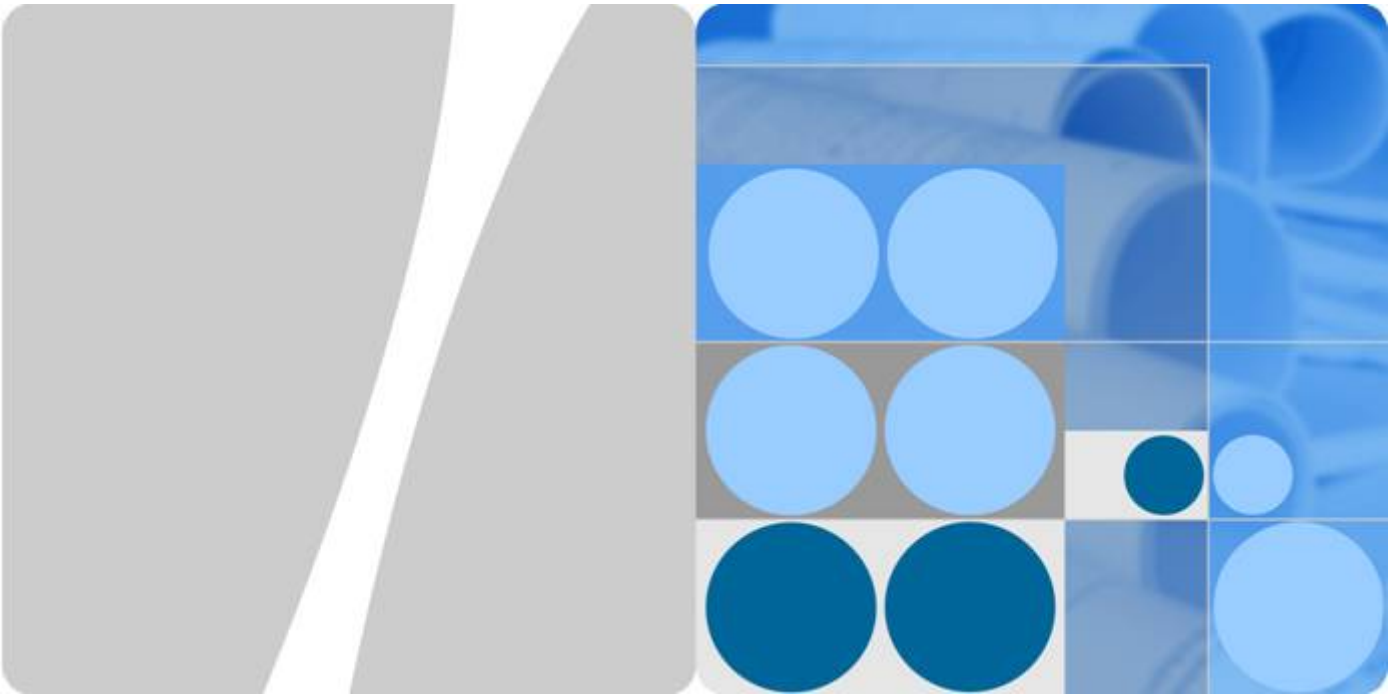


Relational Database Service

Best Practices

Issue 01

Date 2020-12-29

HUAWEI TECHNOLOGIES CO., LTD.

HUAWEI

Trademarks and Permissions


and other Huawei trademarks are trademarks of Huawei Technologies Co., Ltd.
All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Huawei Technologies Co., Ltd.

Address: Huawei Industrial Base
Bantian, Longgang
Shenzhen 518129
People's Republic of China

Website: https://e.huawei.com

Contents

1 Overview

This document provides best practices for HUAWEI CLOUD Relational Database Service (RDS) and guides you through buying DB instances that meet your service requirements.

DB Engine	Reference	Description
MySQL	**Using RDS for MySQL to Set Up WordPress**	Describes how to set up WordPress in the 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 WordPress in the LAMP environment using HUAWEI CLOUD Virtual Private Cloud (VPC), Elastic Cloud Server (ECS), and RDS for MySQL.
	Downloading a Backup File and Restoring Data from It	Describes how to download backup files locally and how to restore data from backup files.
	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.
	Identifying Why CPU Usage of RDS MySQL DB Instances Is High and Providing Solutions	Identify the reason why the CPU usage is high or even reaches 100% and how to troubleshoot this issue.
	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.

DB Engine	Reference	Description
PostgreSQL	Viewing Slow Query Logs of PostgreSQL DB Instances	Describes how to query and optimize PostgreSQL slow query logs.
	Performing Basic Operations Using pgAdmin	Describes how to use pgAdmin to connect to RDS for PostgreSQL and create databases and tables.
	Read/Write Splitting with PostgreSQL Proxy	Describes the principle and process of read/write splitting of PostgreSQL Proxy.
SQL Server	Restoring Data from Backup Files to RDS Microsoft SQL Server DB Instances	Describes the version restrictions on SQL Server backup and restoration.
	Migrating Data from a Self-Built SQL Server Database on an ECS to an RDS Microsoft SQL Server DB Instance	Describes how to migrate a self-built SQL Server database on an ECS to an RDS for SQL Server DB instance.
	Modifying Parameters in a Parameter Template	Describes how to modify parameter templates for Microsoft SQL Server DB instances.
	Supporting DMVs	Describes how to dynamically manage views through DMV on RDS for SQL Server.
	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.
	Migrating Microsoft SQL Server Data Using DRS	Describes how to migrate SQL Server data using DRS.
	Create tempdb Files	Describes how to create tempdb temporary data files on RDS for SQL Server.
	Microsoft SQL Server Publication and Subscription	Describes Microsoft SQL Server publication and subscription.

DB Engine	Reference	Description
	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 SQL Server DB Instance	Describes how to create a linked server to access another SQL Server DB instance.
	How Can I Deploy the Offline SSRS Report Service on RDS for SQL Server?	Describes how to use SQL Server Reporting Services (SSRS).
	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 SQL Server database.
	Using DAS to Create and Configure Agent Job and DBLink on the Master and Slave Databases for SQL Server Instances	Describes how to use DAS to create and configure Agent Job and DBLink on primary and standby SQL Server DB instances.

2 MySQL

2.1 Using RDS for MySQL to Set Up WordPress

WordPress is a blog platform developed based on PHP. It is usually used with 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 Virtual Private Cloud (VPC), Elastic Cloud Server (ECS), and Relational Database Service (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 Under **Network**, click **Virtual Private Cloud**. The VPC console is displayed.

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

Step 5 On the displayed page, enter a VPC name, set **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 has been successfully created, return to network console.

