padding	False
Append to File	False
Check for object existence	False
Continue scripting on Error	False
Convert UDDTs to Base Types	False
Generate Script for Dependent Objects	True
Include Descriptive Headers	True
Include Scripting Parameters Header	False
Include system constraint names	False
Include unsupported statements	False
Schema qualify object names.	True
Script Bindings	False

ANSI Padding
Generates ANSI padding statements.

NOTE

Generate Script for Dependent Objects indicates the script data type option.

5. Click **Next** to generate the script.

Step 5 Use the SSMS client to connect to the RDS DB instance and open the generated SQL script.

NOTE

You need to create an empty database, and then use the script to create structures in the database.

Step 6 Use the import and export function provided by Microsoft SQL Server to migrate data.

1. Right-click the database where data is to be imported and choose **Tasks** > **Import Data**.
2. Click **Next**.
3. On the **Choose a Data Source** page, select a data source and click **Next**.
4. On the **Choose a Destination** page, select a destination database and click **Next**.
   – **Destination**: Select **SQL Server Native Client** (depending on your destination database type).
   – **Server name**: Enter the IP address and port number of the destination DB instance.
   – **Authentication**: Select **Use SQL Server Authentication**. Then, set **User name** to **rdsuser**, and **Password** to the password of **rdsuser**.
   – **Database**: Select the destination database where data is to be imported.
5. Select **Copy data from one or more tables or views** and click **Next**.
6. On the **Select Source Tables and Views** page, select the tables and views that you want to copy. Then, click **Edit Mappings**. In the displayed dialog box, select **Enable identity insert** and edit mappings based on your requirements.
7. Click **Next**.
8. Select **Run immediately** and click **Next**.
9. Click **Finish** to import data. You can view the progress. About 4,000 rows can be processed per second.

----End

4.3 Modifying Parameters of RDS for SQL Server Instances

You can modify parameters in custom parameter templates.

Each DB instance is assigned with a group of parameters when it is being created and modifications to these parameters do not affect other DB instances.

Procedure

Step 1 Log in to the management console.

Step 2 Click in the upper left corner and select a region and a project.

Step 3 Click in the upper left corner of the page and choose **Databases** > **Relational Database Service**.

Step 4 On the **Instances** page, click the target DB instance.

Step 5 On the **Parameters** page, modify the parameters as required.

NOTE

- You cannot modify parameters in a default parameter template.
  - Each Microsoft SQL Server version has a unique default parameter template.
  - To apply a default parameter template to the current DB instance, choose **Parameter Templates** page in the navigation pane on the left, locate the target template on the **Default Templates** page, and click **Apply** in the **Operation** column.
- You can modify parameters in a custom template.
  - To create a custom template, choose **Parameter Templates** page in the navigation pane on the left, click **Create Parameter Template** on the **Custom Templates** page, and configure required information in the displayed dialog box. Then, click **OK**.
  - After you save modifications to the parameters in the custom template, you can apply this parameter template to multiple DB instances running corresponding versions.

You can modify the parameters listed in **Table 4-1** to improve DB instance performance.

Table 4-1 Parameters

Parameter	Description	Application Scenario
max degree of parallelism	Specifies the maximum degree of parallelism option. When an RDS for SQL Server DB instance runs on a computer with more than one microprocessor or CPU, RDS for SQL Server detects the best degree of parallelism (the number of processors used to run a single statement) for each parallel plan execution. The default value is **0**.	• If the DB instance is used for querying results, set the parameter to **0**. • If the DB instance is used for operations such as inserting, updating, and deleting, set the parameter to **1**.
max server memory (MB)	Specifies the server memory option. It is used to reconfigure the amount of memory (in MB) in the buffer pool used by a Microsoft SQL Server DB instance.	You are advised to retain the default value for this parameter. If you want to modify this parameter, the value of this parameter must be: • No less than 2 GB. • Not greater than 95% of the maximum memory of the DB instance.

Parameter	Description	Application Scenario
user connections	Specifies the maximum number of simultaneous user connections allowed on Microsoft SQL Server. Default value: **1000**	● If this parameter is set to **0**, the number of connections to the DB instance is not limited. ● Allowed values: Values excluding 1 to 10

- To save the modifications, click **Save**.
- To cancel the modifications, click **Cancel**.
- To preview the modifications, click **Preview**.

After the parameter values are modified, you can click **Change History** to view the modification details.

----End

4.4 Supporting DMVs

RDS for SQL Server supports dynamic management views (DMVs), which enables users to quickly find 10 SQL statements with the highest performance consumption.

Scenarios

- A performance bottleneck occurs and the database execution efficiency becomes low.
- The monitoring result shows that the CPU and I/O are high in some time segments.

Procedure

Step 1 Use the **rdsuser** account to connect to the target DB instance through a client and run the following statements on the management plane:

```
declare @DatabaseName nvarchar(100)
set @DatabaseName = 'Wisdom_TT_ODS'

select top 100
DB_NAME(st.dbid) as DBName, OBJECT_NAME(st.objectid,st.dbid) as ObjectName,
substring(st.text,(qs.statement_start_offset/2)+1,((case qs.statement_end_offset when -1 then
datalength(st.text) else qs.statement_end_offset end - qs.statement_start_offset)/2) + 1) as
Statement,
st.text as Query,
qp.query_plan,
plan_generation_num,
creation_time,
last_execution_time,
execution_count,
total_worker_time,
min_worker_time,
max_worker_time,
total_logical_reads,
min_logical_reads,
max_logical_reads,
total_elapsed_time,
min_elapsed_time,
max_elapsed_time,
total_rows,
min_rows,
max_rows,
,total_worker_time/execution_count as avg_worker_time --- Average CPU duration
,total_logical_reads/execution_count as avg_logical_reads --- Average logical reads
,total_elapsed_time/execution_count as avg_elapsed_time --- Average total duration
,total_rows/execution_count as avg_rows --- Average data processing rows
sql_handle,
plan_handle,
query_hash,
query_plan_hash
from sys.dm_exec_query_stats qs
cross apply sys.dm_exec_sql_text(plan_handle) st
cross apply sys.dm_exec_query_plan(plan_handle) qp
where st.dbid=DB_ID(@DatabaseName)
and text not like '%sys.%'and text not like '%[[]sys]%'
order by avg_worker_time desc
```

Step 2 You can view the SQL execution records and resource consumption details of the corresponding database in the query result.

----End

4.5 Using the Import and Export Function to Migrate Data from a Local Database to an RDS Microsoft SQL Server DB Instance

Scenarios

- You have created a local Microsoft SQL Server database.
- The local database version cannot be later than the version of the destination RDS Microsoft SQL Server DB instance.
- You want to migrate only tables instead of the whole database.

Procedure

Step 1 **Log in to the management console**.

Step 2 Click in the upper left corner and select a region and a project.

Step 3 Click in the upper left corner of the page and choose **Databases** > **Relational Database Service**.

Step 4 Log in to the RDS console. On the **Instances** page, click the target DB instance name.

Step 5 In the navigation pane on the left, choose **Connectivity & Security**.

Step 6 In the **Connection Information** area, click **Bind** next to the **EIP** field.

Step 7 In the displayed dialog box, select an EIP and click **Yes**.

Step 8 Install the SSMS client locally and use the EIP to connect to the RDS DB instance.

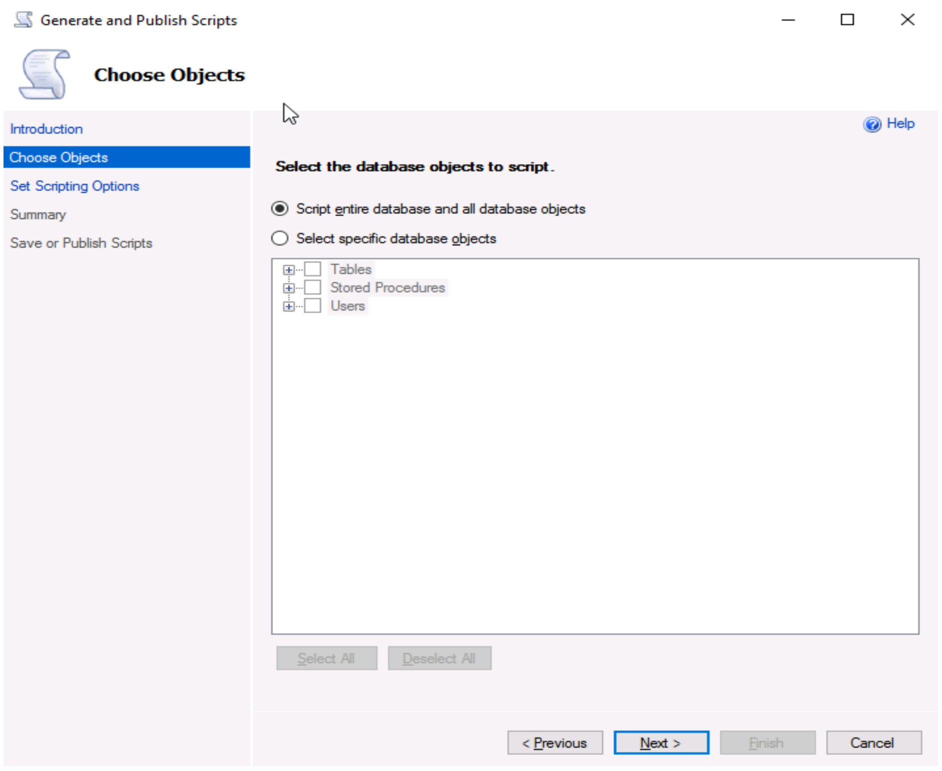
NOTE

Click **here** to download the SSMS client.

Step 9 Use the script generation tool provided by Microsoft SQL Server to generate a database structure script.

1. Right-click the database whose schema script needs to be generated and choose **Tasks** > **Generate Scripts**.
2. On the **Choose Objects** page, choose database objects to script, as shown in **Figure 4-4**. Then, click **Next**.

Figure 4-4 Choosing objects



Generate and Publish Scripts

Choose Objects

Select the database objects to script.

(●) Script entire database and all database objects

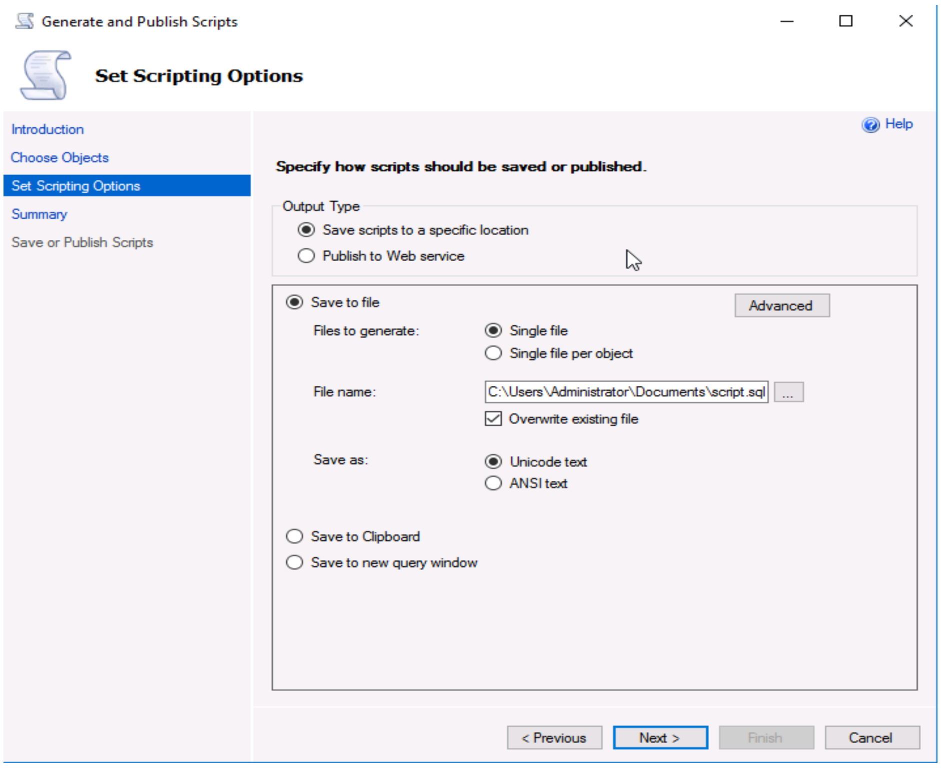
( ) Select specific database objects

3. On the **Set Scripting Options** page, specify a directory for saving the script.

NOTE

You are advised to save the script locally and generate an SQL script for execution.

Figure 4-5 Specifying a directory for saving the script



Specify how scripts should be saved or published.

Output Type

- (●) Save scripts to a specific location
- ( ) Publish to Web service

(●) Save to file

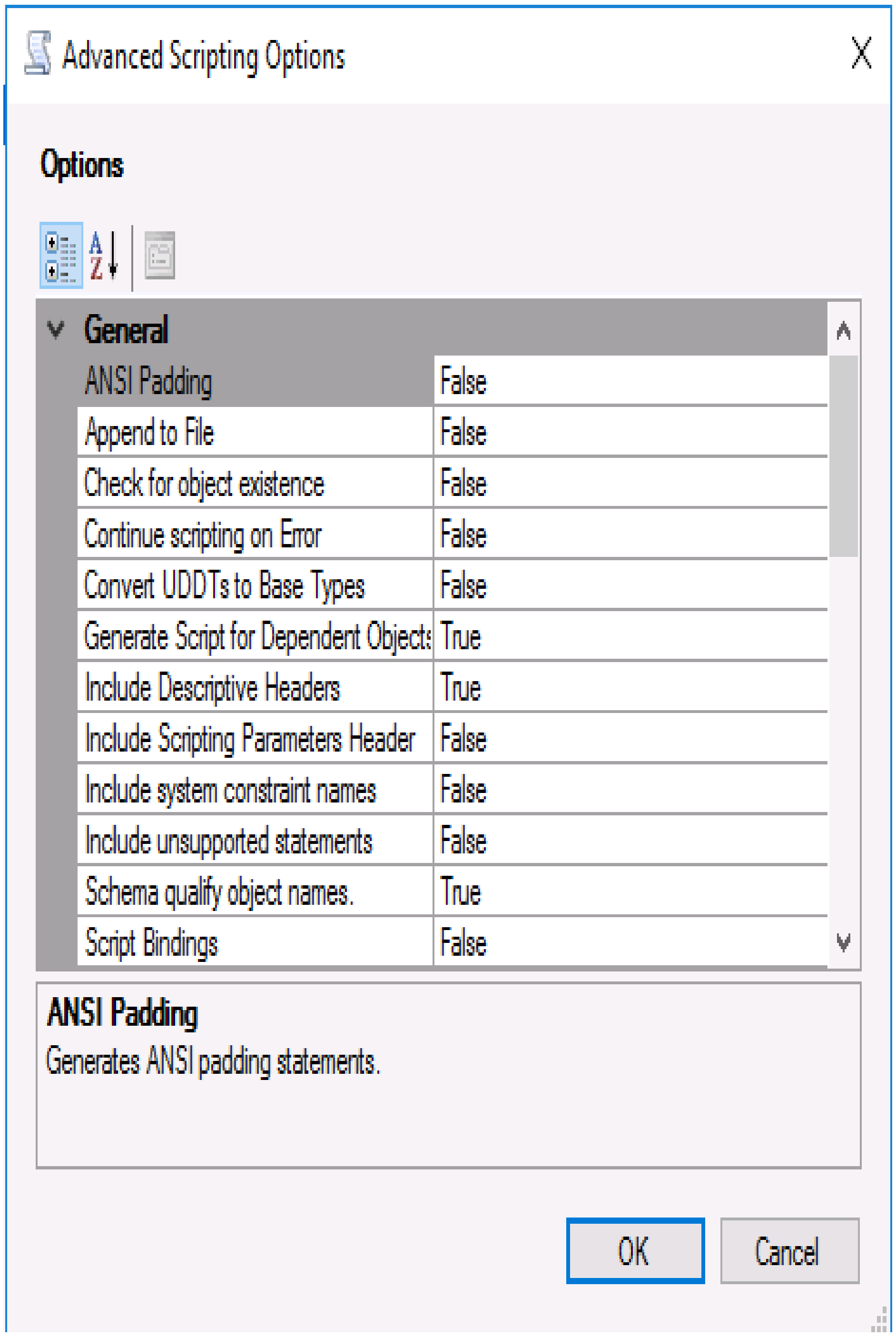
Files to generate:	(●) Single file ( ) Single file per object
File name:	C:\Users\Administrator\Documents\script.sql ☑ Overwrite existing file
Save as:	(●) Unicode text ( ) ANSI text

( ) Save to Clipboard

( ) Save to new query window

4. Click **Advanced**. In the displayed **Advanced Scripting Options** dialog box, specify scripting options for triggers, indexes, unique keys, the primary key, and server version. Then, click **OK**.

Figure 4-6 Specifying advanced scripting options



General

ANSI Padding	False
Append to File	False
Check for object existence	False
Continue scripting on Error	False
Convert UDDTs to Base Types	False
Generate Script for Dependent Objects	True
Include Descriptive Headers	True
Include Scripting Parameters Header	False
Include system constraint names	False
Include unsupported statements	False
Schema qualify object names.	True
Script Bindings	False

ANSI Padding
Generates ANSI padding statements.

NOTE

Generate Script for Dependent Objects indicates the script data type option.

5. Click **Next** to generate the script.

Step 10 Use the SSMS client to connect to the RDS DB instance and open the generated SQL script.

NOTE

You need to create an empty database, and then use the script to create structures in the database.

Step 11 Use the import and export function provided by Microsoft SQL Server to migrate data.

1. Right-click the database where data is to be imported and choose **Tasks** > **Import Data**.
2. Click **Next**.
3. On the **Choose a Data Source** page, select a data source and click **Next**.
4. On the **Choose a Destination** page, select a destination database and click **Next**.
   - **Destination**: Select **SQL Server Native Client** (depending on your destination database type).
   - **Server name**: Enter the IP address and port number of the destination DB instance.
   - **Authentication**: Select **Use SQL Server Authentication**. Then, set **User name** to **rdsuser**, and **Password** to the password of **rdsuser**.
   - **Database**: Select the destination database where data is to be imported.
5. Select **Copy data from one or more tables or views** and click **Next**.
6. On the **Select Source Tables and Views** page, select the tables and views that you want to copy. Then, click **Edit Mappings**. In the displayed dialog box, select **Enable identity insert** and edit mappings based on your requirements.
7. Click **Next**.
8. Select **Run immediately** and click **Next**.
9. Click **Finish** to import data. You can view the progress. About 4,000 rows can be processed per second.