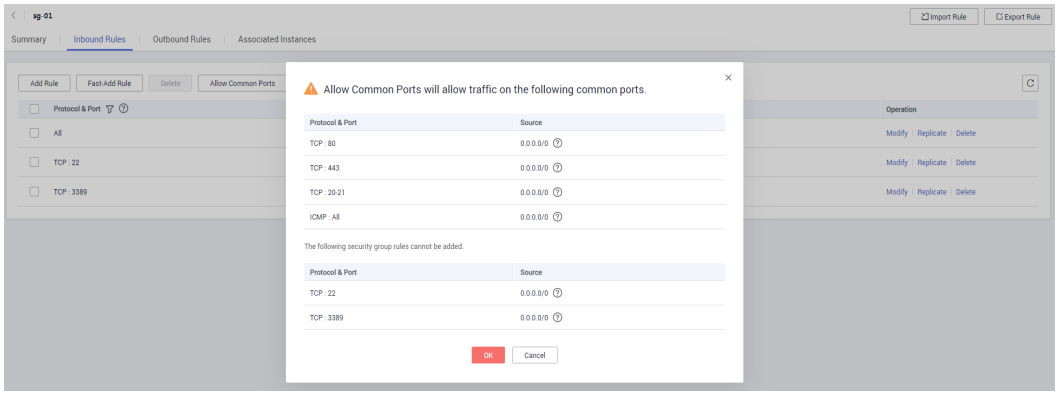
Step 6 On the network console, choose **Access Control** > **Security Groups** and click **Create Security Group**. The following uses sg-01 as an example.

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 2-1 Adding a security group rule



----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.

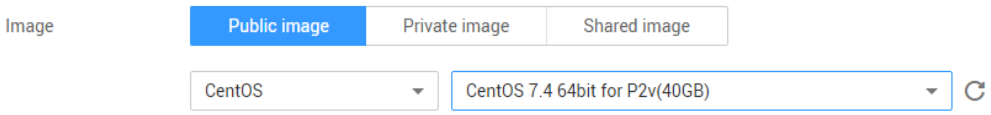
Step 3 Under **Computing**, click **Elastic Cloud Server**. The **Elastic Cloud Server** page is displayed.

Step 4 On the ECS console, buy an ECS. For details about billing information, see **Product Pricing Details**.

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 2-2**.

Figure 2-2 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, you can view and manage it on the ECS console.

----End

Setting Up the LAMP Environment

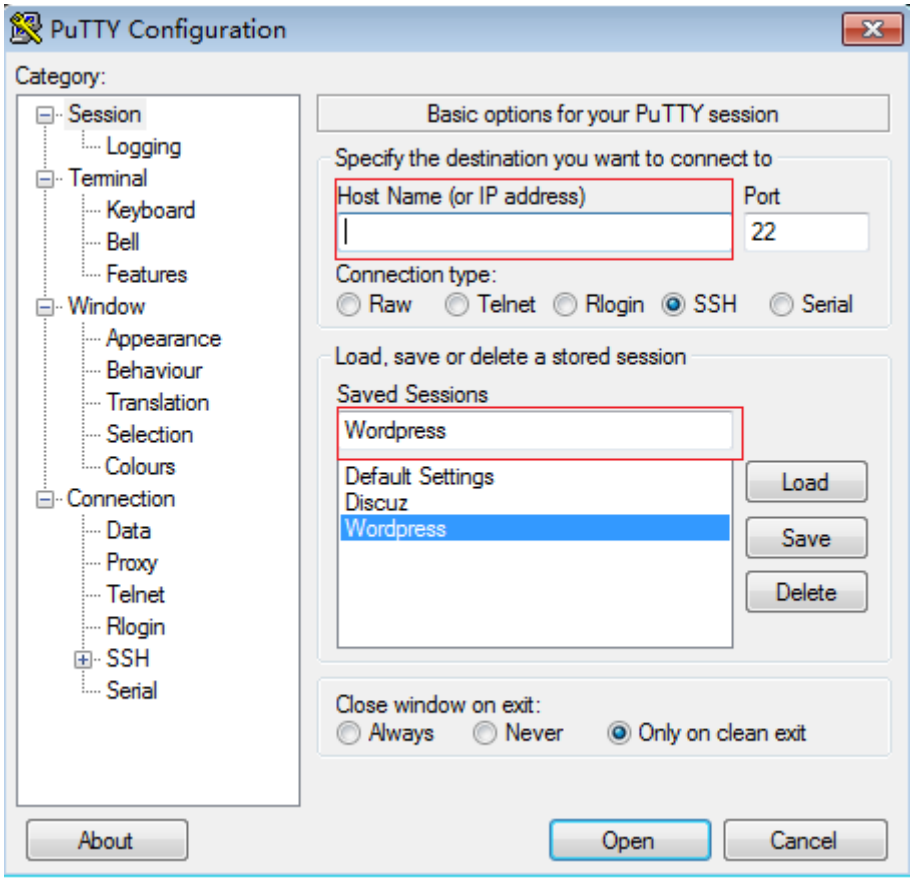
Step 1 Download the PuTTY client.

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 2-3**.

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 2-3 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://cn.wordpress.org/wordpress-4.9.1-zh_CN.tar.gz

tar xzf wordpress-4.9.1-zh_CN.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

----End

Buying and Configuring an RDS DB Instance

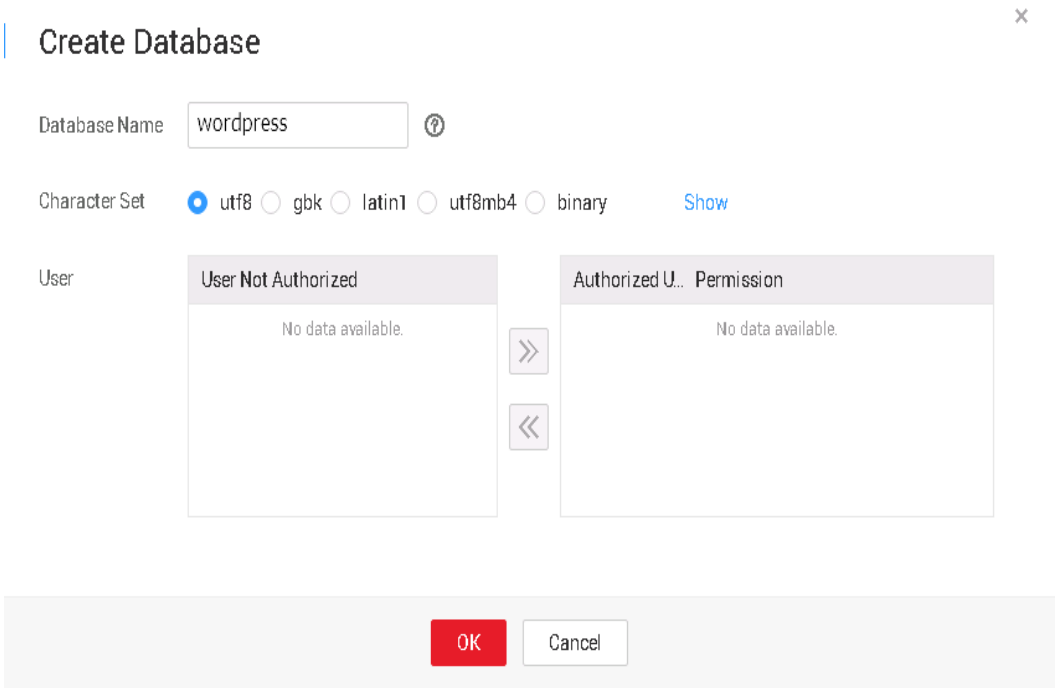
Step 1 **Buy a DB instance** as required.