----End

4.6 Creating a Subaccount of rdsuser

Scenarios

This section describes how to create a subaccount and grant permissions to the subaccount. **Permissions of rdsuser** lists the permissions supported by **rdsuser**.

Prerequisites

You have created a database. For details, see **Creating a Database**.

Procedure

Step 1 Log in to the instance through DAS.

1. **Log in to the management console**.

2. Click in the upper left corner and select a region and a project.
3. Click in the upper left corner of the page and choose **Databases** > **Relational Database Service**.
4. Locate the target instance and click **Log In** in the **Operation** column.
5. On the displayed page, configure required parameters.

Table 4-2 Instance login

Parameter	Description
Login Username	Enter **rdsuser**.
Password	Enter the password of **rdsuser**. **NOTE** You can select **Remember Password** so that you can directly log in to the instance next time.
Collect Metadata Periodically	Enable this function as required. If this function is enabled: – DAS can store structure definition data such as database names, table names, and field names in instances, but does not store data in tables. – Metadata is collected in the early morning every day.
Show Executed SQL Statements	Enable this function as required. This function allows you to view executed SQL statements. You can re-execute an SQL statement without having to enter it again.

6. Click **Log In**.

Step 2 Create a subaccount.

1. On the main menu of the DAS console, choose **Account Management** > **Login Name**.
2. On the displayed page, click **Create Login Name**.
3. On the **Create Login Name** page, configure login information.

Table 4-3 Login information

Parameter	Description
Login Name	Enter a new login name.
Authentication Type	The value is fixed to **Microsoft SQL Server Authentication**.

Parameter	Description
Password	The password for the new login username must meet the following requirements: – It must contain at least three of the following: uppercase letters, lowercase letters, digits, and special characters ~!@#$^*-_=+?% – It must be 8 to 128 characters long. – It cannot contain the login username. – It cannot be a weak password.
Confirm Password	Enter the password again. **NOTE** For security purposes, select **Enforce Password Policy**.
Default Database	From the drop-down list, select a database the new login user will log in by default.
Default Language	Select a language for the new login user.

4. Click **Save**.
5. Click **Back to Login Name List**.
6. In the login name list, view the new login name.

Step 3 Grant permissions to the new login user.

NOTE

- **Table 4-4** describes how to add a single permission. To add multiple permissions to the new login user at the same time, for example, to grant both read and write permissions, select both **db_datareader** and **db_datawriter** on the **Edit Database Role** page.
- For details about the permissions supported by **rdsuser**, see **Permissions of rdsuser**.

Table 4-4 Permissions that can be granted to a subaccount

Permission	Procedure
Database operation permission	1. Locate the new login name and click **Edit** in the **Operation** column. 2. On the displayed page, click the **User Mapping** tab. 3. In the **Users mapped to this login** list, click **Edit** in the row where both the user database and the new login username exist. 4. On the **Edit Database Role** page, select **db_owner** and click **OK**. 5. Click **Save**.

Permission	Procedure
Server role permission	1. Locate the new login name and click **Edit** in the **Operation** column. 2. On the displayed page, click the **Server Roles** tab. 3. In the **Server Roles** list, select the desired server role. 4. Click **Save**.
Securable object permission	1. Locate the new login name and click **Edit** in the **Operation** column. 2. On the displayed page, click the **Securables** tab. 3. In the securables list, select the desired server permission. 4. Click **Save**.
Read-only permission	1. Locate the new login name and click **Edit** in the **Operation** column. 2. On the displayed page, click the **User Mapping** tab. 3. In the **Users mapped to this login** list, click **Edit** in the row where both the user database and the new login username exist. 4. On the **Edit Database Role** page, select **db_datareader** and click **OK**. 5. Click **Save**.
Write permission	1. Locate the new login name and click **Edit** in the **Operation** column. 2. On the displayed page, click the **User Mapping** tab. 3. In the **Users mapped to this login** list, click **Edit** in the row where both the user database and the new login username exist. 4. On the **Edit Database Role** page, select **db_datawriter** and click **OK**. 5. Click **Save**.

----End

Permissions of rdsuser

Table 4-5 Permissions of rdsuser

Name	Category	Permission
DB instance permissions	DB instance role permissions	[processadmin]
		[setupadmin]
	DB instance object permissions	ALTER ANY CONNECTION
		ALTER ANY LOGIN
		ALTER ANY SERVER ROLE
		ALTER SERVER STATE
		ALTER TRACE
		CONNECT ANY DATABASE
		CONTROL SERVER
		CONNECT SQL
		CREATE ANY DATABASE
		SELECT ALL USER SECURABLES
		VIEW ANY DEFINITION
		VIEW ANY DATABASE
		VIEW SERVER STATE
	Database permissions	master: Public
		Msdb: Public SQLAgentUserRole
		Model: Public
		Rdsadmin: Public
		OtherDB: Db_Owner

4.7 Creating tempdb Files

Scenarios

The tempdb system database is a global resource that is available to all users connected to an instance of SQL Server or SQL Database. It is a temporary database that cannot store data permanently. It is used to process intermediate data for various requests in the instance. Physical properties of tempdb in SQL

Server are classified into the primary data files (.mdf), secondary data files (.ndf), and log files (.ldf). **tempdb** is re-created every time SQL Server is started.

There may be some issues or even service interruption if applications frequently create and drop tempdb files, especially in high-concurrency scenarios.

Microsoft recommends that the tempdb files be divided into multiple files. Generally, the number of files depends on the number of vCPUs (logical). If the number of vCPUs is greater than eight, use eight data files and then if contention continues, increase the number of data files by multiples of 4 until the contention is reduced to acceptable levels or make changes to the workload/code.

For more information, see **tempdb Database** in the Microsoft official website.

Constraints

- By default, each RDS for SQL Server instance running SQL Server 2008, 2012, or 2014 Edition has one tempdb file, each instance running SQL Server 2016 Edition has four tempdb files, and each instance running SQL Server 2017 Edition has eight tempdb files.
- Each RDS for SQL Server instance has only one log file no matter which SQL Server Edition they run.

Application Scenario

You need to determine the number of tempdb files to be created based on the instance specifications and scenarios. The following uses an example to show how to create 8 tempdb files for a SQL Server 2014 Enterprise Edition instance with 32 vCPUs.

Prerequisites

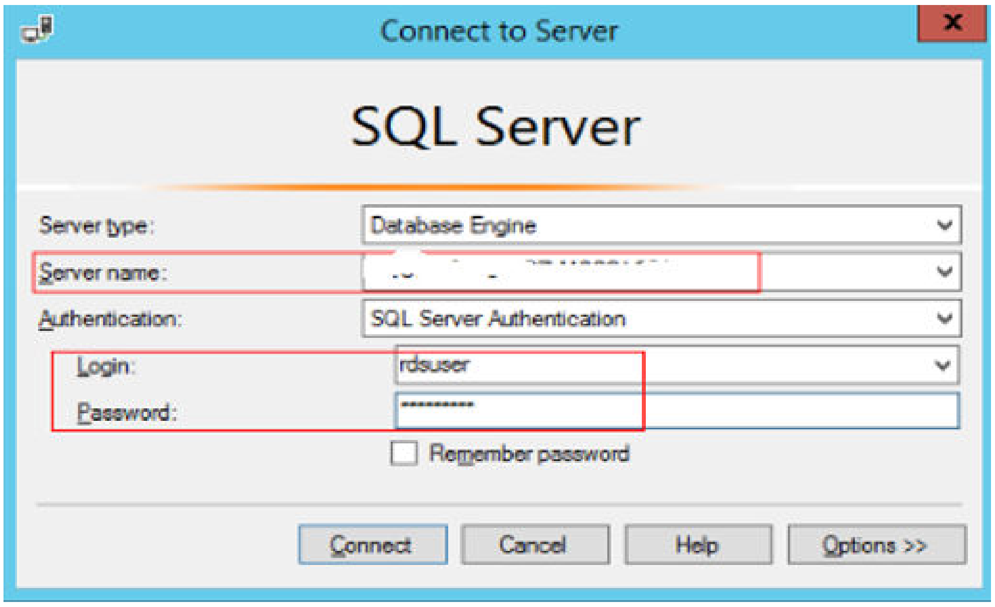
- You have installed the SQL Server Management Studio client. For details, see **How Can I Install SQL Server Management Studio?**
- You have created an instance with 32 vCPUs running Microsoft SQL Server 2014 Enterprise Edition. For details, see **Buy a DB Instance**

Procedure

Step 1 Start SQL Server Management Studio.

Step 2 Choose **Connect** > **Database Engine**. In the displayed dialog box, enter login information.

Figure 4-7 Connecting to the server



Server type: Database Engine
Server name:
Authentication: SQL Server Authentication
Login: rdsuser
Password: **********
☐ Remember password

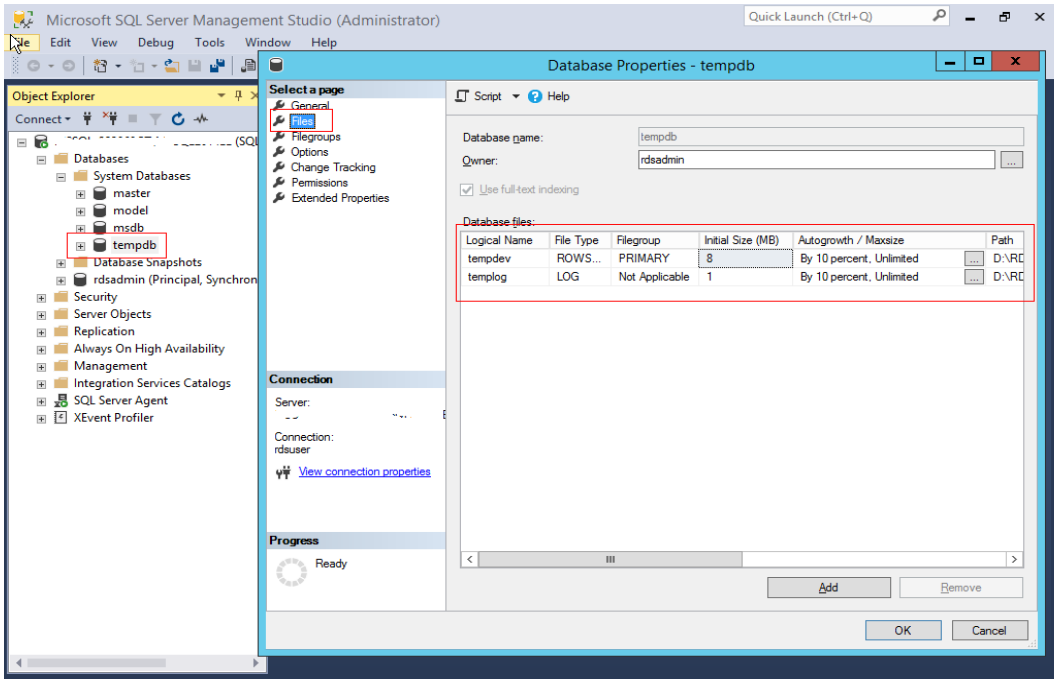
Table 4-6 Parameter description

Parameter	Description
Server name	Indicates the IP address and port of the DB instance. Use a comma (,) to separate them. For example: x.x.x.x,8080. ● The IP address is the EIP that has been bound to the DB instance. ● The port is the database port in the **Connection Information** area on the **Basic Information** page of the DB instance on the RDS console.
Authentication	Indicates the authentication mode. Select **SQL Server Authentication**.
Login	Indicates the RDS database username. The default administrator is **rdsuser**.
Password	Indicates the password of the RDS database username.

Step 3 View the tempdb information.

- Choose **Databases** > **System Databases**> **tempdb**. Right-click **tempdb** and choose **Database Properties**. In the displayed dialog box, view the current tempdb information.

Figure 4-8 Viewing current tempdb information



Database Properties - tempdb

Database name: tempdb

Owner: rdsadmin

Database files:

Logical Name	File Type	Filegroup	Initial Size (MB)	Autogrowth / Maxsize	Path
tempdev	ROWS...	PRIMARY	8	By 10 percent, Unlimited	D:\RD
templog	LOG	Not Applicable	1	By 10 percent, Unlimited	D:\RD

- You can also run the following SQL statements to query the tempdb information:

  SELECT name AS FileName,

  size*1.0/128 AS FileSizeInMB,

  CASE max_size

  WHEN 0 THEN 'Autogrowth is off.'

  WHEN -1 THEN 'Autogrowth is on.'

  ELSE 'Log file grows to a maximum size of 2 TB.'

  END,

  growth AS 'GrowthValue',

  'GrowthIncrement' =

  CASE

  WHEN growth = 0 THEN 'Size is fixed.'

  WHEN growth > 0 AND is_percent_growth = 0

  THEN 'Growth value is in 8-KB pages.'

  ELSE 'Growth value is a percentage.'

  END

  FROM tempdb.sys.database_files;

  GO

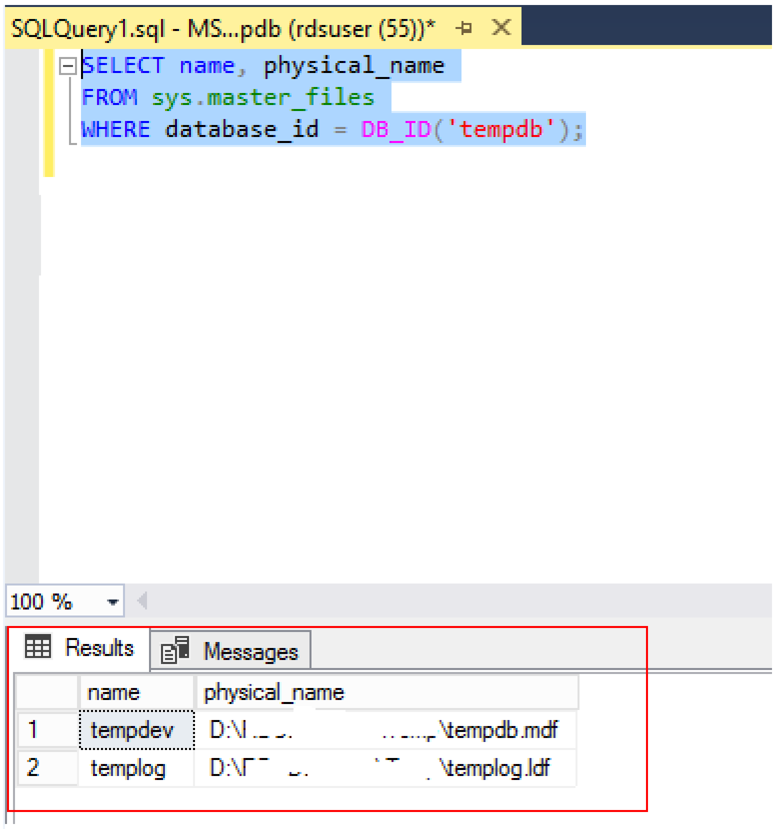
Step 4 Run the following statements to query the tempdb file name of the current DB instance:

SELECT name, physical_name

FROM sys.master_files

WHERE database_id = DB_ID('tempdb');

Figure 4-9 Viewing tempdb file names

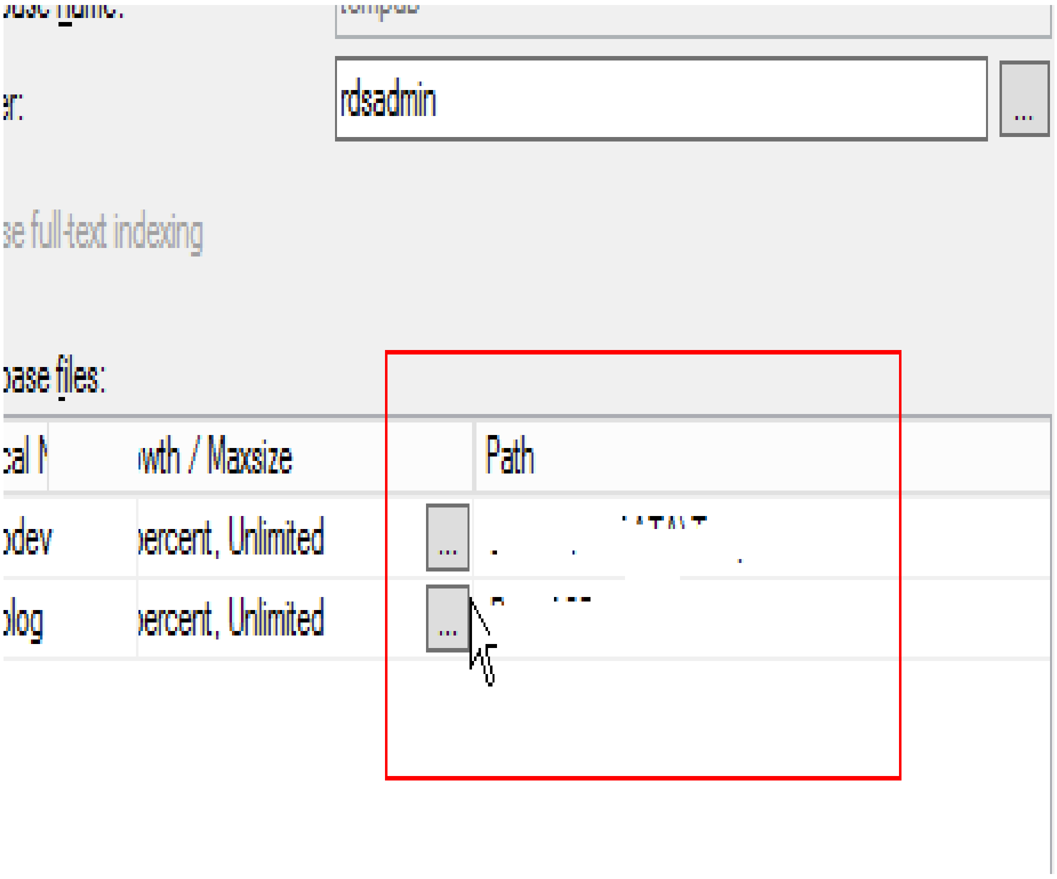


```
SELECT name, physical_name
FROM sys.master_files
WHERE database_id = DB_ID('tempdb');
```

```
   name      physical_name
1  tempdev   D:\[illegible]\tempdb.mdf
2  templog   D:\[illegible]\templog.ldf
```

Step 5 On the **Files** tab in **Step 3**, view the paths of tempdb files.

Figure 4-10 Viewing tempdb paths



Step 6 Run the following statements to migrate the tempdb files to **D:\RDSDBDATA\DATA** and specify the initial size and growth as required.

USE master;

GO

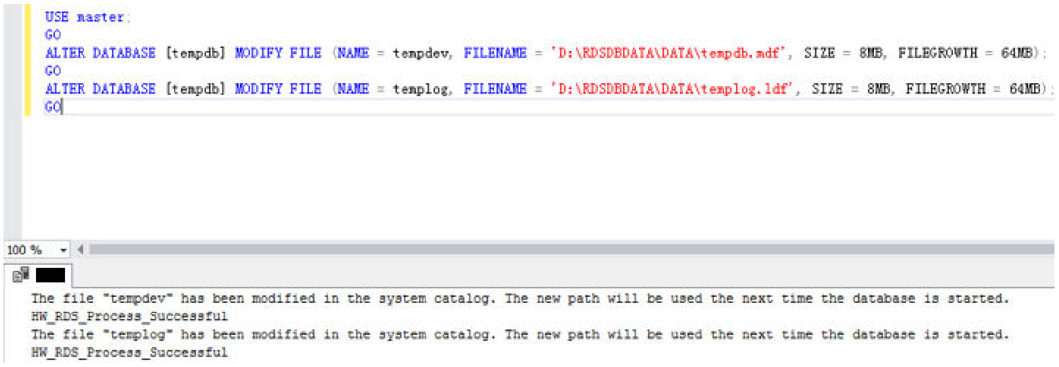
ALTER DATABASE [tempdb] MODIFY FILE (NAME = tempdev, FILENAME = 'D:\RDSDBDATA\DATA\tempdb.mdf', SIZE = 8MB, FILEGROWTH = 64MB);

GO

ALTER DATABASE [tempdb] MODIFY FILE (NAME = templog, FILENAME = 'D:\RDSDBDATA\DATA\templog.ldf', SIZE = 8MB, FILEGROWTH = 64MB);

GO

Figure 4-11 Moving tempdb files



```
USE master;
GO
ALTER DATABASE [tempdb] MODIFY FILE (NAME = tempdev, FILENAME = 'D:\RDSDBDATA\DATA\tempdb.mdf', SIZE = 8MB, FILEGROWTH = 64MB);
GO
ALTER DATABASE [tempdb] MODIFY FILE (NAME = templog, FILENAME = 'D:\RDSDBDATA\DATA\templog.ldf', SIZE = 8MB, FILEGROWTH = 64MB);
GO
```

[redacted]

```
The file "tempdev" has been modified in the system catalog. The new path will be used the next time the database is started.
HW_RDS_Process_Successful
The file "templog" has been modified in the system catalog. The new path will be used the next time the database is started.
HW_RDS_Process_Successful
```

Step 7 On the **Instances** page of the RDS console, locate the target DB instance and choose **More** > **Reboot** in the **Operation** column to reboot the DB instance.

You can also click the target DB instance. On the displayed page, click **Reboot** in the upper right corner of the page.

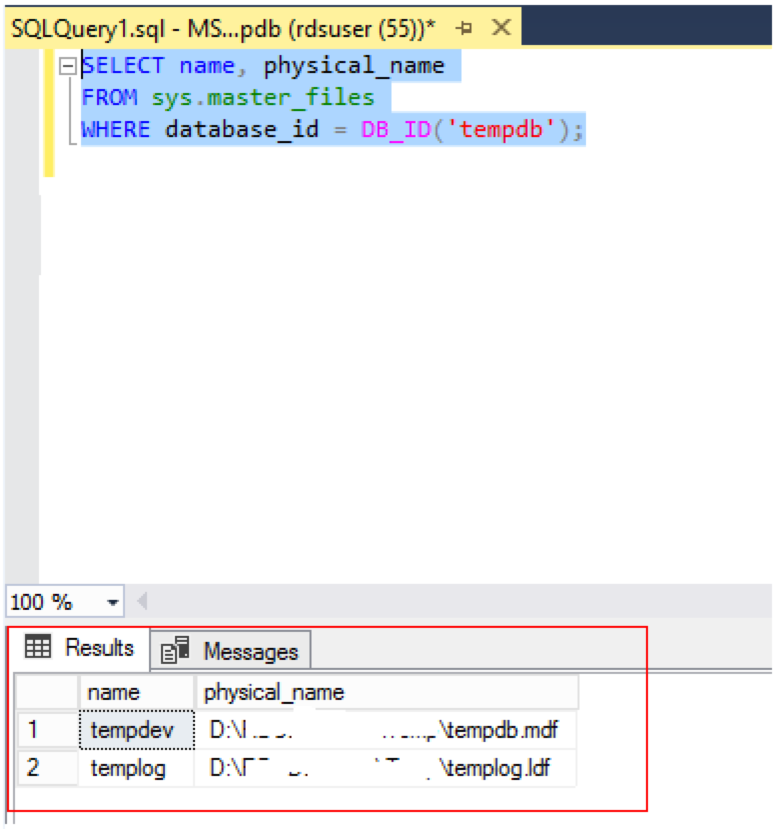
Step 8 Run the following SQL statements to check whether the tempdb files are successfully migrated:

SELECT name, physical_name

FROM sys.master_files

WHERE database_id = DB_ID('tempdb');

Figure 4-12 Viewing the migration results



```
SELECT name, physical_name
FROM sys.master_files
WHERE database_id = DB_ID('tempdb');
```

	name	physical_name
1	tempdev	D:\[illegible]\tempdb.mdf
2	templog	D:\[illegible]\templog.ldf

Step 9 Configure the file name, initial size, and growth as required. Add tempdb files using either of the following methods:

- Adding tempdb files through SQL statements

  Based on the number of vCPUs and tempdb files to be added, set the initial size to 8 MB and file growth to 64 MB.

  USE [master]

  GO

  ALTER DATABASE [tempdb] ADD FILE (NAME = N'temp2', FILENAME = N'D:\RDSDBDATA\DATA\tempdb2.ndf', SIZE = 8MB, FILEGROWTH = 64MB)

  GO

  ALTER DATABASE [tempdb] ADD FILE (NAME = N'temp3', FILENAME = N'D:\RDSDBDATA\DATA\tempdb3.ndf', SIZE = 8MB, FILEGROWTH = 64MB)

  GO

  ALTER DATABASE [tempdb] ADD FILE (NAME = N'temp4', FILENAME = N'D:\RDSDBDATA\DATA\tempdb4.ndf', SIZE = 8MB, FILEGROWTH = 64MB)

  GO

  ALTER DATABASE [tempdb] ADD FILE (NAME = N'temp5', FILENAME = N'D:\RDSDBDATA\DATA\tempdb5.ndf', SIZE = 8MB, FILEGROWTH = 64MB)

  GO

  ALTER DATABASE [tempdb] ADD FILE (NAME = N'temp6', FILENAME = N'D:\RDSDBDATA\DATA\tempdb6.ndf', SIZE = 8MB, FILEGROWTH = 64MB)

  GO

  ALTER DATABASE [tempdb] ADD FILE (NAME = N'temp7', FILENAME = N'D:\RDSDBDATA\DATA\tempdb7.ndf', SIZE = 8MB, FILEGROWTH = 64MB)

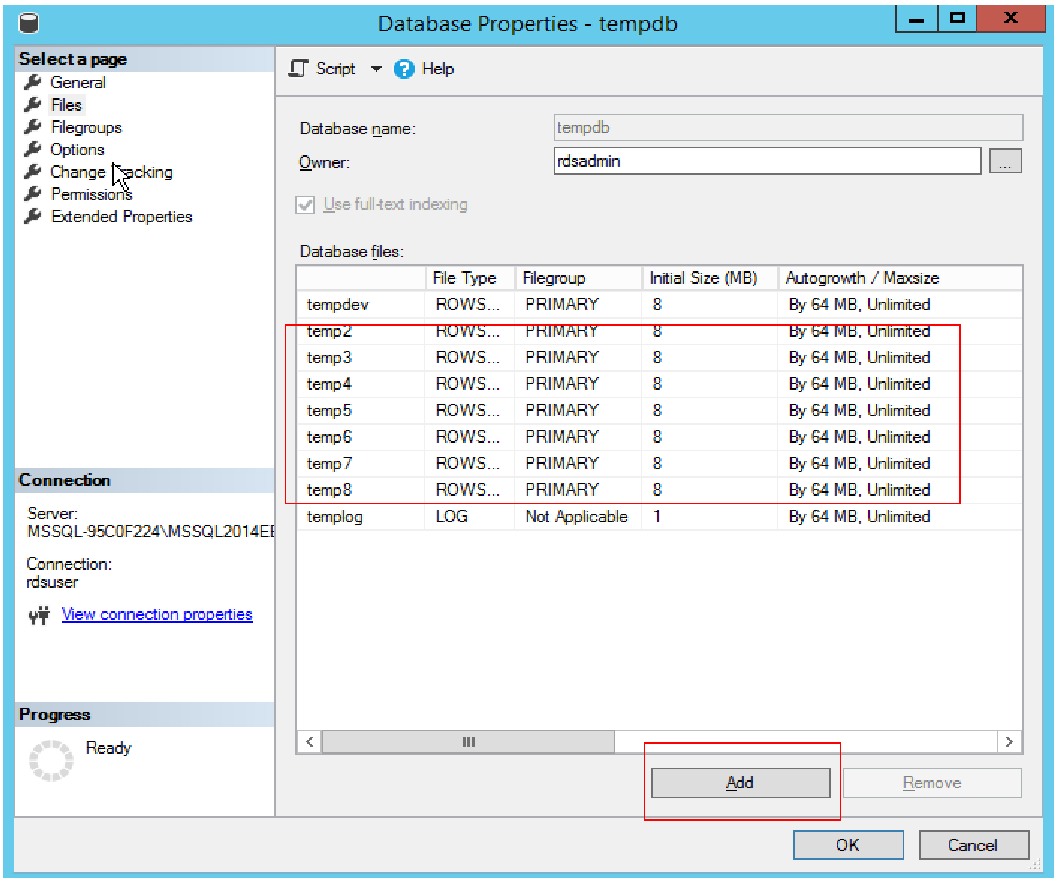
  GO

  ALTER DATABASE [tempdb] ADD FILE (NAME = N'temp8', FILENAME = N'D:\RDSDBDATA\DATA\tempdb8.ndf', SIZE = 8MB, FILEGROWTH = 64MB)

  GO

- Adding tempdb files through SQL Server Management Studio On the **Files** tab in **Step 3**, click **Add**.

Figure 4-13 Adding tempdb files through the client



Database name: tempdb
Owner: rdsadmin
Use full-text indexing

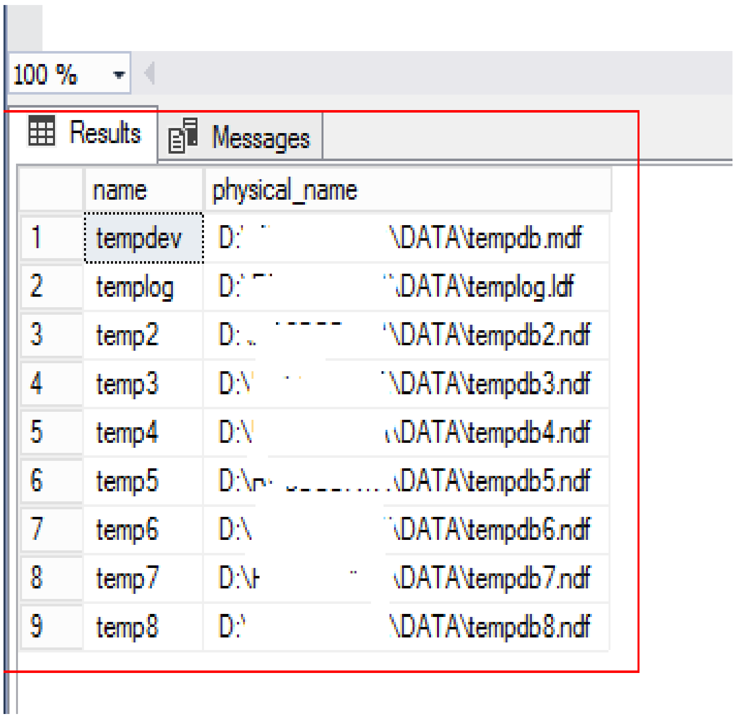
Database files:

	File Type	Filegroup	Initial Size (MB)	Autogrowth / Maxsize
tempdev	ROWS...	PRIMARY	8	By 64 MB, Unlimited
temp2	ROWS...	PRIMARY	8	By 64 MB, Unlimited
temp3	ROWS...	PRIMARY	8	By 64 MB, Unlimited
temp4	ROWS...	PRIMARY	8	By 64 MB, Unlimited
temp5	ROWS...	PRIMARY	8	By 64 MB, Unlimited
temp6	ROWS...	PRIMARY	8	By 64 MB, Unlimited
temp7	ROWS...	PRIMARY	8	By 64 MB, Unlimited
temp8	ROWS...	PRIMARY	8	By 64 MB, Unlimited
templog	LOG	Not Applicable	1	By 64 MB, Unlimited

Step 10 After the configuration is complete, reboot the DB instance again by referring to **Step 7**.

Step 11 Repeat **Step 8** to check whether the tempdb files are successfully added.

Figure 4-14 Viewing the added tempdb files



	name	physical_name
1	tempdev	D: \DATA\tempdb.mdf
2	templog	D: \DATA\templog.ldf
3	temp2	D: \DATA\tempdb2.ndf
4	temp3	D:\ \DATA\tempdb3.ndf
5	temp4	D:\ \DATA\tempdb4.ndf
6	temp5	D:\ \DATA\tempdb5.ndf
7	temp6	D:\ \DATA\tempdb6.ndf
8	temp7	D:\ \DATA\tempdb7.ndf
9	temp8	D: \DATA\tempdb8.ndf

----End

4.8 Microsoft SQL Server Publication and Subscription

The publication and subscription function provided by Microsoft SQL Server uses the replication technology for data synchronization. This function makes it possible to split data read and write operations and synchronize offline and online data.

This section describes how to use SQL Server Management Studio (SSMS) to configure publication and subscription. You can create publications and subscriptions for RDS for SQL Server instances on the console. For details, see **Creating a Publication**.

Preparations

Environments

1. Local environment: a local database running Microsoft SQL Server 2014 Standard Edition in Windows
2. Online environment:
   - One single DB instance with 2 vCPUs and 16 GB of memory running Microsoft SQL Server 2014 Standard Edition, bound with an EIP
   - One primary/standby DB instance with 4 vCPUs and 8 GB of memory running Microsoft SQL Server 2014 Standard Edition, bound with an EIP

Environment Setup

- Publisher: maintains source data and distributes specific data to the distributor. In this example, use the local server as the publisher.

a. Use SSMS to log in to the local database as user **sa**. Right-click the **Replication** folder and then click **Configure Distribution**. You can use your local server as the distributor or use another server as the distributor. Click **Next**.

NOTICE

- **sa** is the administrator account.
- The login account must have the **sysadmin** permission. Otherwise, publication and subscription cannot be configured.

b. Specify a root snapshot folder and click **Next**.

NOTE

Related agent permissions must be configured for the publication so that the agent account has the permissions to read from and write to the folder.

c. Specify the names and directories of the distribution database and log files, and click **Next**.

d. Specify the distributor for the publisher and then click **Next**.

e. Click **Finish** to complete the configuration.

- Configuring the agent account control file

  a. You need to add the agent account to the control property of the snapshot folder according to the folder directory. Otherwise, an error message will be displayed, indicating that the access is denied.

  b. Open the local SQL Server Configuration Manager, right-click the corresponding agent, choose **Properties** from the shortcut menu, and copy the account name.

  c. Return to the directory of the snapshot folder. Right-click the folder and choose **Properties** from the shortcut menu. In the displayed dialog box, choose **Security** > **Edit** > **Add**. Select the local path and agent account name, click **OK**, and select all permissions.

- Distributor: distributes data to specific subscribers. In this example, the distributor and publisher share the same server. Therefore, no extra configuration is required. For more information, see **Configure Distribution** at the official website.
- Subscriber: receives data from the distributor. Subscription includes push subscription and pull subscription.
  - Push subscription: The publisher propagates changes to a subscriber without a request from the subscriber. Changes can be pushed to subscribers continuously or on a frequently recurring schedule.
  - Pull subscription: The subscriber requests changes made at the publisher. The data is usually synchronized on demand or on a schedule rather than continuously. RDS for SQL Server instances do not support pull subscription. In this example, only push subscription can be configured.

  Before the subscription, ensure that the RDS for SQL Server instance can be accessed from the local server.

  Before configuring the local subscription, you need to configure the RDS instance information on the local server.

a. Configure an alias name for the subscriber on the local server. The subscription service does not support IP address-based access. Therefore, you need to map the EIP of the RDS DB instance to an alias name. To obtain the alias name, log in to the RDS DB instance and run the following SQL statement:

```
select @@SERVERNAME
```

b. After obtaining the alias name, open the local SQL Server Configuration Manager, select the native clients, right-click **Aliases**, and choose **New Aliases** from the shortcut menu.

c. Enter related information and click **OK**.

Table 4-7 Parameter description

Parameter	Description
Alias Name	Alias name configured in **a**
Port No	Port number of the RDS instance
Server	EIP bound to the RDS instance

d. In **C:\Windows\System32\drivers\etc**, open the host file and add a mapping:

```
Server_address MSSQL-177FFD84\MSSQL2014STD
```

Creating a Publication

Step 1 Create a publication.

Expand the **Replication** folder, and then right-click the **Local Publications** folder. Click **New Publication**.

Step 2 Select **Transactional publication**.

Step 3 Select a table as the publication object.

Step 4 Add the object to be filtered for personalized publication.

Step 5 Create a snapshot to replicate the current state of the table. You can also set up a snapshot agent to execute the plan.

Step 6 Configure the agent security. You need to set the login account to the local **sa** account.