- DB instance rds-01 is used as an example. Select MySQL 5.6 or 5.7.
- Ensure that the RDS DB instance use 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 **Instance Management** page, click the target DB instance rds-01. The **Basic Information** page is displayed.

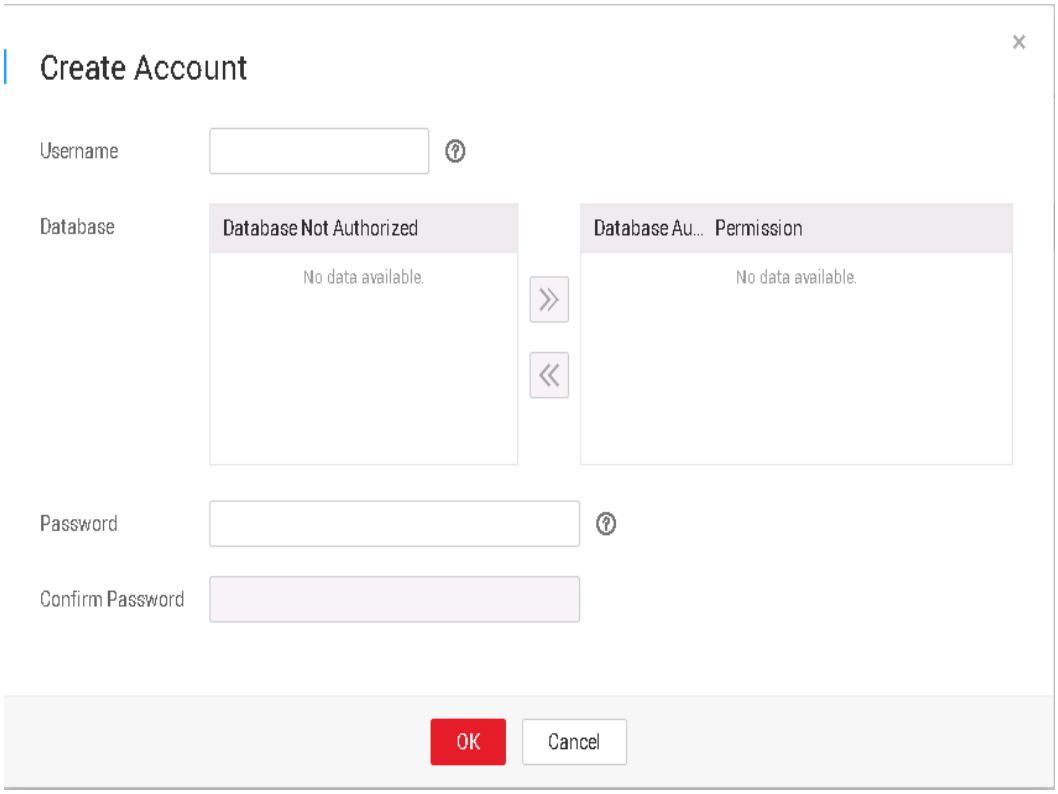
Step 3 Choose **Databases** in the navigation pane on the left and click **Create Database**. In the displayed dialog box, enter a database name, such as ***wordpress***, select a character set, and authorize permissions for database users. Then, click **OK**.

Figure 2-4 Creating a database



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 2-5 Creating an account



----End

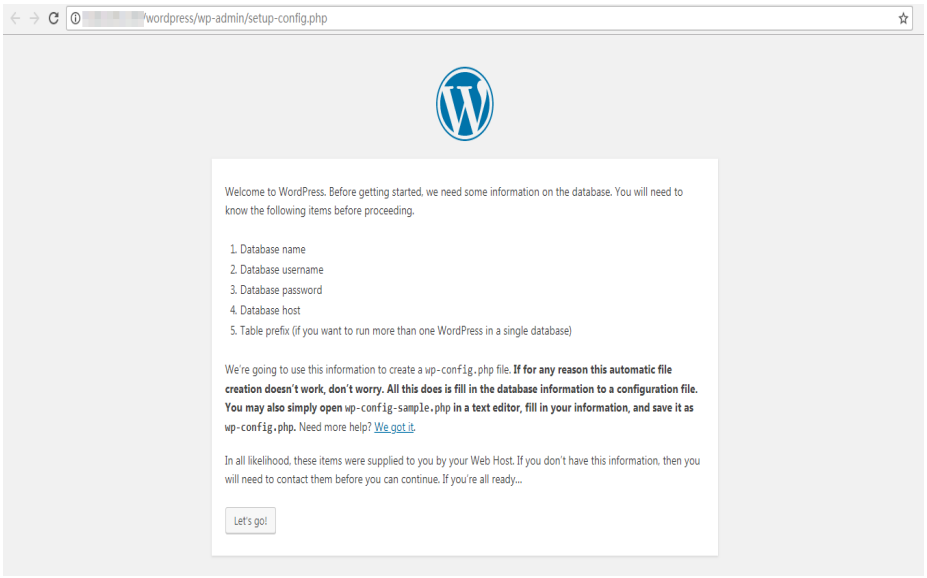
Installing WordPress

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**.