Step 7 Configure the publication name and click **Finish**.

Step 8 Check whether the publication is created by using the replication monitor.

----End

Creating a Subscription

Step 1 Right-click the publication for which you want to create one or more subscriptions, and then select **New Subscriptions**.

Step 2 Configure the required parameters and click **Next**.

Step 3 Select push subscription and click **Next**.

Step 4 Click **Add Subscriber**. Both the SQL Server engine and non-SQL Server engine can be used as subscribers. In this example, use the RDS for SQL Server instance as the subscriber.

Step 5 Select a database as the subscription object.

Step 6 Configure the connection to the subscriber.

Step 7 Use a database account that is valid for a long time to ensure the subscription validity. You can use the account for logging in to the RDS for SQL Server instance. Then, click **OK**.

Step 8 Check whether the subscription is created successfully.

Step 9 Move the cursor to the publication to view the subscription information.

----End

4.9 Installing a C# CLR Assembly in RDS for SQL Server

Microsoft SQL Server provides assemblies to make database operations simple and convenient.

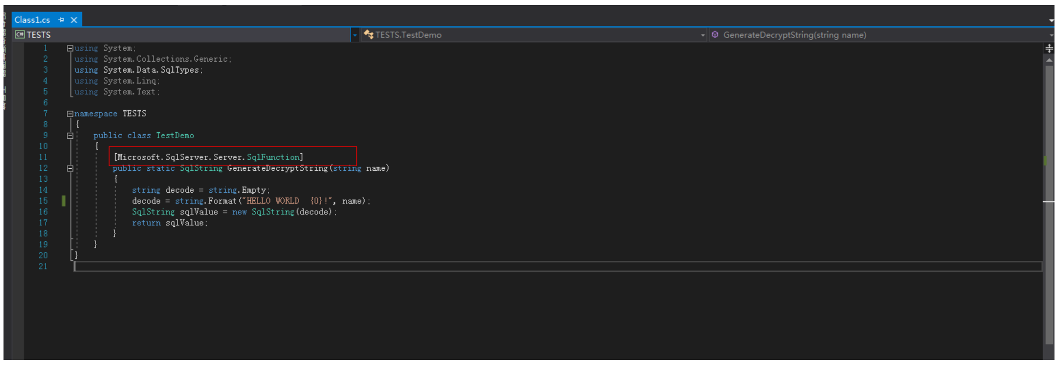
NOTE

When you restore data to a new or an existing DB instance, the **clr enabled** parameter is disabled by default. To use the CLR integration function, you need to enable **clr enabled** first. For details about how to enable the CLR integration function, see **Enabling CLR Integration**.

Procedure

Step 1 Create a C# function to compile an SQL Server DLL.

Figure 4-15 C# function code



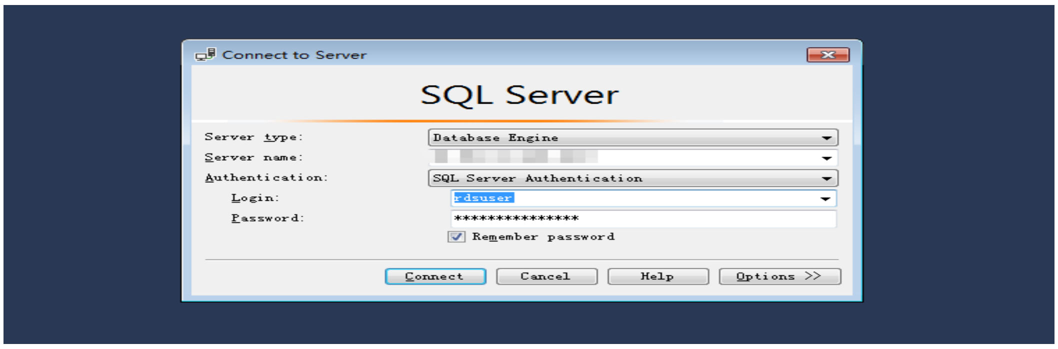
```
using System;
using System.Collections.Generic;
using System.Data.SqlTypes;
using System.Linq;
using System.Text;

namespace TESTS
{
    public class TestDemo
    {
        [Microsoft.SqlServer.Server.SqlFunction]
        public static SqlString GenerateDecryptString(string name)
        {
            string decode = string.Empty;
            decode = string.Format("HELLO WORLD  {0}!", name);
            SqlString sqlValue = new SqlString(decode);
            return sqlValue;
        }
    }
}
```

NOTICE

For more information about user-defined functions, see **CLR User-Defined Functions**.

Step 2 Use SQL Server Management Studio to connect to the database.

Figure 4-16 Connecting to the server



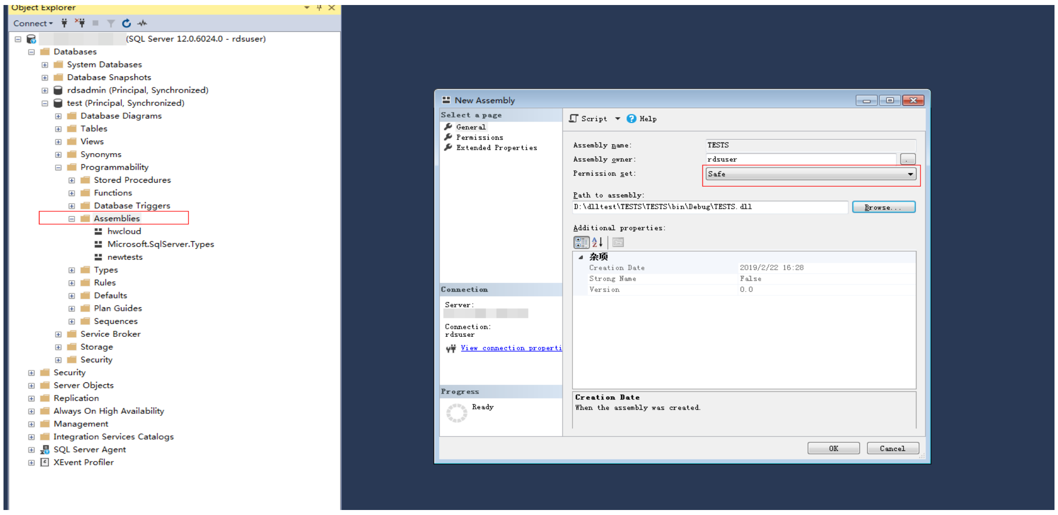
Server type: Database Engine
Server name: [redacted]
Authentication: SQL Server Authentication
Login: rdsuser
Password: ***************
[✓] Remember password

Step 3 Select the target database and create the corresponding assembly.

NOTE

- Only the SAFE assembly (**Permission** set is **Safe**) can be created.
- The DLL file is saved in the hexadecimal format, as shown in **Figure 4-18**.

Figure 4-17 Creating an assembly



```
[redacted] (SQL Server 12.0.6024.0 - rdsuser)
  Databases
    System Databases
    Database Snapshots
    rdsadmin (Principal, Synchronized)
    test (Principal, Synchronized)
      Database Diagrams
      Tables
      Views
      Synonyms
      Programmability
        Stored Procedures
        Functions
        Database Triggers
        Assemblies
          hwcloud
          Microsoft.SqlServer.Types
          newtests
        Types
        Rules
        Defaults
        Plan Guides
        Sequences
      Service Broker
      Storage
      Security
  Security
  Server Objects
  Replication
  Always On High Availability
  Management
  Integration Services Catalogs
  SQL Server Agent
  XEvent Profiler
```

Assembly name: TESTS
Assembly owner: rdsuser
Permission set: Safe
Path to assembly: D:\dlltest\TESTS\TESTS\bin\Debug\TESTS.dll
Additional properties:

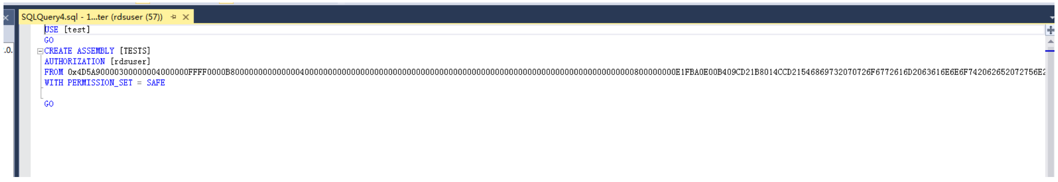
杂项	
Creation Date	2019/2/22 16:28
Strong Name	False
Version	0.0

Creation Date
When the assembly was created.

Connection
Server: [redacted]
Connection: rdsuser
View connection properti

Progress
Ready

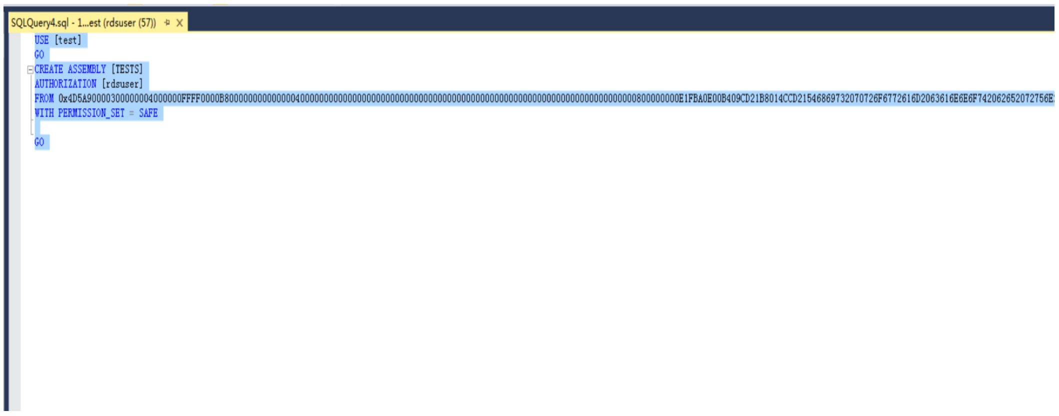
Figure 4-18 DLL file



```
USE [test]
GO
CREATE ASSEMBLY [TESTS]
AUTHORIZATION [rdsuser]
FROM 0x4D5A90000300000004000000FFFF0000B800000000000000400000000000000000000000000000000000000000000000000000000000000000000000000000000800000000E1FBA0E00B409CD21B8014CCD21546869732070726F6772616D2063616E6E6F742062652072756E2
WITH PERMISSION_SET = SAFE

GO
```

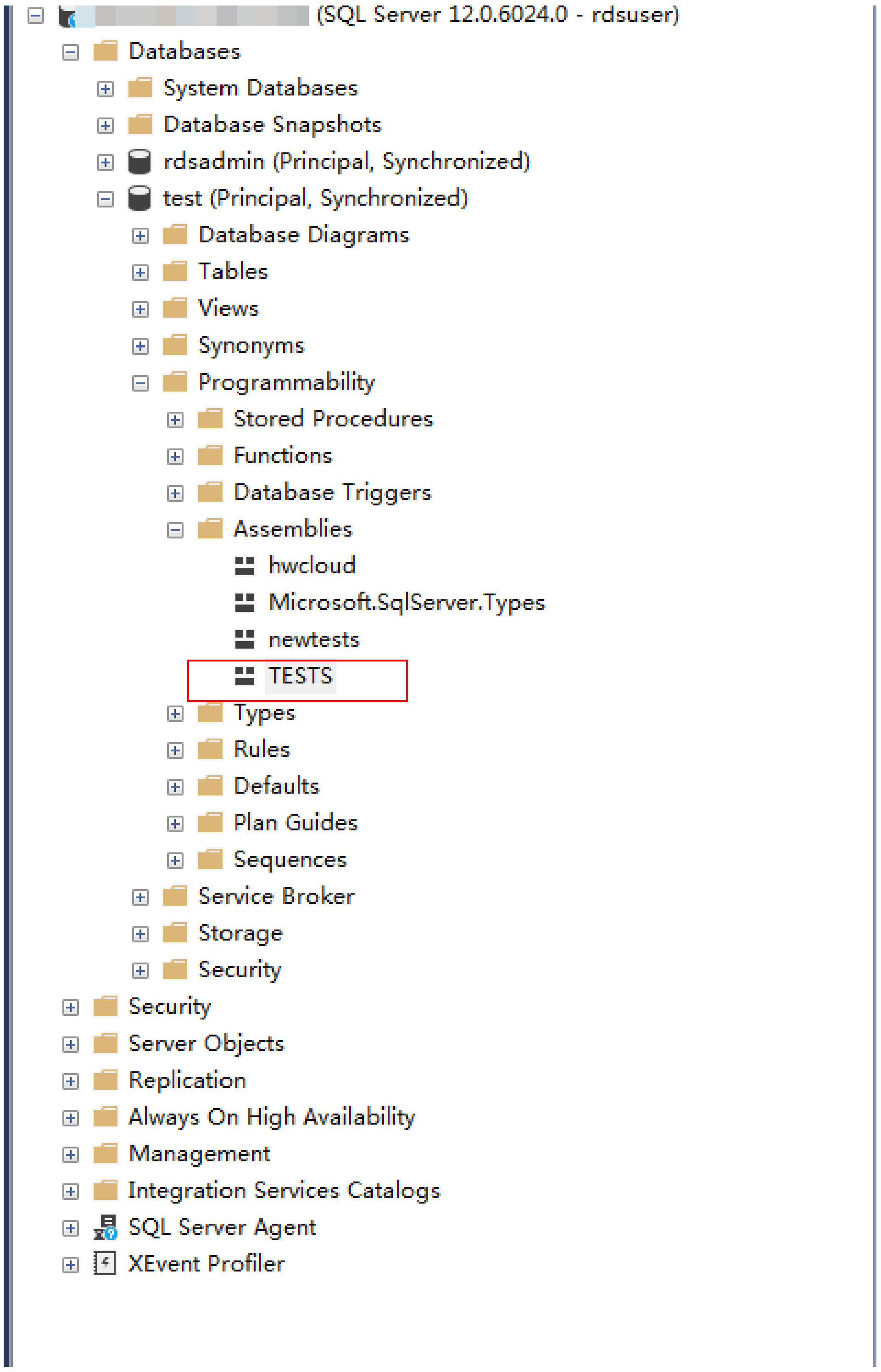
Step 4 Execute the program. If the execution result is shown as **Figure 4-19**, the execution is successful. The TESTS assembly is added, as shown in **Figure 4-20**.

Figure 4-19 Execution result



```
USE [test]
GO
CREATE ASSEMBLY [TESTS]
AUTHORIZATION [rdsuser]
FROM 0x4D5A90000300000004000000FFFF0000B800000000000000400000000000000000000000000000000000000000000000000000000000000000000000000000000080000000 0E1FBA0E00B409CD21B8014CCD21546869732070726F6772616D2063616E6E6F742062652072756E
WITH PERMISSION_SET = SAFE

GO
```

Figure 4-20 TESTS assembly



```
[redacted] (SQL Server 12.0.6024.0 - rdsuser)
  Databases
    System Databases
    Database Snapshots
    rdsadmin (Principal, Synchronized)
    test (Principal, Synchronized)
      Database Diagrams
      Tables
      Views
      Synonyms
      Programmability
        Stored Procedures
        Functions
        Database Triggers
        Assemblies
          hwcloud
          Microsoft.SqlServer.Types
          newtests
          TESTS
        Types
        Rules
        Defaults
        Plan Guides
        Sequences
      Service Broker
      Storage
      Security
  Security
  Server Objects
  Replication
  Always On High Availability
  Management
  Integration Services Catalogs
  SQL Server Agent
  XEvent Profiler
```

----End

4.10 Creating a Linked Server for an RDS for SQL Server DB Instance

Create a linked server for SQL Server DB instance named 2 to access another SQL Server DB instance named 1.

Step 1 Enable distributed transactions of the two DB instances by referring to **Distributed Transactions** and add the peer-end host information to each other. For offline servers or ECS servers, **Name Resolution on Remote Servers (ECSs)**.

NOTE

If two DB instances 1 and 2 are in the same VPC, use the floating IP address. If the ECS server and RDS DB instances are not in the same VPC or a DB instance is offline, use an EIP. For details on how to bind an EIP to a DB instance, see **Binding and Unbinding an EIP**.

Step 2 In DB instance 1, create database dbtest1 as user **rdsuser**.

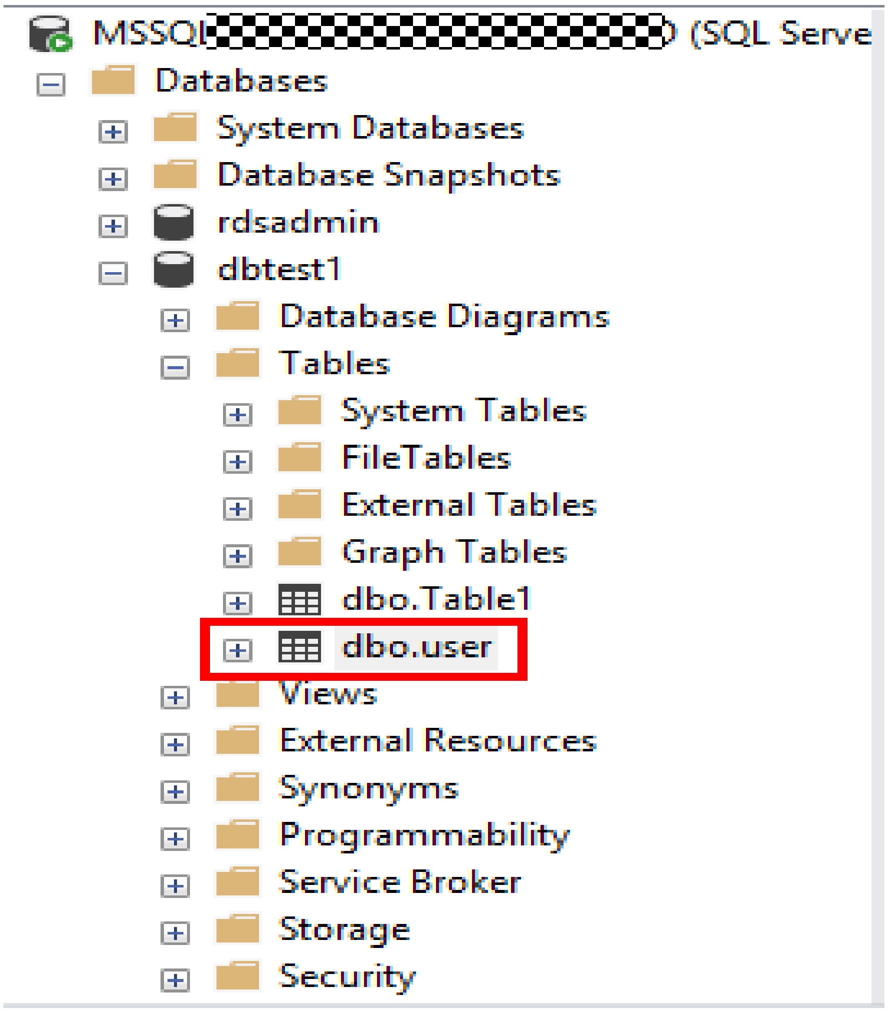
Step 3 In DB instance 2, run the following commands to create a linked server as user **rdsuser**.

```
USE [master]

GO

EXEC master.dbo.sp_addlinkedserver @server = N'TEST',
@srvproduct=N'mytest', @provider=N'SQLOLEDB',
@datasrc=N'192.168.***.***,1433'

EXEC master.dbo.sp_addlinkedsrvlogin @rmtsrvname = N'TEST', @locallogin =
NULL , @useself = N'False', @rmtuser = N'rdsuser', @rmtpassword = N'********'

GO
```

Table 4-8 Parameter description

Parameter	Description
@server	Specifies the linked server name.
@srvproduct	Specifies the product name.
@provider	Use the default value.
@datasrc	Specifies the IP address and port of the DB instance to be accessed.
@rmtsrvname	Specifies the name for logging in to the linked server.
@rmtuser	Specifies the username (**rdsuser**).
@rmtpassword	Specifies the user password.

Step 4 After the DBLink is created, you can view the databases created in DB instance 1 on the linked server.



```
MSSQL[redacted] (SQL Serve
⊟ Databases
    ⊞ System Databases
    ⊞ Database Snapshots
    ⊞ rdsadmin
    ⊟ dbtest1
        ⊞ Database Diagrams
        ⊟ Tables
            ⊞ System Tables
            ⊞ FileTables
            ⊞ External Tables
            ⊞ Graph Tables
            ⊞ dbo.Table1
            ⊞ dbo.user
        ⊞ Views
        ⊞ External Resources
        ⊞ Synonyms
        ⊞ Programmability
        ⊞ Service Broker
        ⊞ Storage
        ⊞ Security
```

Step 5 Run the following commands to check whether the data is successfully inserted, as shown in **Figure 4-21**:

begin tran

set xact_abort on

INSERT INTO [LYNTEST].[dbtest1].[dbo].[user1]

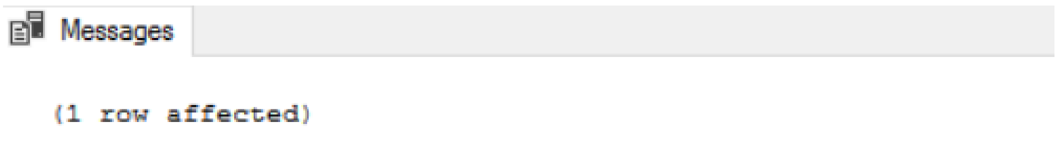
([id],[lname],[rname])

VALUES('19','w','x')

GO

commit tran

Figure 4-21 Insert result



```
Messages

(1 row affected)
```

----End

4.11 Deploying SQL Server Reporting Services (SSRS) on RDS for SQL Server

You can use SSRS to make various simple or complex reports. In addition, RDS provides the subscription function for you to subscribe to reports. This section describes how to deploy SSRS on RDS for SQL Server.

Scenarios

Microsoft SQL Server contains server components such as the SQL Server database engine, SSRS, and SQL Server Analysis Services (SSAS). The SQL Server database engine is a standard relational database component. RDS for SQL Server is a PaaS service that provides this database engine. However, other components, such as SSRS, are not provided as PaaS services on Huawei Cloud. To use SSRS on Huawei Cloud, you need to create a Windows-based ECS before installing and configuring SSRS.

SSRS has been separated from the Microsoft SQL Server component package and become an independent component service since SQL Server 2017. To migrate SSRS to the cloud, download the component from the Microsoft official website, install it on a Windows-based ECS, and use RDS for SQL Server as the backend database.

Prerequisites

- You have **created an RDS for SQL Server instance**
- You have created a Windows-based ECS. (The ECS and RDS DB instance must be in the same VPC, security group, and subnet.)

Procedure

Step 1 Download **SSRS** and install it on the ECS.

Step 2 After the installation is complete, click **Configure Report Server**.

Step 3 In Report Server Configuration Manager, configure **Server Name** and click **Connect**.

Step 4 In the navigation pane on the left, click **Service Account** and **Web Service URL** and configure parameters based on your service requirements.

NOTE

For details, see the **official documentation**.

Step 5 Configure the report server.

1. In the navigation pane on the left, click **Database**. On the right of the page, click **Change Database** to create a report server database on the ECS.
2. In the displayed dialog box, select **Create a new report server database** and click **Next**.

If a local report database is available, you can use Data Replication Service (DRS) to **migrate the full backup files** of the local report database to the RDS for SQL Server instance.

3. Configure the connection information of the RDS for SQL Server instance. Set **Server Name** to the RDS for SQL Server instance address in the format of *IP address,port*. Use a comma (,) to separate the IP address and port. Set **Username** to **rdsuser**. Click **Test Connection**. After the connection test is successful, click **Next**.
4. Enter the database name, select a language for the script, and then click **Next**.
5. Configure the credentials for the account **rdsuser** to connect to the report server and click **Next**.
6. Confirm the report server information and click **Next**.
7. After the configuration is successful, click **Finish**.

NOTE

For details, see the **official documentation**.

Step 6 In the navigation pane on the left, click **Web Portal URL** and click **Apply**. After the operation is complete, click the URL to access the web page of the report server.

Step 7 In the upper right corner, choose **New** > **Data Source**.

Step 8 Configure the parameters as follows:

Table 4-9 Parameter description

Category	Parameter	Description
Properties	Name	Name of the data source. The name cannot contain the following characters: / @ $ & * + = < > : ' , ? | \
	Description	Description of the data source, which is used to identify different data sources.
	Hide	If this parameter is selected, the data source is hidden.
	Enable	If this parameter is selected, the data source is enabled.
Connections	Type	Type of the data source. Select **Microsoft SQL Server**.

Category	Parameter	Description
	Connection String	Domain name and database name of the RDS for SQL Server instance in the following format: Data Source=<*Floating IP address of the RDS for SQL Server instance*, *port of the RDS for SQL Server instance*>; Initial Catalog=<*Database name*>
Login	Data Source Login	Select **Use the following credentials**.
	Credential Type	Select **Database username and password**.
	Username	Account of the RDS for SQL Server instance
	Password	Password of the database account

Step 9 Click **Test Connection**. After the connection test is successful, click **Create**.

Step 10 After the data source is created, design reports using Report Builder or Visual Studio.

For details, see **Report Builder in SQL Server**.

----End

4.12 Shrinking an RDS for SQL Server Database

Scenarios

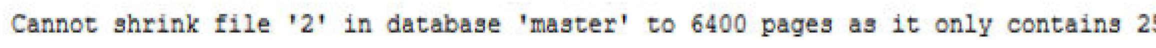
You can use a stored procedure to shrink the size of the data and log files in a specified RDS for SQL Server database.

Prerequisites

An RDS for SQL Server DB instance has been connected. Connect to the DB instance through the Microsoft SQL Server client. For details, see **Connecting to a DB Instance Through a Public Network**.

Constraints

- The database can be shrunk only when the database file size exceeds 50 MB. Otherwise, the following message is displayed:



```
Cannot shrink file '2' in database 'master' to 6400 pages as it only contains 2
```

- Transactions running at the row version control-based isolation level may prevent shrinking operations. To solve this problem, perform the following steps:

a. Terminate the transactions that prevent shrinking.
b. Terminate the shrinking operation. All completed tasks will be retained.
c. Do not perform any operations and wait until the blocking transactions are complete.

Best Practices

When you plan to shrink a database, consider the following:

- A shrink operation is most effective after an operation that creates lots of unused space, such as a database reboot.
- Most databases require some free space to be available for regular day-to-day operations. If you shrink a database repeatedly and notice that the database size grows again, this indicates that the space that was shrunk is required for regular operations. In these cases, repeatedly shrinking the database is a wasted operation.
- A shrink operation does not preserve the fragmentation state of indexes in the database, and generally increases fragmentation to a degree. This is another reason not to repeatedly shrink the database.

Procedure

Step 1 Run the following command to shrink a database:

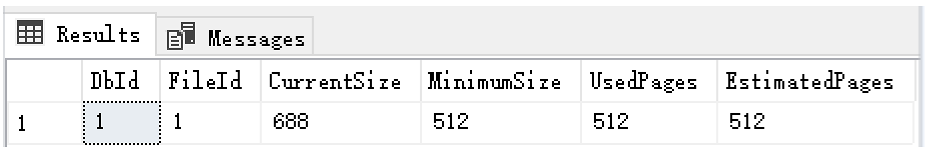
EXEC [master].[dbo].[rds_shrink_database] @DBName='*myDbName***';**

Table 4-10 Parameter description

Parameter	Description
myDbName	Name of the database to be shrunk. If this parameter is not specified, all databases are shrunk by default.

The following figure shows the execution result set. Each result corresponds to the information about each file in the specified database (or all databases).

Figure 4-22 Result set



```
   DbId  FileId  CurrentSize  MinimumSize  UsedPages  EstimatedPages
1  1     1       688          512          512        512
```

Table 4-11 Result set parameter description

Column Name	Description
DbId	Database ID of the current shrink file.
FileId	File ID of the current shrink file.

Column Name	Description
CurrentSize	Number of 8 KB pages occupied by the file.
MinimumSize	Minimum number of 8 KB pages occupied by the file. The value indicates the minimum size or the initial size of the file.
UsedPages	Number of 8 KB pages used by the file.
EstimatedPages	Number of 8 KB pages that the database engine estimates the file can be shrunk to.

Step 2 After the command is successfully executed, the following information is displayed:

```
HW_RDS_Process_Successful: Shrink Database Done.
```

----End

Fault Rectification

If the file size does not change after the database is shrunk, run the following SQL statement to check whether the file has sufficient available space:

SELECT name, size/128.0 - CAST(FILEPROPERTY(name, 'SpaceUsed') AS int)/ 128.0 AS AvailableSpaceInMB FROM sys.database_files;

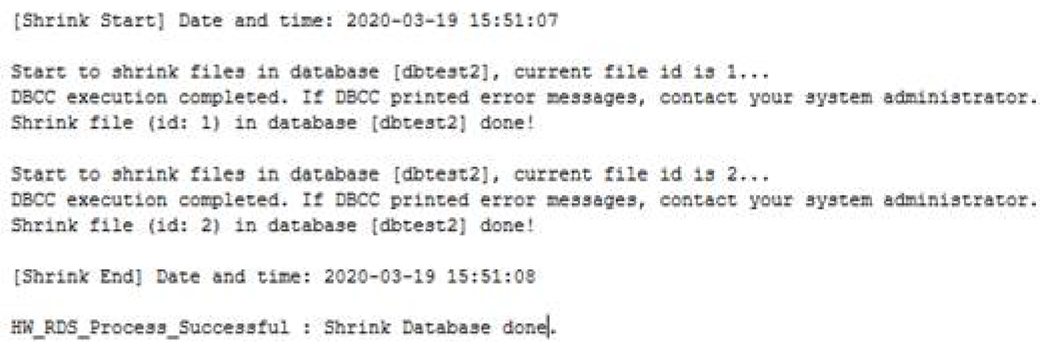
Example

1. Run the following command to shrink the **dbtest2** database:

   EXEC [master].[dbo].[rds_shrink_database] @DBName = 'dbtest2';

   The command output is as follows.

   Figure 4-23 Execution result



```
[Shrink Start] Date and time: 2020-03-19 15:51:07

Start to shrink files in database [dbtest2], current file id is 1...
DBCC execution completed. If DBCC printed error messages, contact your system administrator.
Shrink file (id: 1) in database [dbtest2] done!

Start to shrink files in database [dbtest2], current file id is 2...
DBCC execution completed. If DBCC printed error messages, contact your system administrator.
Shrink file (id: 2) in database [dbtest2] done!

[Shrink End] Date and time: 2020-03-19 15:51:08

HW_RDS_Process_Successful : Shrink Database done.
```

2. Run the following command to shrink all databases:

   EXEC [master].[dbo].[rds_shrink_database];

4.13 Using DAS to Create and Configure Agent Job and DBLink on the Master and Slave Databases for RDS for SQL Server Instances

Scenarios

Data Admin Service (DAS) is a one-stop database management platform that allows you to manage databases on a web console. It offers database development, O&M, and intelligent diagnosis, facilitating your database usage and maintenance. Currently, DAS supports primary/standby switchover of RDS for SQL Server databases, facilitating synchronization between master and slave databases.

Logging In to DAS

Step 1 Log in to the management console.

Step 2 Click in the upper left corner and select a region and a project.

Step 3 Click in the upper left corner of the page and choose **Databases** > **Relational Database Service**.

Step 4 On the **Instances** page, locate the target DB instance and click **Log In** in the **Operation** column.

Alternatively, click the instance name on the **Instances** page. On the displayed **Basic Information** page, click **Log In** in the upper right corner of the page.

Step 5 On the displayed login page, enter the correct username and password and click **Log In**.

----End

Creating a Job for Data Synchronization to the Slave Database

Step 1 Create a job on the master database.

On the DAS console, choose **SQL Operations** > **SQL Query** on the top menu bar. In the msdb database, run the following commands to create a job:

NOTE

If a job has been created on the master database, skip this step.

```
USE [msdb]
GO
DECLARE @jobId BINARY(16)
EXEC  msdb.dbo.sp_add_job @job_name=N'hwtest',
                    @enabled=1,
                    @notify_level_eventlog=0,
                    @notify_level_email=2,
                    @notify_level_page=2,
                    @delete_level=0,
                @category_name=N'[Uncategorized (Local)]',
                @owner_login_name=N'rdsuser', @job_id = @jobId OUTPUT
select @jobId
GO
EXEC msdb.dbo.sp_add_jobserver @job_name=N'hwtest', @server_name = N'*******'
GO
USE [msdb]
GO
EXEC msdb.dbo.sp_add_jobstep @job_name=N'hwtest', @step_name=N'select orders',
                @step_id=1,
                @cmdexec_success_code=0,
                @on_success_action=1,
                @on_fail_action=2,
                @retry_attempts=0,
                @retry_interval=0,
                @os_run_priority=0, @subsystem=N'TSQL',
                @command=N'select * from orders;',
                @database_name=N'test',
                @flags=0
GO
USE [msdb]
GO
EXEC msdb.dbo.sp_update_job @job_name=N'hwtest',
                @enabled=1,
                @start_step_id=1,
                @notify_level_eventlog=0,
                @notify_level_email=2,
                @notify_level_page=2,
                @delete_level=0,
                @description=N'',
                @category_name=N'[Uncategorized (Local)]',
                @owner_login_name=N'zf1',
                @notify_email_operator_name=N'',
                @notify_page_operator_name=N''
GO
```

Run the following SQL statements to check whether the job has been created:

use [msdb]

select * from msdb.dbo.sysjobs where name ='hwtest';

Step 2 Switch to the slave database.

NOTE

Currently, RDS for SQL Server does not support job synchronization between the master and slave databases. Therefore, you need to create a job on the slave database and synchronize the job. Click **Switch SQL Execution Node** next to **Master** to switch to the slave database.

Step 3 Run the commands in **Step 1** to create a job on the slave database.

Alternatively, use SQL Server Management Studio (SSMS) to export the created job to the editor window, copy the job to the SQL query window, and then run the commands in **Step 1** to create a job on the slave database.

If the job fails to be created, you are advised to delete the job first and then create the job again.