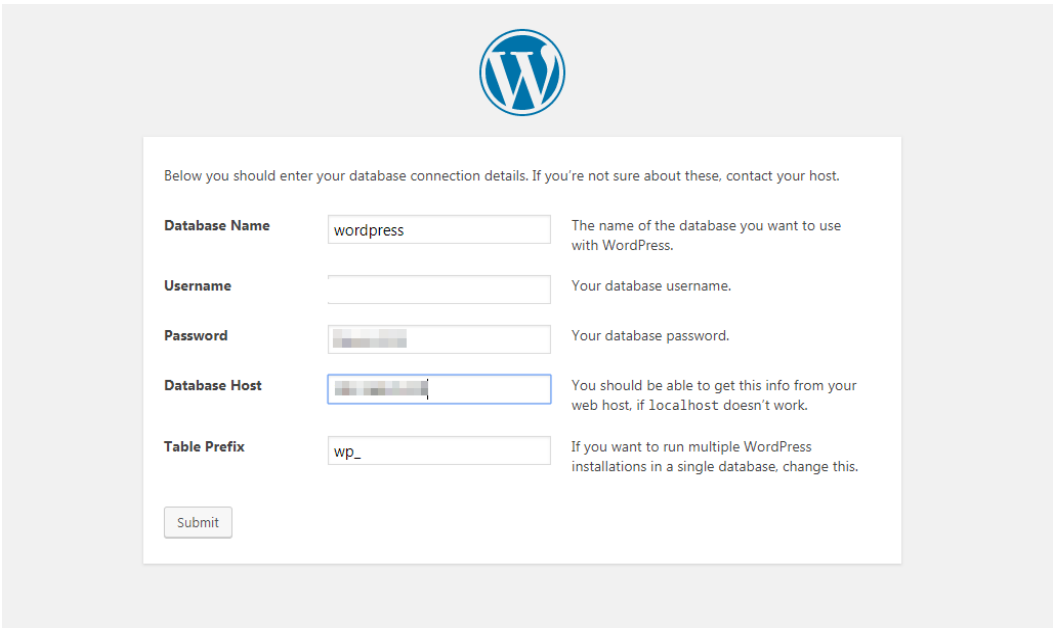
Figure 2-6 Visiting WordPress



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

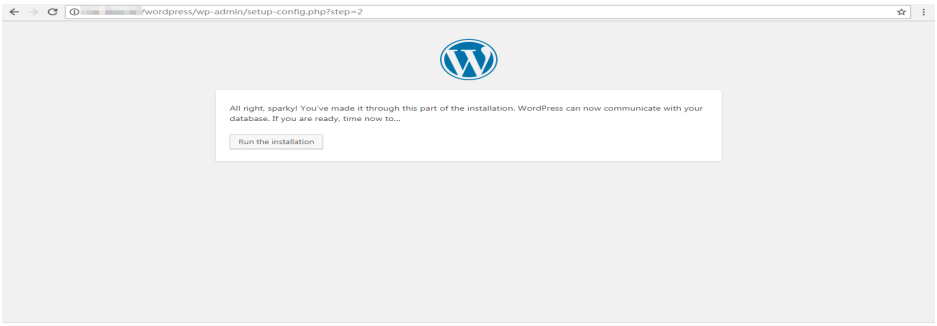
- The database name is ***wordpress***.
- The username is ***tony***.
- The password is the one that you set for ***tony***.
- The database host is the private IP address of DB instance rds-01.

Figure 2-7 Entering database connection information



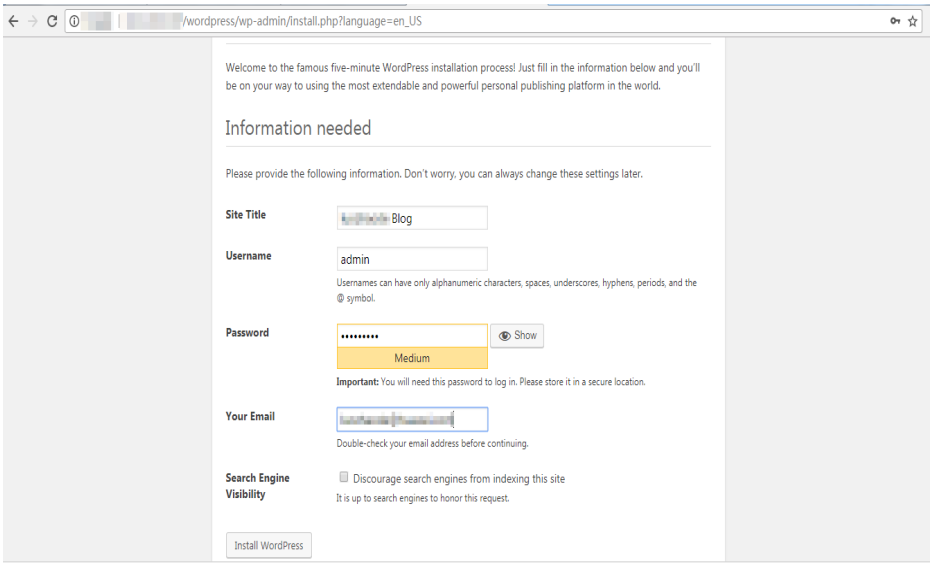
Step 4 After the database connection details are verified, click **Run the installation**.

Figure 2-8 Running the installation



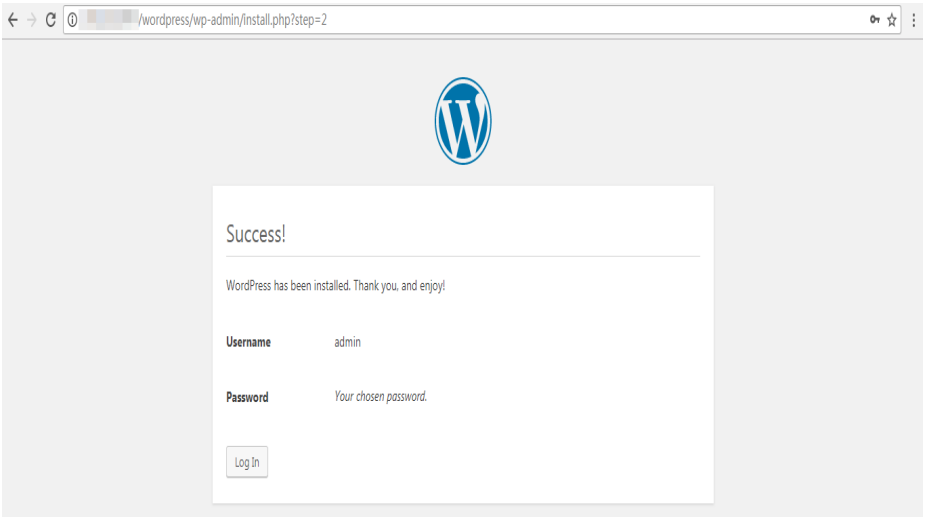
Step 5 Set **Site Title**, **Username**, and **Password** for logging in to your blog. Then, click **Install WordPress**.

Figure 2-9 Setting basic information



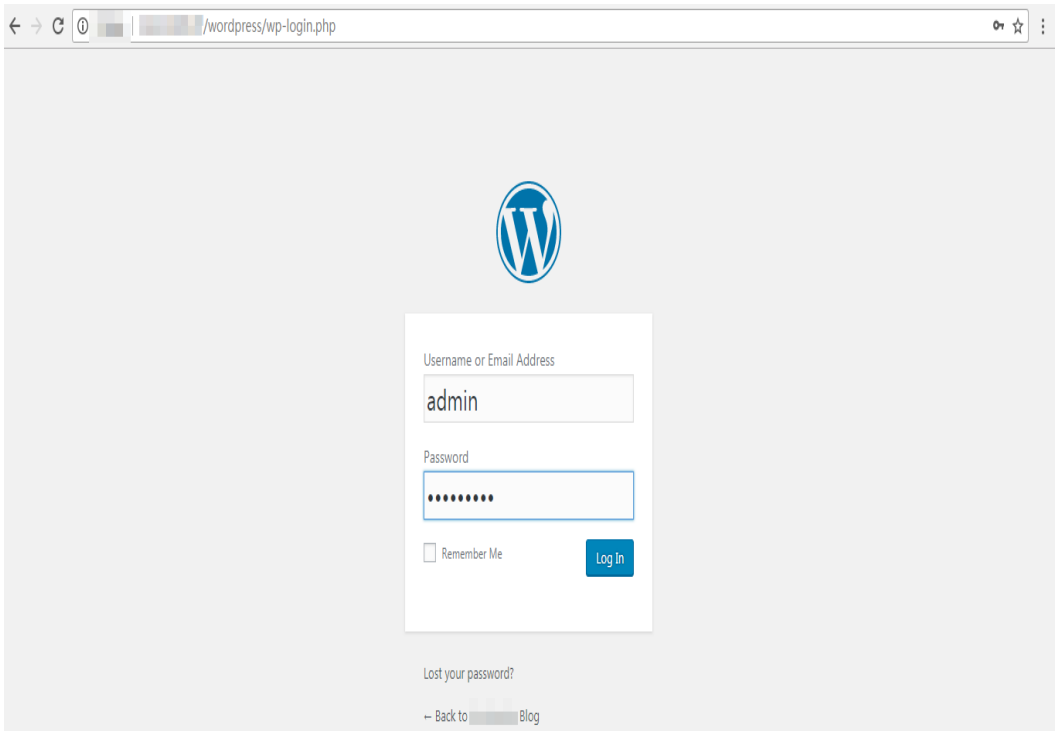
Step 6 Click **Log In** after WordPress has been successfully installed.

Figure 2-10 Successful installation



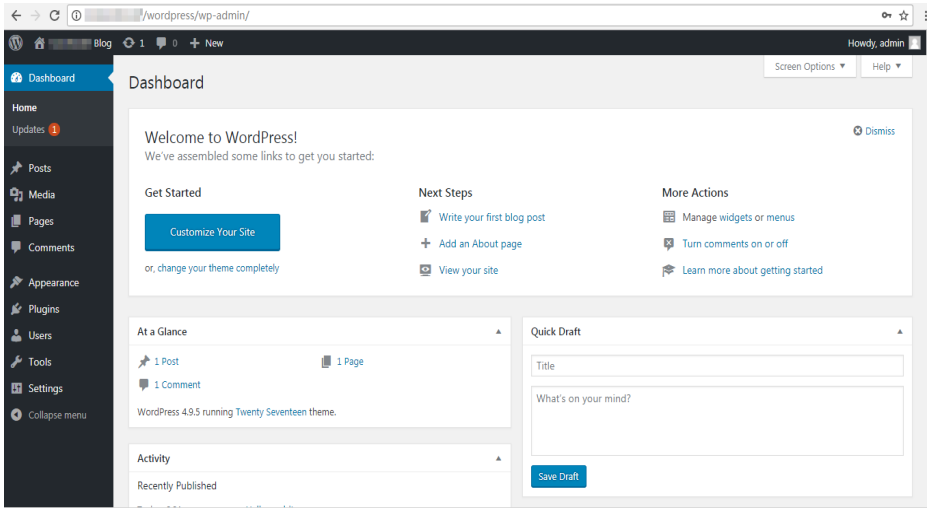
Step 7 Enter the user name and password on the displayed login page. Then, click **Log In**.

Figure 2-11 Logging in



Step 8 Check that WordPress has been deployed successfully.

Figure 2-12 Verification



----End

2.2 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 Under **Network**, click **Virtual Private Cloud**. The VPC console is displayed.

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

Step 5 On the displayed page, enter a VPC name, set **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 has been successfully created, return to network console.

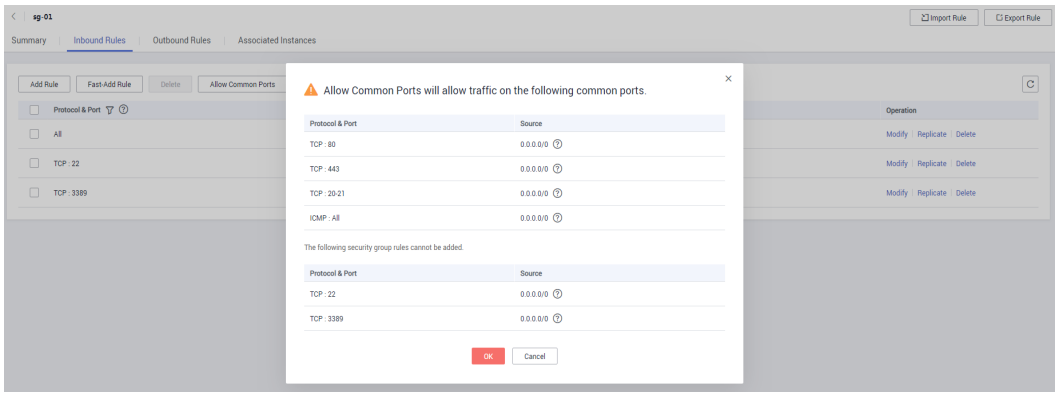
Step 6 On the network console, choose **Access Control** > **Security Groups** and click **Create Security Group**. The following uses sg-01 as an example.

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 2-13 Adding a security group rule



----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.

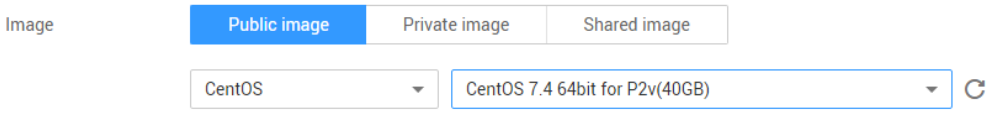
Step 3 Under **Computing**, click **Elastic Cloud Server**. The **Elastic Cloud Server** page is displayed.