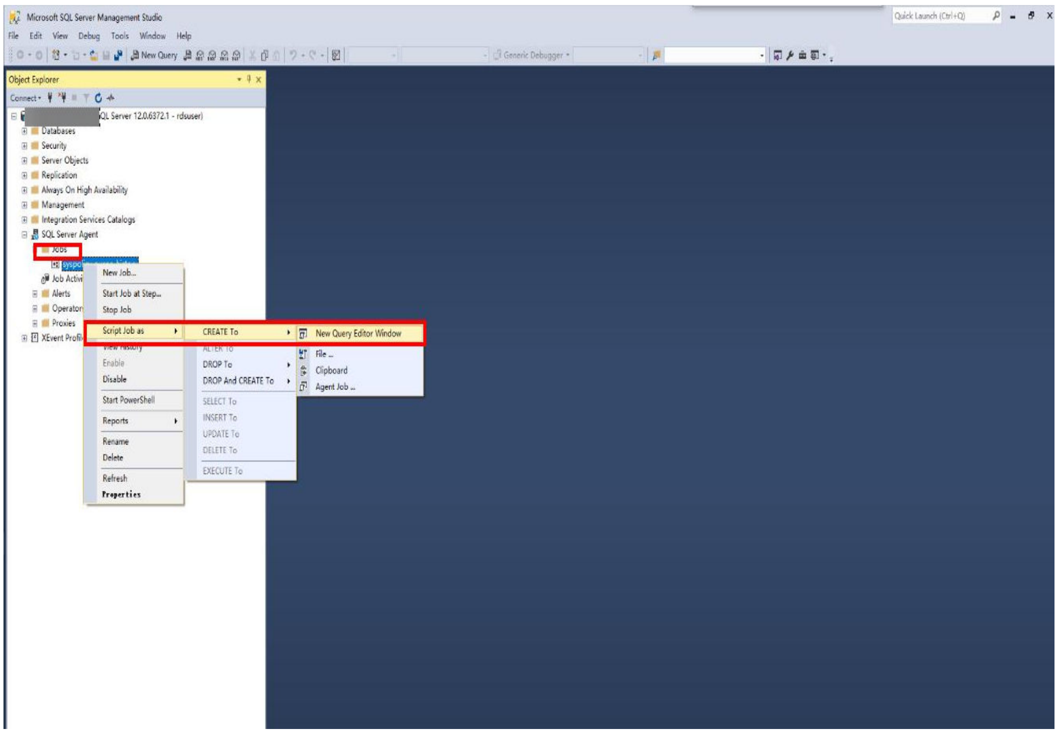
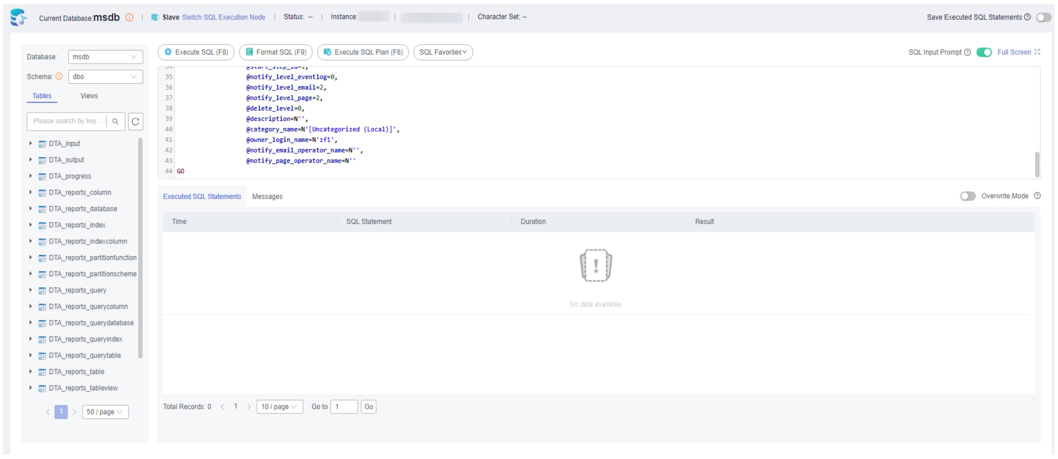


Figure 4-24 Exporting a job



```
                @notify_level_eventlog=0,
                @notify_level_email=2,
                @notify_level_page=2,
                @delete_level=0,
                @description=N'',
                @category_name=N'[Uncategorized (Local)]',
                @owner_login_name=N'zf1',
                @notify_email_operator_name=N'',
                @notify_page_operator_name=N''
GO
```

Figure 4-25 Using the DAS console to create a job on the slave database

Run the following SQL statements to delete the job:

USE [msdb]

GO

EXEC msdb.dbo.sp_delete_job @job_name=N'hwtest', @delete_unused_schedule=1

GO

----End

Creating a DBLink for Data Synchronization to the Slave Database

DAS allows you to create linked servers to synchronize data between master and slave databases.

NOTE

Check whether the MSDTC is configured by referring to **Creating a Linked Server for an RDS for SQL Server DB Instance**.

Step 1 Create a DBLink on the master database.

USE [master]

GO

EXEC master.dbo.sp_addlinkedserver @server = N'TEST', @srvproduct=N'mytest', @provider=N'SQLOLEDB', @datasrc=N'abcd'

EXEC master.dbo.sp_addlinkedsrvlogin @rmtsrvname = N'TEST', @locallogin = NULL , @useself = N'False', @rmtuser = N'rdsuser', @rmtpassword = N'*******'**

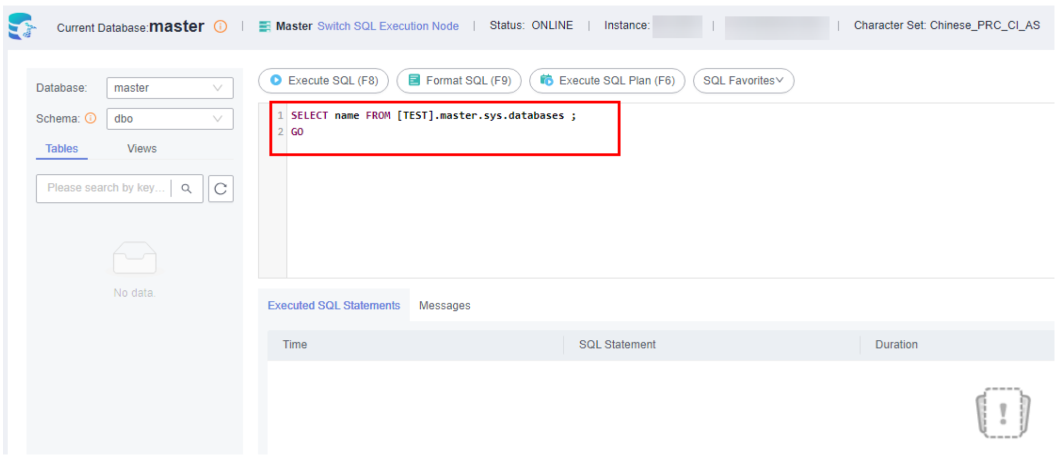
GO

After the creation is successful, the corresponding DB instance or databases can be linked to view data verification. Run the following statements to query databases:

SELECT name FROM [TEST].master.sys.databases ;

GO

Figure 4-26 Database query



```
SELECT name FROM [TEST].master.sys.databases ;
GO
```

Step 2 Create a DBLink on the slave database.

On the DAS console, click **Switch SQL Execution Node** next to **Master** and run the SQL statement for creating a DBLink.

NOTE

If the current DB instance and the interconnected database are not in the same VPC or distributed transactions are enabled with an EIP bound to the primary DB instance, the query statement cannot be executed on the standby DB instance. This step is used only to synchronize the DBLink configuration. After a switchover or failover, the DBLink can be used.

----End

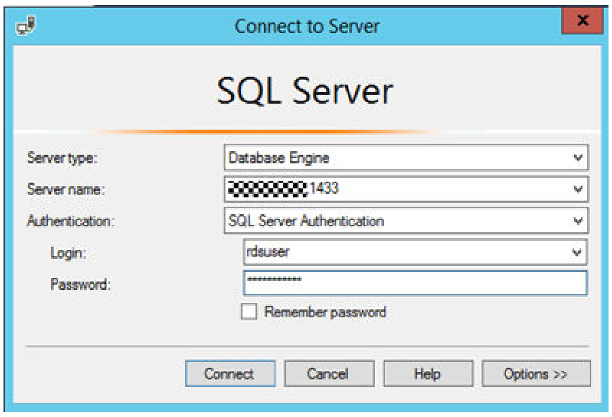
4.14 Creating a Job for Scheduled Instance Maintenance

Scenarios

After a DB instance runs for a period of time, the system performance deteriorates because index fragments increase and statistics are not updated in a timely manner. You are advised to create a SQL agent job to periodically re-create indexes, update statistics, and shrink the database.

Creating an Index Re-creating Job

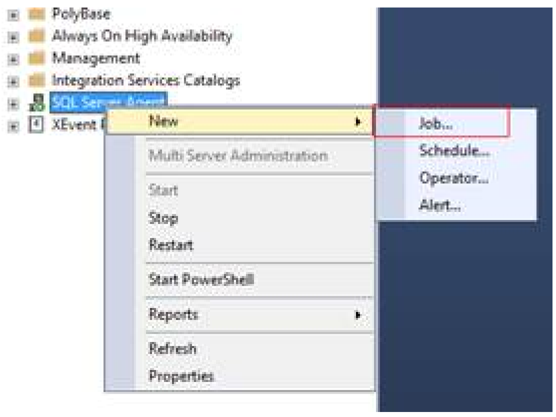
Step 1 Start the SQL Server Management Studio client and log in to it as user **rdsuser**.


Connect to Server

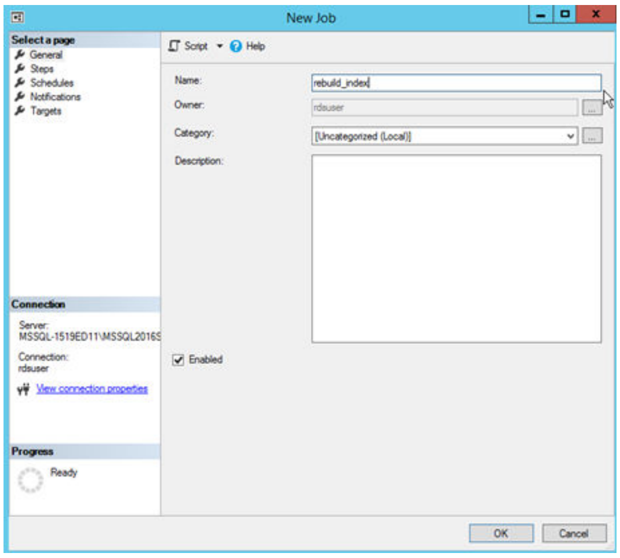
SQL Server

Server type: Database Engine
Server name: [redacted]1433
Authentication: SQL Server Authentication
Login: rdsuser
Password: ************
Remember password

Step 2 Right-click **SQL Server Agent** and choose **New** > **Job** to create an SQL agent job.

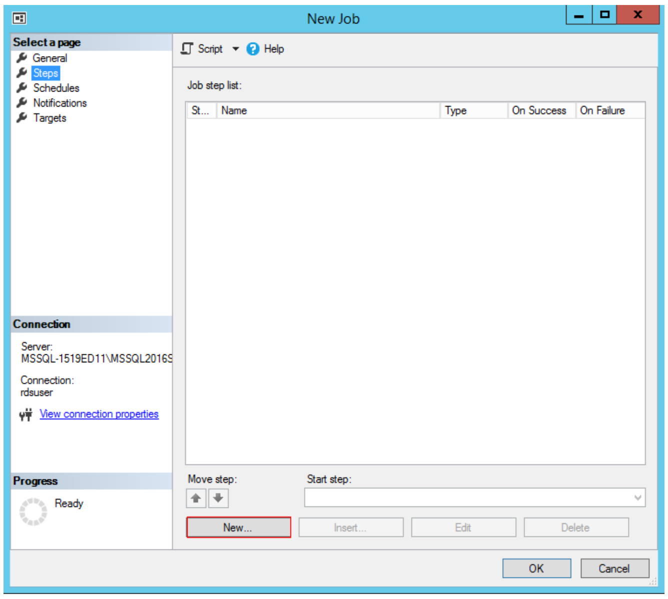


Step 3 Enter the name and description, and click **OK**.



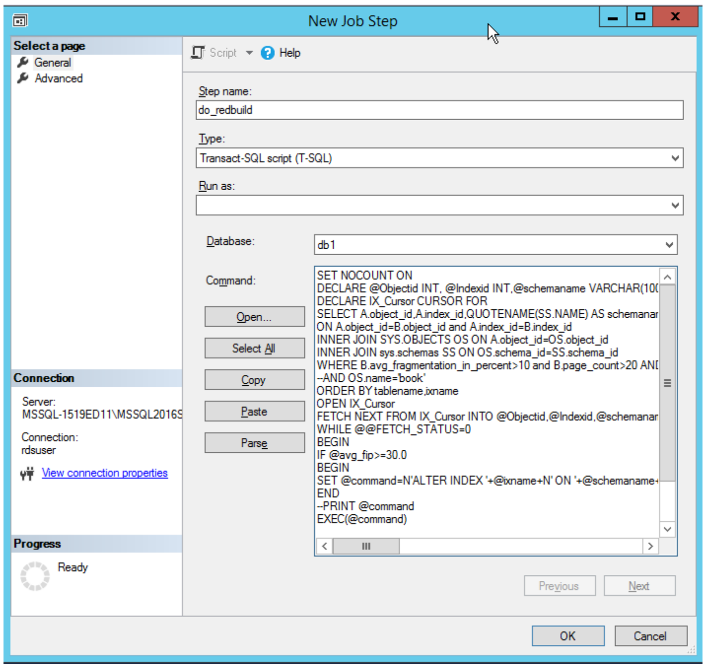
Step 4 Select **Steps** and click **New** to add an execution step.

Figure 4-27 Adding an execution step



Step 5 Enter the step name, type, and command, and click **OK**. Set **Command** to the SQL statements that need to be executed periodically. When the number of index fragments reaches a specified value, for example, 30%, the index can be recreated.

Figure 4-28 Configure parameters



Step name: do_redbuild

Type: Transact-SQL script (T-SQL)

Run as:

Database: db1

Command:

```
SET NOCOUNT ON
DECLARE @Objectid INT, @Indexid INT,@schemaname VARCHAR(10
DECLARE IX_Cursor CURSOR FOR
SELECT A.object_id,A.index_id,QUOTENAME(SS.NAME) AS schemanar
ON A.object_id=B.object_id and A.index_id=B.index_id
INNER JOIN SYS.OBJECTS OS ON A.object_id=OS.object_id
INNER JOIN sys.schemas SS ON OS.schema_id=SS.schema_id
WHERE B.avg_fragmentation_in_percent>10 and B.page_count>20 AN
--AND OS.name='book'
ORDER BY tablename,ixname
OPEN IX_Cursor
FETCH NEXT FROM IX_Cursor INTO @Objectid,@Indexid,@schemanar
WHILE @@FETCH_STATUS=0
BEGIN
IF @avg_fip>=30.0
BEGIN
SET @command=N'ALTER INDEX '+@ixname+N' ON '+@schemaname
END
--PRINT @command
EXEC(@command)
```

Server:
MSSQL-1519ED11\MSSQL2016S

Connection:
rdsuser

Run the following SQL statement to recreate the index because the number of index fragments of all tables in the specified **dbname** exceeds 30%:

```
use [dbname]
SET NOCOUNT ON
DECLARE @Objectid INT, @Indexid INT,@schemaname VARCHAR(100),@tablename
VARCHAR(300),@ixname VARCHAR(500),@avg_fip float,@command VARCHAR(4000)
DECLARE IX_Cursor CURSOR FOR
SELECT A.object_id,A.index_id,QUOTENAME(SS.name) AS
schemaname,QUOTENAME(OBJECT_NAME(B.object_id,B.database_id))as
tablename ,QUOTENAME(A.name) AS ixname,B.avg_fragmentation_in_percent AS avg_fip FROM
sys.indexes A inner join sys.dm_db_index_physical_stats(DB_ID(),NULL,NULL,NULL,'LIMITED') AS B
ON A.object_id=B.object_id and A.index_id=B.index_id
INNER JOIN sys.objects OS ON A.object_id=OS.object_id
INNER JOIN sys.schemas SS ON OS.schema_id=SS.schema_id
WHERE B.avg_fragmentation_in_percent>10 and B.page_count>20 AND A.index_id>0 AND A.is_disabled<>1
--AND OS.name='book'
ORDER BY tablename,ixname
OPEN IX_Cursor
FETCH NEXT FROM IX_Cursor INTO @Objectid,@Indexid,@schemaname,@tablename,@ixname,@avg_fip
WHILE @@FETCH_STATUS=0
BEGIN
IF @avg_fip>=30.0
BEGIN
SET @command=N'ALTER INDEX '+@ixname+N' ON '+@schemaname+N'.'+ @tablename+N' REBUILD ';
END
--PRINT @command
EXEC(@command)
FETCH NEXT FROM IX_Cursor INTO @Objectid,@Indexid,@schemaname,@tablename,@ixname,@avg_fip
END
CLOSE IX_Cursor
DEALLOCATE IX_Cursor
```

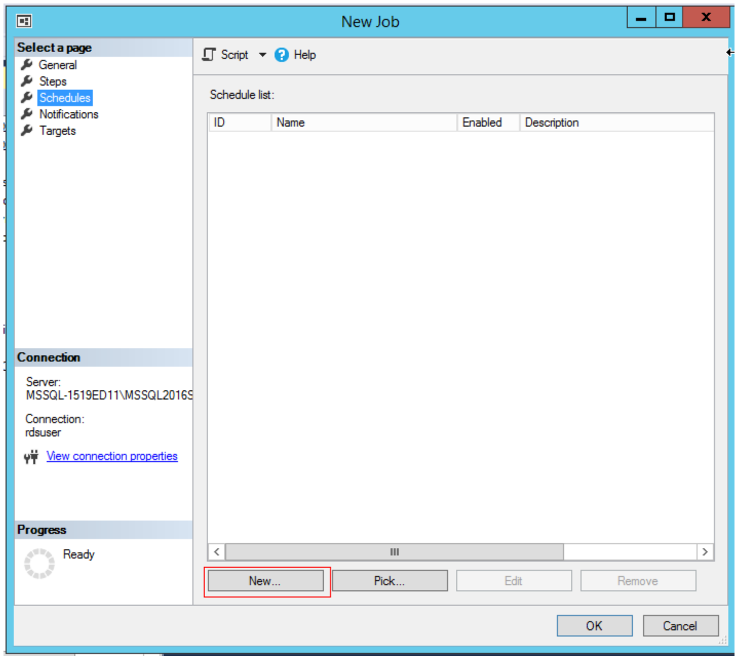
NOTE

In the preceding SQL statements, you only need to change the value of Use [**dbname**] in the first line to the specified database name.

If you need to execute the SQL statements for all databases, modify the SQL statements to add cyclic execution for all databases.

Step 6 Select **Schedules** and click **New** to add a scheduled execution plan.

Figure 4-29 Adding a scheduled execution plan



Schedule list:

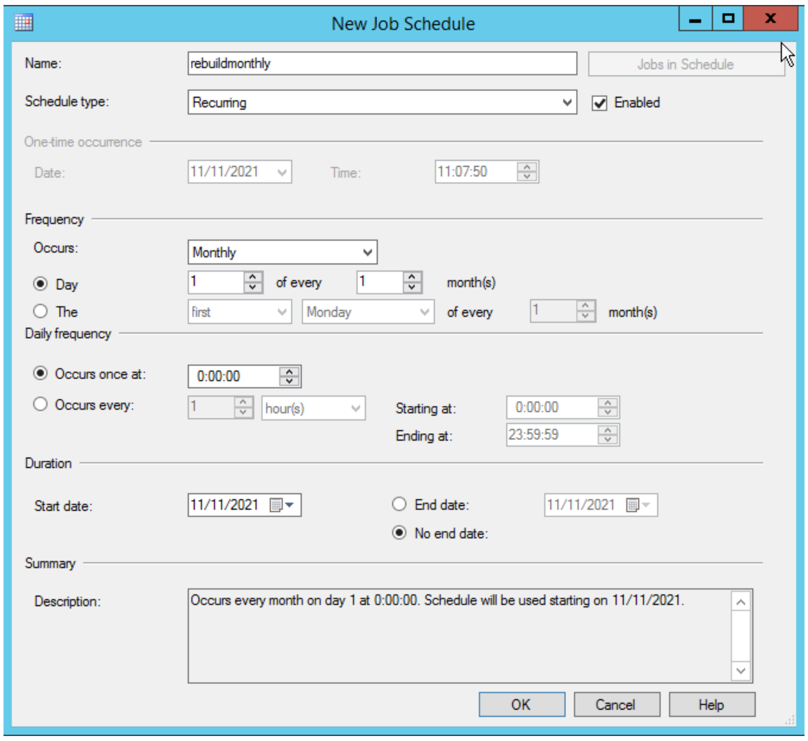
ID	Name	Enabled	Description

Server:
MSSQL-1519ED11\MSSQL2016S

Connection:
rdsuser

Step 7 Add a schedule that is executed once a month, modify the daily frequency and duration, and click **OK**.

Figure 4-30 Configuring a scheduled execution plan



Name: rebuildmonthly

Schedule type: Recurring ☑ Enabled

One-time occurrence
Date: 11/11/2021 Time: 11:07:50

Frequency
Occurs: Monthly
◉ Day 1 of every 1 month(s)
○ The first Monday of every 1 month(s)

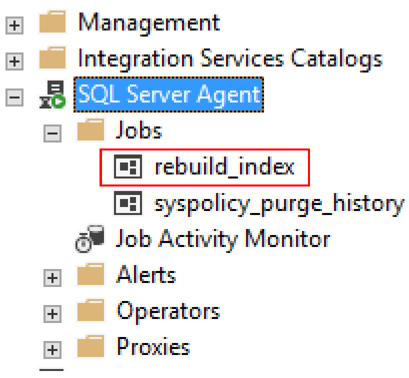
Daily frequency
◉ Occurs once at: 0:00:00
○ Occurs every: 1 hour(s) Starting at: 0:00:00 Ending at: 23:59:59

Duration
Start date: 11/11/2021 ○ End date: 11/11/2021
◉ No end date:

Summary
Description: Occurs every month on day 1 at 0:00:00. Schedule will be used starting on 11/11/2021.

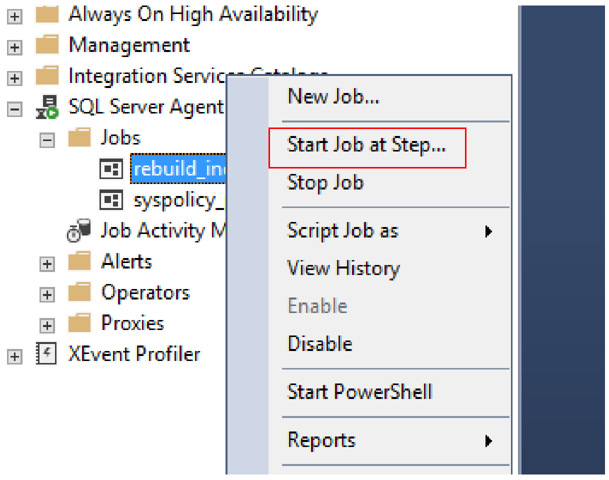
Step 8 View that the job has been created.

Figure 4-31 job

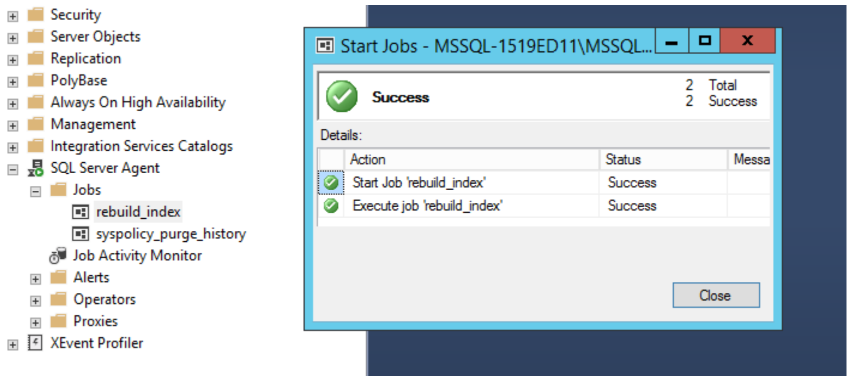


Step 9 Right-click the job and choose **Start Job at Step** to manually run the job.

Figure 4-32 Runing a job.



Step 10 Check whether the job can run properly. If the job runs normally, the maintenance job for periodically recreating the indexes of the **db1** database has been created.



Success

2 Total
2 Success

Details:

Action	Status	Messa
Start Job 'rebuild_index'	Success	
Execute job 'rebuild_index'	Success	

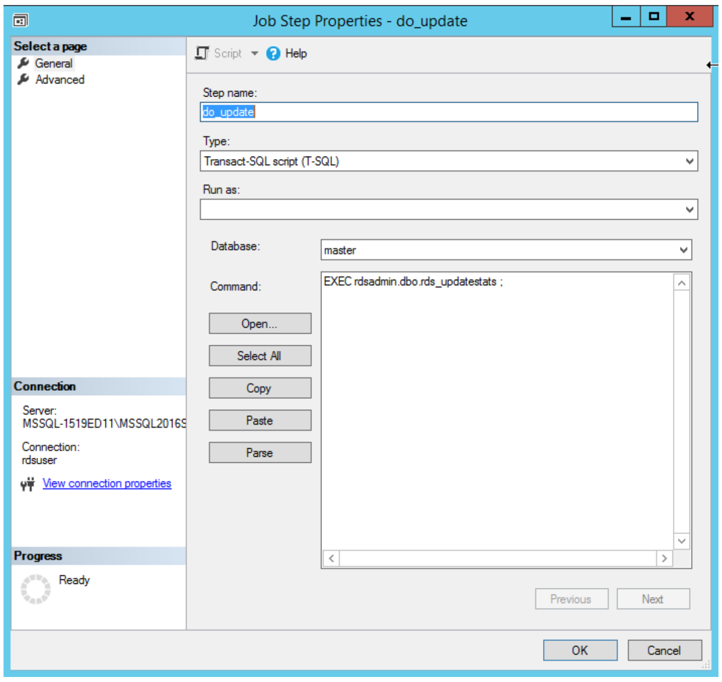
----End

Updating Statistics

Step 1 Perform **Step 1** to **Step 4**.

Step 2 Enter the step name, type, and command, and click **OK**. Set **Command** to the stored procedure for updating statistics. For details, see **Updating Database Statistics**.

Figure 4-33 Updating statistics



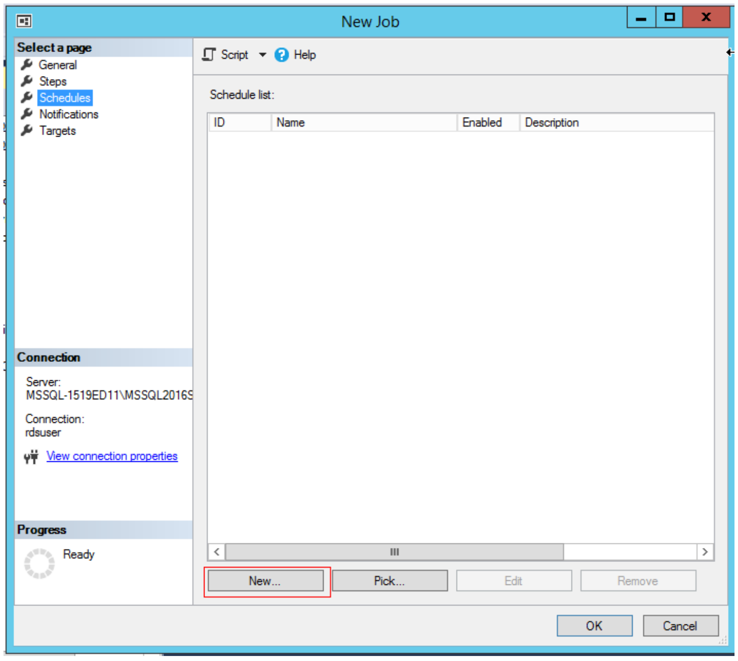
Step name: do_update
Type: Transact-SQL script (T-SQL)
Run as:
Database: master
Command:

```
EXEC rdsadmin.dbo.rds_updatestats ;
```

Server: MSSQL-1519ED11\MSSQL2016S
Connection: rdsuser

Step 3 Select **Schedules** and click **New** to add a scheduled execution plan.

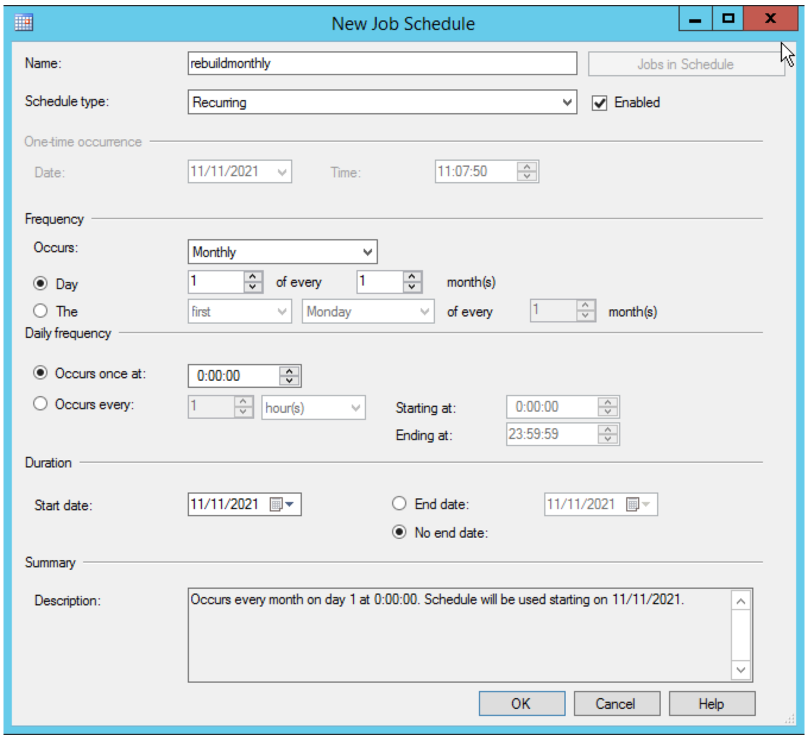
Figure 4-34 Adding a scheduled execution plan



Server: MSSQL-1519ED11\MSSQL2016S
Connection: rdsuser

Step 4 Add a schedule that is executed once a month, modify the daily frequency and duration, and click **OK**.

Figure 4-35 Configuring a scheduled execution plan


New Job Schedule

Name: rebuildmonthly
Schedule type: Recurring ✓ Enabled

One-time occurrence
Date: 11/11/2021 Time: 11:07:50

Frequency
Occurs: Monthly
● Day 1 of every 1 month(s)
○ The first Monday of every 1 month(s)

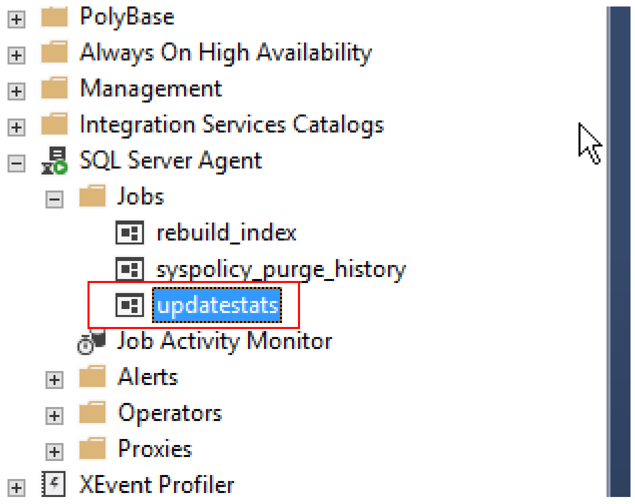
Daily frequency
● Occurs once at: 0:00:00
○ Occurs every: 1 hour(s) Starting at: 0:00:00 Ending at: 23:59:59

Duration
Start date: 11/11/2021 ○ End date: 11/11/2021
● No end date:

Summary
Description: Occurs every month on day 1 at 0:00:00. Schedule will be used starting on 11/11/2021.

Step 5 View that the job has been created.

Figure 4-36 Updating statistics job



Step 6 Right-click the job and choose **Start Job at Step** to manually run the job.

----End

Shrinking the Database Periodically

Step 1 Perform **Step 1** to **Step 4**.

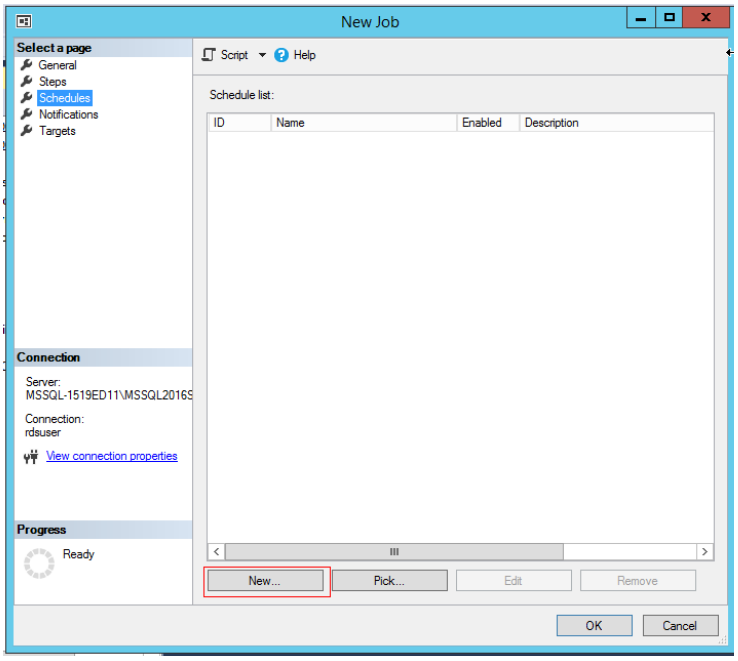
Step 2 Enter the step name, type, and command, and click **OK**. Set **Command** to the SQL commands for shrinking the database.

```
EXEC [master].[dbo].[rds_shrink_database_log] @dbname='myDbName';
```

Set **@dbname** to the database name.

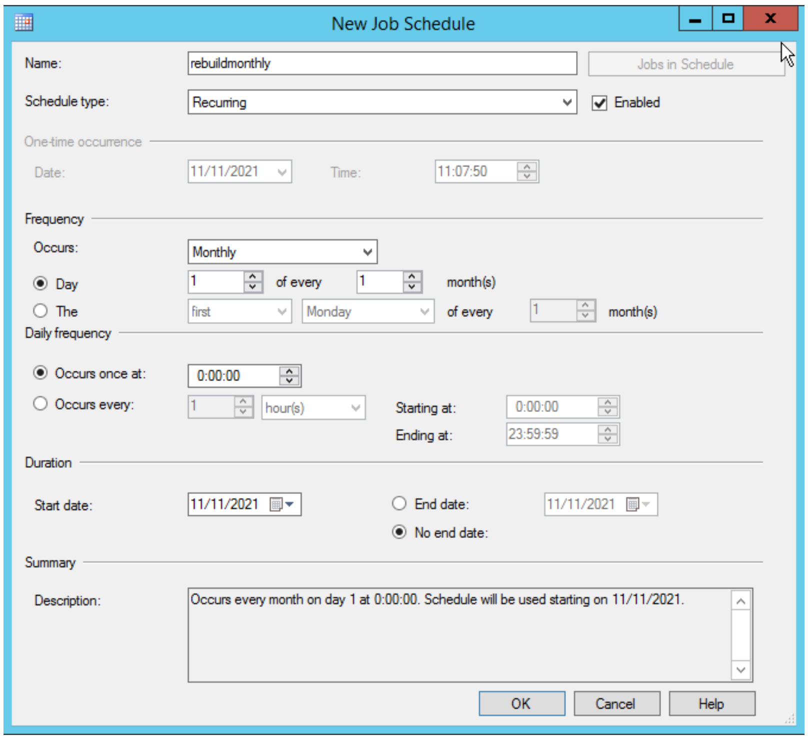
Step 3 Select **Schedules** and click **New** to add a scheduled execution plan.

Figure 4-37 Adding a scheduled execution plan



Step 4 Add a schedule that is executed once a month, modify the daily frequency and duration, and click **OK**.

Figure 4-38 Configuring a scheduled execution plan



Name: rebuildmonthly
Schedule type: Recurring ✓ Enabled
One-time occurrence: Date: 11/11/2021 Time: 11:07:50
Frequency
Occurs: Monthly
[✓ Day] 1 of every 1 month(s)
The first Monday of every 1 month(s)
Daily frequency
[✓ Occurs once at:] 0:00:00
Occurs every: 1 hour(s) Starting at: 0:00:00 Ending at: 23:59:59
Duration
Start date: 11/11/2021 End date: 11/11/2021 [✓ No end date:]
Summary
Description: Occurs every month on day 1 at 0:00:00. Schedule will be used starting on 11/11/2021.

Step 5 Right-click the job and choose **Start Job at Step** to manually run the job.

----End

4.15 Using Extended Events

Extended event permissions are available now. You can use **rdsuser** to manage extended events or grant extended event permissions to other users.

For more information, see **Quickstart: Extended Events in SQL Server**.

Constraints

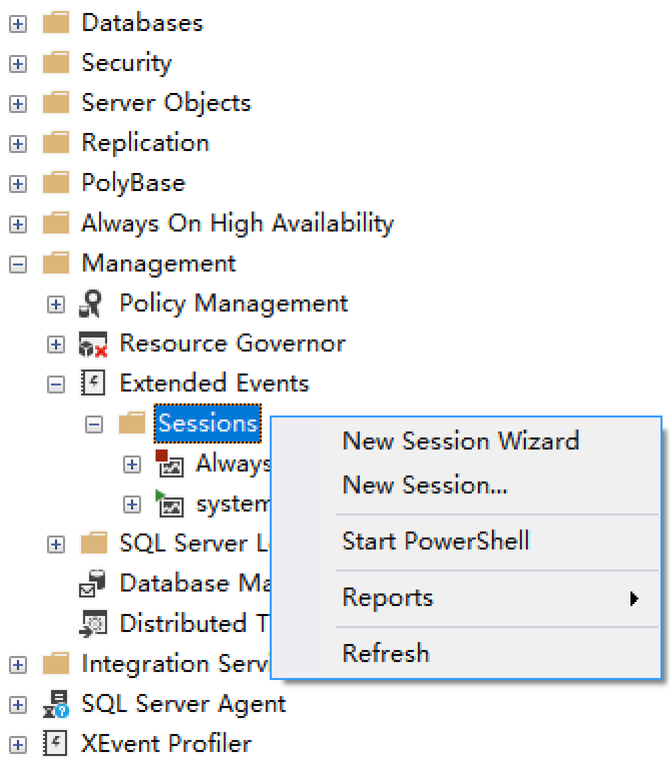
- All RDS for SQL Server 2008 versions do not support extended events because Microsoft SQL Server 2008 does not support extended events.
- The etw_classic_sync_target type is not available for extended event targets.
- When an extended event is created or updated, only the path **D:\RDSDBDATA \Log\error** is supported. The file name can be customized.