Step 4 On the ECS console, buy an ECS. For details about billing information, see **Product Pricing Details**.

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 2-14**.

   Figure 2-14 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, you can view and manage it on the ECS console.

----End

Setting Up the LAMP Environment

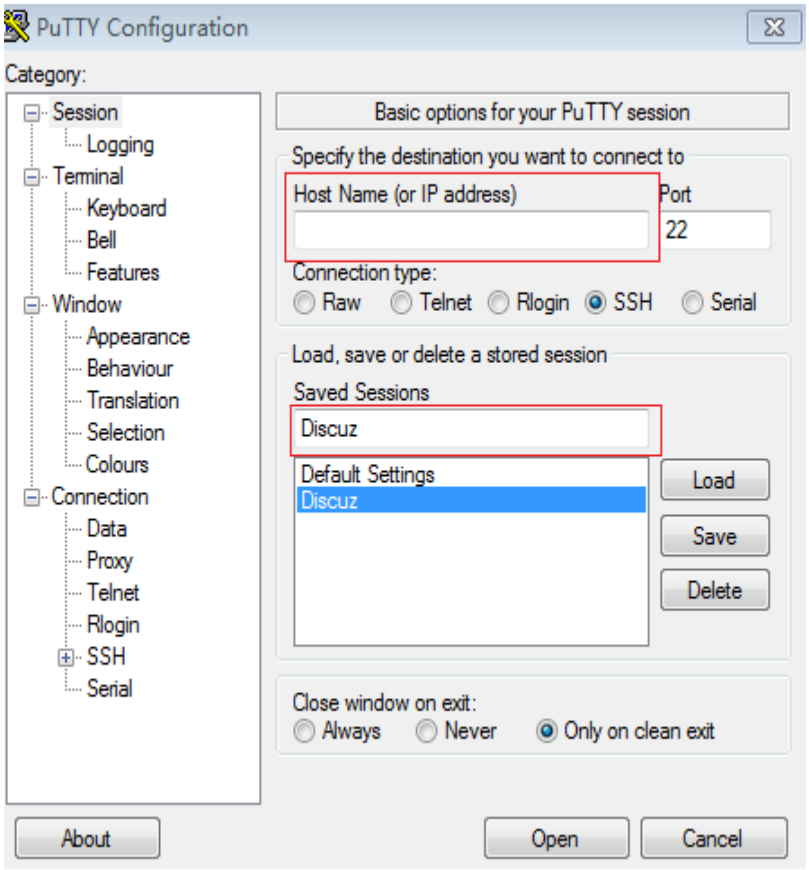
Step 1 Download the PuTTY client.

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 2-15**.

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 2-15 Configuring PuTTY



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 use 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, you can view or manage it on the **management console**.

----End

Installing Discuz!

Step 1 On the **Elastic Cloud Server** page, locate the target ECS and click **Remote Login** in the **Operation** column. Run the following command to download the Discuz! software:

wget http://download.comsenz.com/DiscuzX/3.3/Discuz_X3.3_SC_UTF8.zip

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 2 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 private IP address of DB instance rds-01.

- The database password is the root user password of DB instance rds-01.
- Enter administrator information.

Step 3 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

2.3 Downloading a Backup File and Restoring Data from It

You can **download a backup file** and restore data from it.

NOTICE

Backup data cannot be restored to local databases that run the Windows operating system.

Prerequisites

When you restore data from backup files to self-built MySQL databases, ensure that the target MySQL version is later than or equal to the original MySQL version.

During data restoration, run the following command to view the restoration process:

ps -ef | grep mysql

Procedure

Step 1 Download the qpress program and upload it to the ECS for installation.

Download **qpress-11-linux.x64.tar** from the **website** and upload it to the ECS.

tar -xvf *qpress-11-linux-x64.tar*

mv *qpress /usr/bin/*

Step 2 Download the XtraBackup software and upload it to the ECS for installation.

NOTICE

- For MySQL 5.6 and 5.7, download **XtraBackup 2.4.9** or later versions.
- For MySQL 8.0, download **XtraBackup 8.0** or later versions.

Download **XtraBackup** from the website and upload it to the ECS. The following uses percona-xtrabackup-24-2.4.9-1.el7.x86_64.rpm as an example.

rpm -ivh *percona-xtrabackup-24-2.4.9-1.el7.x86_64.rpm* **--nodeps --force**

Step 3 On the ECS, decompress the full backup file that has been downloaded.

1. Create a temporary directory **backupdir**.

   mkdir *backupdir*

2. Decompress the package.

   xbstream -x -p 4 < *./Full backup file.qp* -C *./backupdir/*

   - For MySQL 5.6 and 5.7, run **innobackupex --parallel 4 --decompress *./backupdir***.
   - For MySQL 8.0, run **xtrabackup --parallel 4 --decompress --target-dir=*./backupdir***.

   find *./backupdir/* **-name** *'*.qp'* **|** *xargs* **rm -f**

Step 4 Apply the log.

- For MySQL 5.6 and 5.7, run **innobackupex --apply-log *./backupdir***.
- For MySQL 8.0, run **xtrabackup --prepare --target-dir=./backupdir**.

Step 5 Back up data.

1. Stop MySQL database services.

   service *mysql* **stop**

   NOTE

   For MySQL 5.7, run the following command to stop MySQL database services:
   /bin/systemctl stop mysqld.service

2. Back up the original database directory.

   mv /var/lib/mysql/data /var/lib/mysql*/data_bak*

3. Create a new database directory and change the permissions.

   mkdir */var/lib/mysql/data*

   chown *mysql:mysql /var/lib/mysql/data*

Step 6 Copy the full backup file and modify the directory permissions.

- For MySQL 5.6 and 5.7, run **innobackupex --defaults-file=/etc/my.cnf --copy-back *./backupdir***.
- For MySQL 8.0, run **xtrabackup --defaults-file=/etc/my.cnf --copy-back --target-dir=./backupdir**.

chown -R *mysql:mysql /var/lib/mysql/data*

Step 7 Start the database.

service *mysql* **start**

NOTE

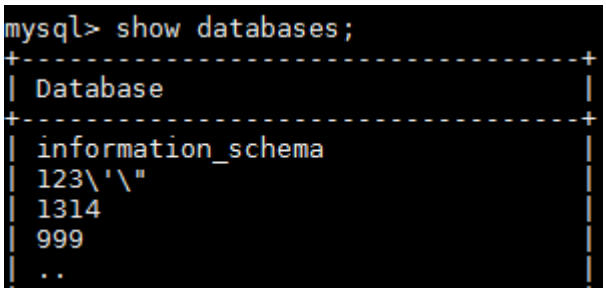
For MySQL 5.7, run the following command to start the database:
/bin/systemctl start mysqld.service

Step 8 Log in to the database and view the restoration result.

mysql -u -root

show databases

Figure 2-16 Viewing the restoration result



----End

2.4 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 MySQL disk write policy and data security. Different parameter values have different impacts on performance and security.

Table 2-1 Parameter description

Parameter	Allowed Value	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 **2**. For details, see **Parameter Description**.
sync_binlog	0 to 4,294,967,295	Sync binlog (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 2-2 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=*N* (*N*>1) and innodb_flush_log_at_trx_commit=2, the MySQL write operation achieves the optimal performance.

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

If the CPU usage is high or close to 100% when you use RDS for MySQL, data read/write processing is slow, connections cannot be obtained, and errors are reported, affecting your service running.

Solution 1

Analyze slow SQL logs (if any) and CPU usage to locate slow queries and then optimize them.

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

   For details on viewing MySQL logs, see section **Viewing and Downloading Slow Query Logs**.
2. View the CPU usage of your RDS DB instance to facilitate problem locating.

   For more about supported monitoring metrics, see **Configuring Displayed Metrics**.
3. Create read replicas to offload 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 RDS MySQL DB instances.

For details, see the private and public network connections in 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

```
*************************** 1. row ***************************
             Id: 16193
           User: rdsAdmin
           Host: localhost
             db: NULL
        Command: Query
           Time: 0
          State: starting
           Info: show full processlist
    Memory_used: 16424
Memory_used_by_query: 8208
       CPU_time: 294611
  Trx_Executed_Time: 0
  logical_page_read: 0
    disk_page_read: 0
```



```
   iops_limit_count: 0
*************************** 2. row ***************************
                Id: 16246
              User: root
              Host: ********
                db: test04_1
           Command: Query
              Time: 36
             State: creating table
              Info: create table test04_1.table_test_7(a int, b varchar(21632))
       Memory_used: 151378
Memory_used_by_query: 8208
          CPU_time: 128329
  Trx_Executed_Time: 0
  logical_page_read: 57
     disk_page_read: 0
   iops_limit_count: 0
*************************** 3. row ***************************
                Id: 16247
              User: root
              Host: ********
                db: test01_1
           Command: Query
              Time: 36
             State: creating table
              Info: create table test01_1.table_test_7(a int, b varchar(21632))
       Memory_used: 151378
Memory_used_by_query: 8208
          CPU_time: 127529
  Trx_Executed_Time: 0
  logical_page_read: 0
     disk_page_read: 0
   iops_limit_count: 0
*************************** 4. row ***************************
                Id: 16261
              User: root
              Host: ********
                db: test05_1
           Command: Query
              Time: 16
             State: creating table
              Info: create table test05_1.table_test_6(a int, b varchar(21632))
       Memory_used: 151378
Memory_used_by_query: 8208
          CPU_time: 150650
  Trx_Executed_Time: 0
  logical_page_read: 0
     disk_page_read: 0
   iops_limit_count: 0
*************************** 5. row ***************************
                Id: 16262
              User: root
              Host: ********
                db: test_1
           Command: Query
              Time: 14
             State: creating table
              Info: create table test_1.table_test_5838(a int, b varchar(21632))
       Memory_used: 151376
Memory_used_by_query: 8208
          CPU_time: 179596
  Trx_Executed_Time: 0
  logical_page_read: 0
     disk_page_read: 0
   iops_limit_count: 0
*************************** 6. row ***************************
                Id: 16266
              User: root
              Host: ********
```



```
                db: test02_1
           Command: Query
              Time: 8
             State: creating table
              Info: create table test02_1.table_test_9(a int, b varchar(21632))
       Memory_used: 151378
Memory_used_by_query: 8208
          CPU_time: 154582
  Trx_Executed_Time: 0
  logical_page_read: 0
    disk_page_read: 0
   iops_limit_count: 0
*************************** 7. row ***************************
                Id: 16269
              User: root
              Host: ********
                db: test03_1
           Command: Query
              Time: 6
             State: creating table
              Info: create table test03_1.table_test_11(a int, b varchar(21632))
       Memory_used: 151378
Memory_used_by_query: 8208
          CPU_time: 153918
  Trx_Executed_Time: 0
  logical_page_read: 0
    disk_page_read: 0
   iops_limit_count: 0
7 rows in set (0.00 sec)
```

Step 3 Use SQL tuning 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. On the top menu bar, choose **SQL Operation** > **SQL Window**. On the displayed page, click **SQL Tuning**.
3. On the **SQL Tuning** page, click **Add SQL Performance Tuning** to add a SQL tuning task. In the displayed dialog box, enter the SQL statement or upload the SQL file as required, and then click **OK**.
4. In the SQL tuning list, select a database and date range and click **Search** to filter tuning reports. Then, view tuning details in the **Operation** column.

   In the SQL tuning list, you can query the basic information about the tuning, preview the tuning status, view tuning details, obtain the optimization suggestions.

   NOTE

   - The SQL tuning function obtains the table structure and data distribution information (non-original). This information is used only for logic diagnosis and is not stored on the DAS server.
   - The process of obtaining the table structure and data distribution information may bring additional load to the DB instance, but has little impact on its performance.
   - Only the SQL tuning history is stored on the DAS server. You can delete it from the server permanently.

----End

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

MySQL uses metadata locking to manage concurrent access to database objects and to ensure data consistency. Since MySQL 5.5, metadata locks were introduced. Metadata locking applies to tables. Data cannot be read or written, resulting in SQL statements 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 **Service List**. Under **Database**, click **Relational Database Service** to go to the RDS console. The RDS console is displayed.

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

Alternatively, click the target DB instance on the **Instance Management** 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 On the top menu bar, choose **SQL Operation** > **SQL Window**.

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

show full processlist

Step 8 Check whether a large number of **Waiting for table metadata lock** are displayed in the **State** column, which indicates that SQL blocking occurs. 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*

----End

3 PostgreSQL

3.1 Viewing Slow Query Logs of 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 supports the following statement types:

- SELECT
- INSERT
- UPDATE
- DELETE
- CREATE
- DROP
- ALTER
- DO
- CALL
- COPY

Parameter Description

Table 3-1 Parameters related to 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.

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 **Service List**. Under **Database**, click **Relational Database Service** to go to the RDS console.