Creating an Extended Event

Step 1 Start the SQL Server Management Studio (SSMS) client and log in to it as user **rdsuser**.

Step 2 Choose **Management** > **Sessions** > **New Session**.

Figure 4-39 Creating an extended event



Step 3 Click **General** and enter a session name.

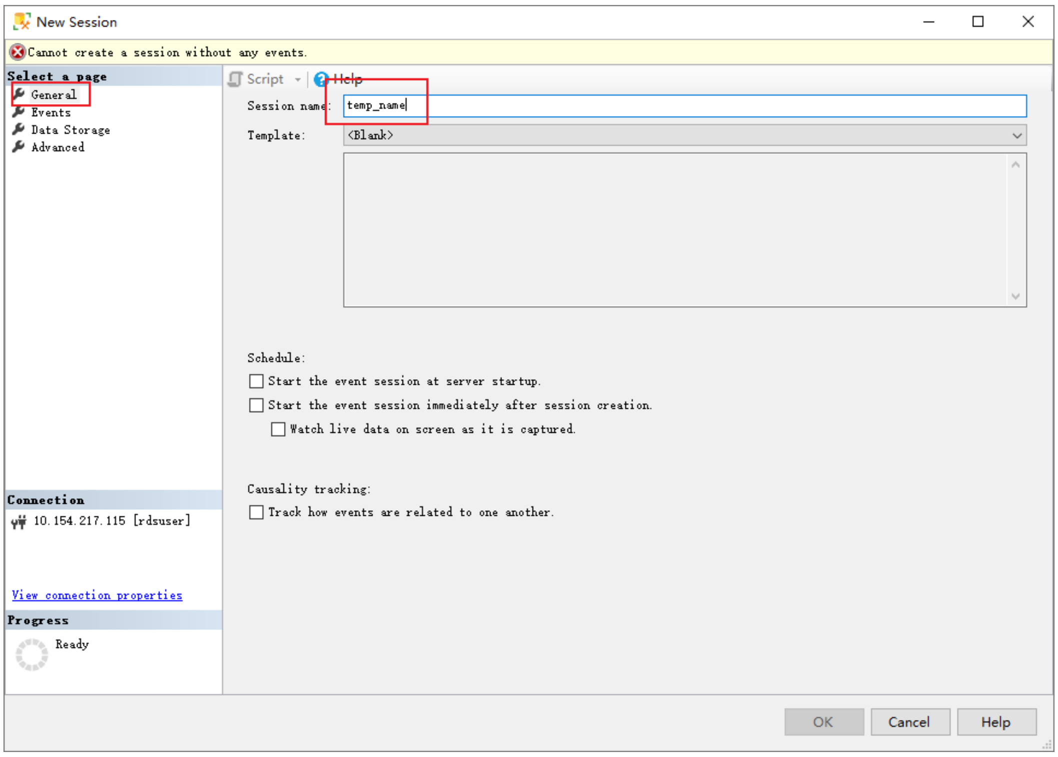
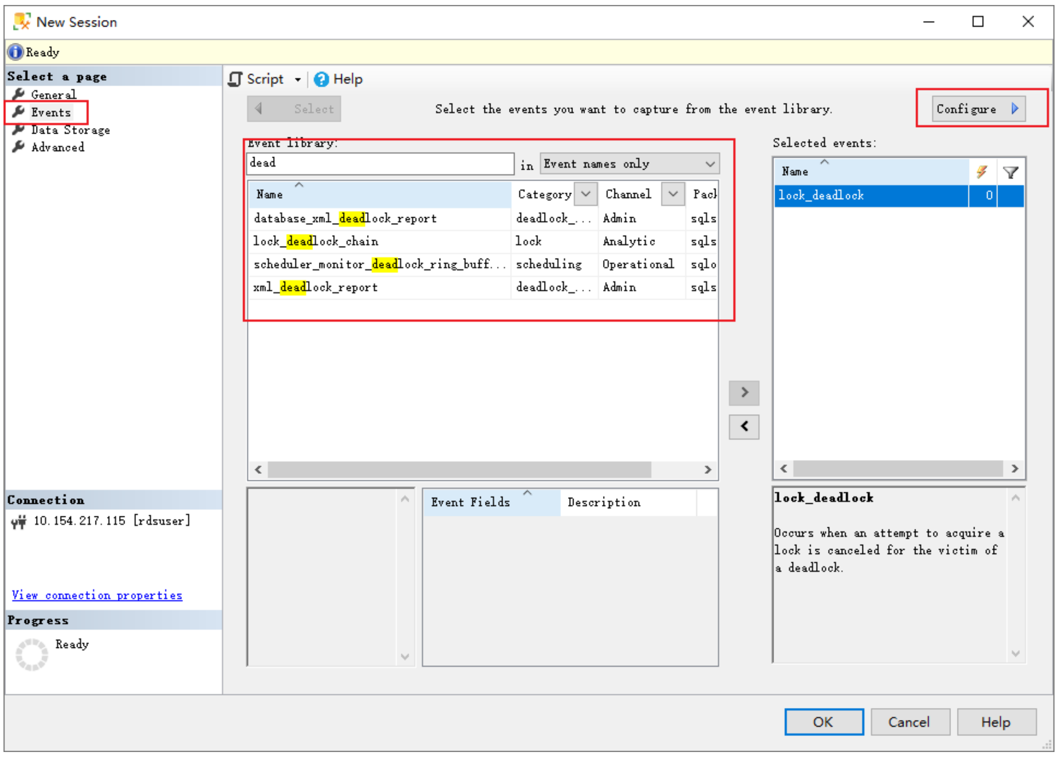

New Session
Cannot create a session without any events.
Select a page
General
Events
Data Storage
Advanced
Script
Help
Session name: temp_name
Template: <Blank>
Schedule:
Start the event session at server startup.
Start the event session immediately after session creation.
Watch live data on screen as it is captured.
Causality tracking:
Track how events are related to one another.
Connection
10.154.217.115 [rdsuser]
View connection properties
Progress
Ready
OK
Cancel
Help

Figure 4-40 Entering a session name

Step 4 Click **Events** and select an event.



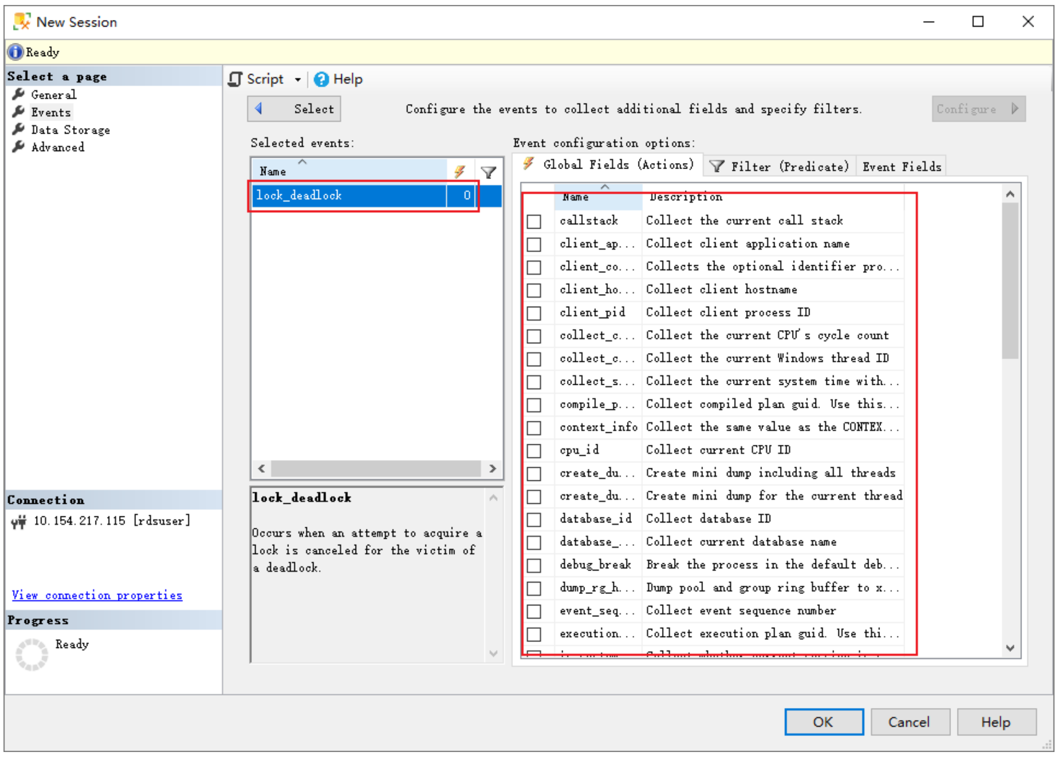
Name	Category	Channel	Pack
database_xml_deadlock_report	deadlock_...	Admin	sqls
lock_deadlock_chain	lock	Analytic	sqls
scheduler_monitor_deadlock_ring_buff...	scheduling	Operational	sqlo
xml_deadlock_report	deadlock_...	Admin	sqls

lock_deadlock

Occurs when an attempt to acquire a lock is canceled for the victim of a deadlock.

Figure 4-41 Selecting an event

Step 5 Click **Configure** on the page displayed in **Step 4**.



Configure the events to collect additional fields and specify filters.

Selected events:

Name	
lock_deadlock	0

10.154.217.115 [rdsuser]

Ready

lock_deadlock

Occurs when an attempt to acquire a lock is canceled for the victim of a deadlock.

Event configuration options: Global Fields (Actions) | Filter (Predicate) | Event Fields

	Name	Description
☐	callstack	Collect the current call stack
☐	client_ap...	Collect client application name
☐	client_co...	Collects the optional identifier pro...
☐	client_ho...	Collect client hostname
☐	client_pid	Collect client process ID
☐	collect_c...	Collect the current CPU's cycle count
☐	collect_c...	Collect the current Windows thread ID
☐	collect_s...	Collect the current system time with...
☐	compile_p...	Collect compiled plan guid. Use this...
☐	context_info	Collect the same value as the CONTEX...
☐	cpu_id	Collect current CPU ID
☐	create_du...	Create mini dump including all threads
☐	create_du...	Create mini dump for the current thread
☐	database_id	Collect database ID
☐	database_...	Collect current database name
☐	debug_break	Break the process in the default deb...
☐	dump_rg_h...	Dump pool and group ring buffer to x...
☐	event_seq...	Collect event sequence number
☐	execution...	Collect execution plan guid. Use thi...

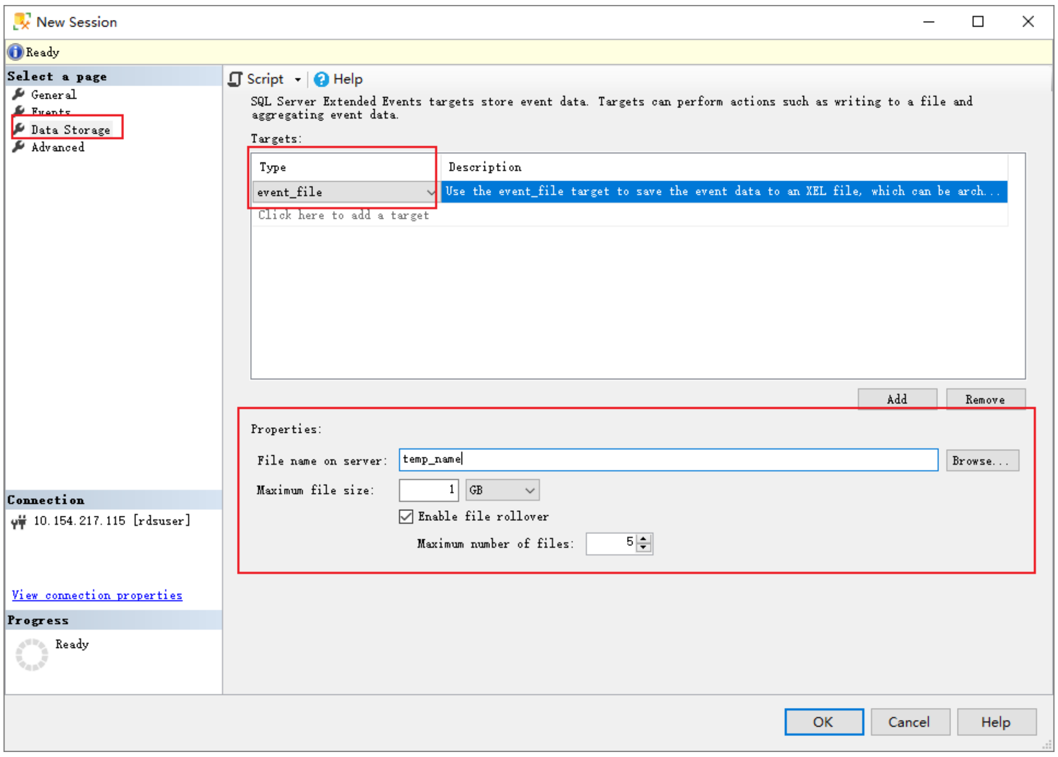
Figure 4-42 Configuring an event

Step 6 Click **Data Storage** to configure the data storage.

NOTE

The file name can be customized. There is no need to click **Browse** because you can only browse the file system of the client where the SSMS is located, but not the file system of the RDS for SQL Server server. RDS for SQL Server supports the **D:\RDSDBDATA\Log\error** path only, so you only need to change the file name.

Figure 4-43 Configuring the data storage



New Session

Ready

Select a page
- General
- Events
- Data Storage
- Advanced

Script | Help

SQL Server Extended Events targets store event data. Targets can perform actions such as writing to a file and aggregating event data.

Targets:

Type	Description
event_file	Use the event_file target to save the event data to an XEL file, which can be arch...
Click here to add a target	

Properties:

File name on server: temp_name

Maximum file size: 1 GB

[✓] Enable file rollover

Maximum number of files: 5

Connection

10.154.217.115 [rdsuser]

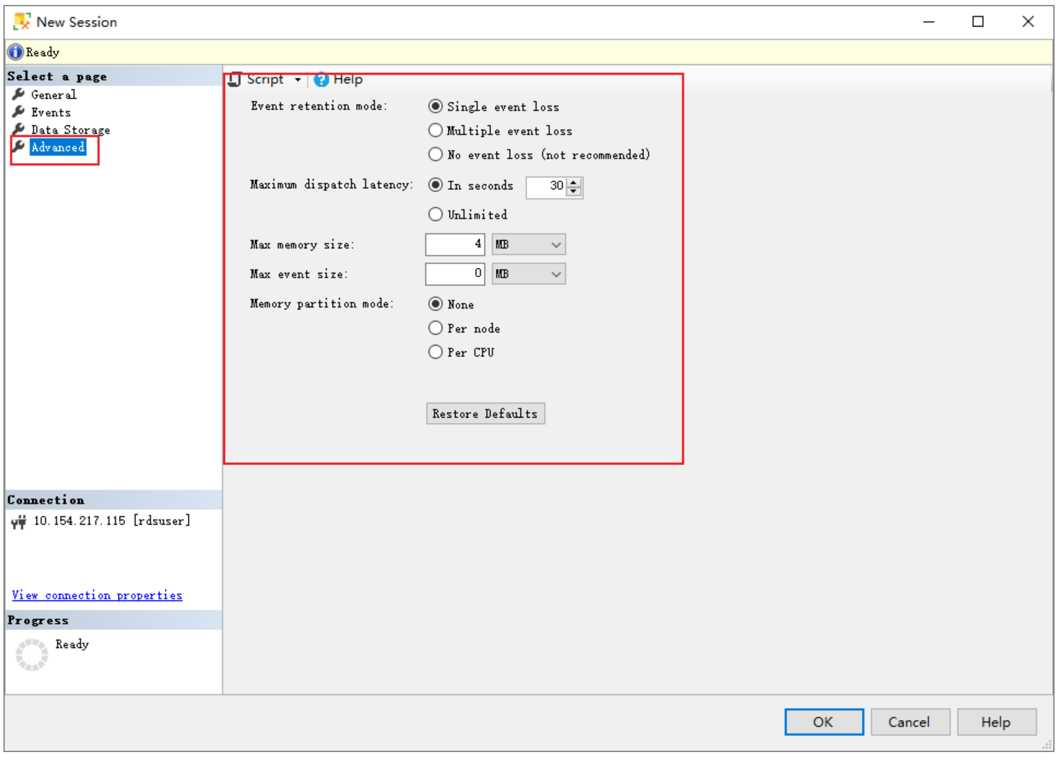
View connection properties

Progress

Ready

Step 7 Click **Advanced** to configure the file generation policy.

Figure 4-44 Configuring the file generation policy



New Session

Ready

Select a page
- General
- Events
- Data Storage
- Advanced

Script | Help

Event retention mode: [✓ Single event loss] Multiple event loss No event loss (not recommended)

Maximum dispatch latency: [✓ In seconds] 30 Unlimited

Max memory size: 4 MB

Max event size: 0 MB

Memory partition mode: [✓ None] Per node Per CPU

Connection

10.154.217.115 [rdsuser]

View connection properties

Progress

Ready

Step 8 Use the script to generate SQL statements. After confirming that the SQL statements are correct, run the SQL statements to create an extended event.

```
-- Example SQL statements generated
CREATE EVENT SESSION [temp_name] ON SERVER
ADD EVENT sqlserver.lock_deadlock(
ACTION(sqlserver.session_id,sqlserver.sql_text,sqlserver.username))
ADD TARGET package0.event_file(SET filename=N'temp_name')
GO
```

----End