Step 4 On the **Instance Management** 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

3.2 Performing Basic Operations Using pgAdmin

3.2.1 Connecting to a PostgreSQL DB Instance

pgAdmin is client management software that designs, maintains, and manages PostgreSQL databases. It allows you to connect to specific databases, create tables, and run various simple and complex SQL statements. It supports Windows, Linux, Mac OS X, 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 RDS for PostgreSQL and create databases and tables. The 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 a PostgreSQL DB 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

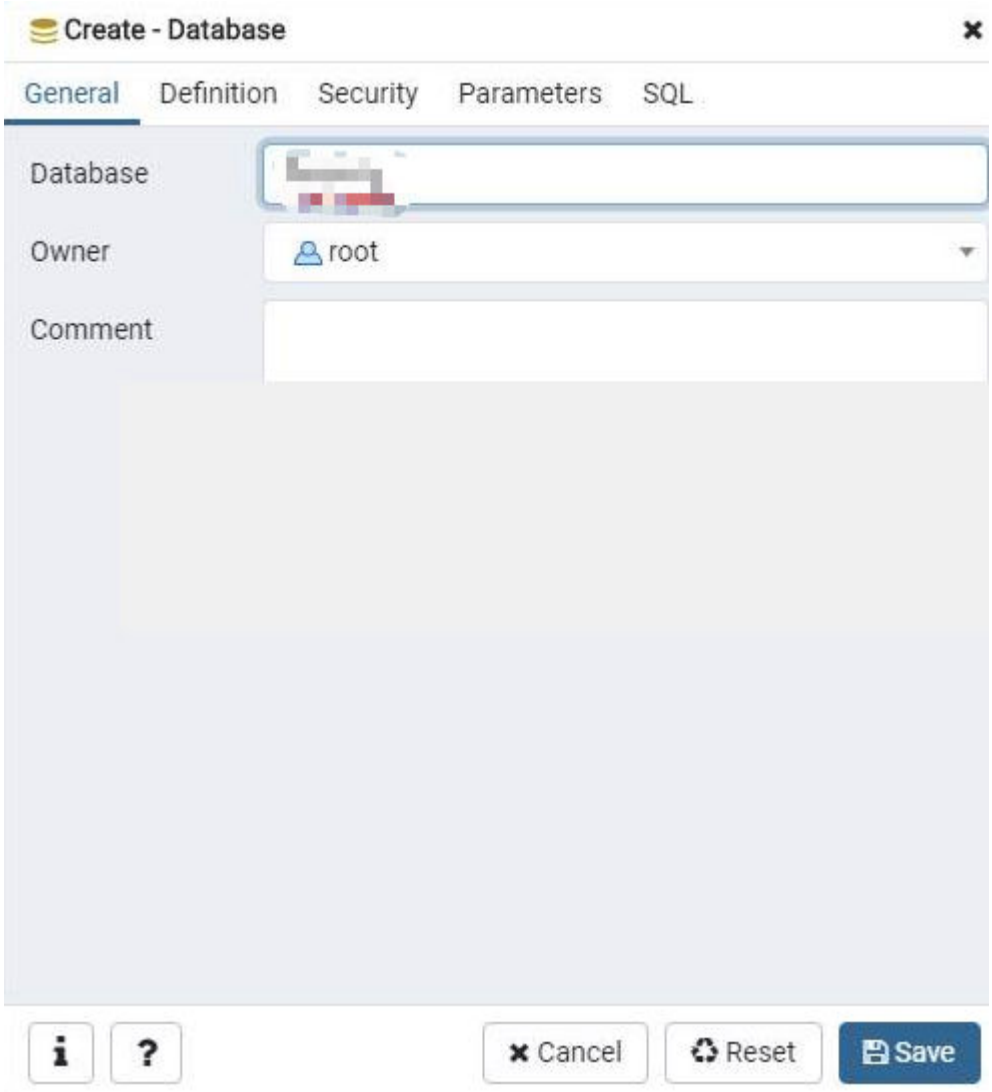
A PostgreSQL DB instance has been connected.

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 3-1 Creating a database



----End

3.2.3 Creating a Table

Scenarios

This section describes how to create a table.

Prerequisites

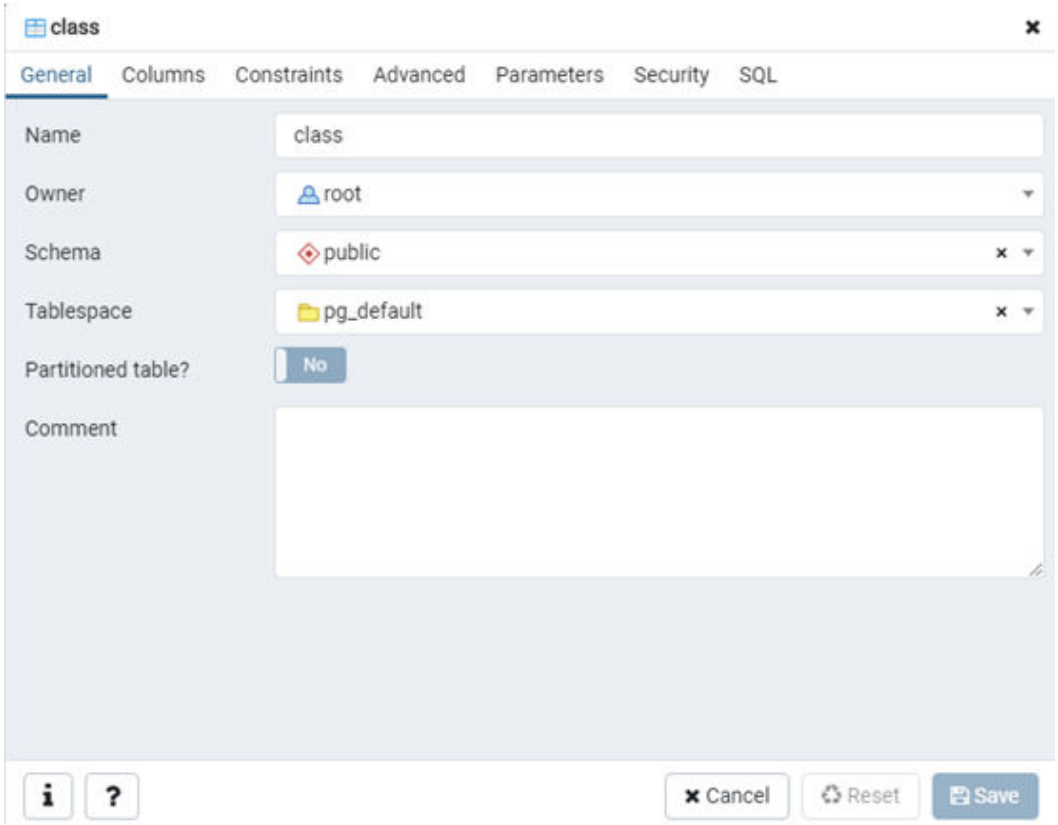
A PostgreSQL DB instance has been connected.

Procedure

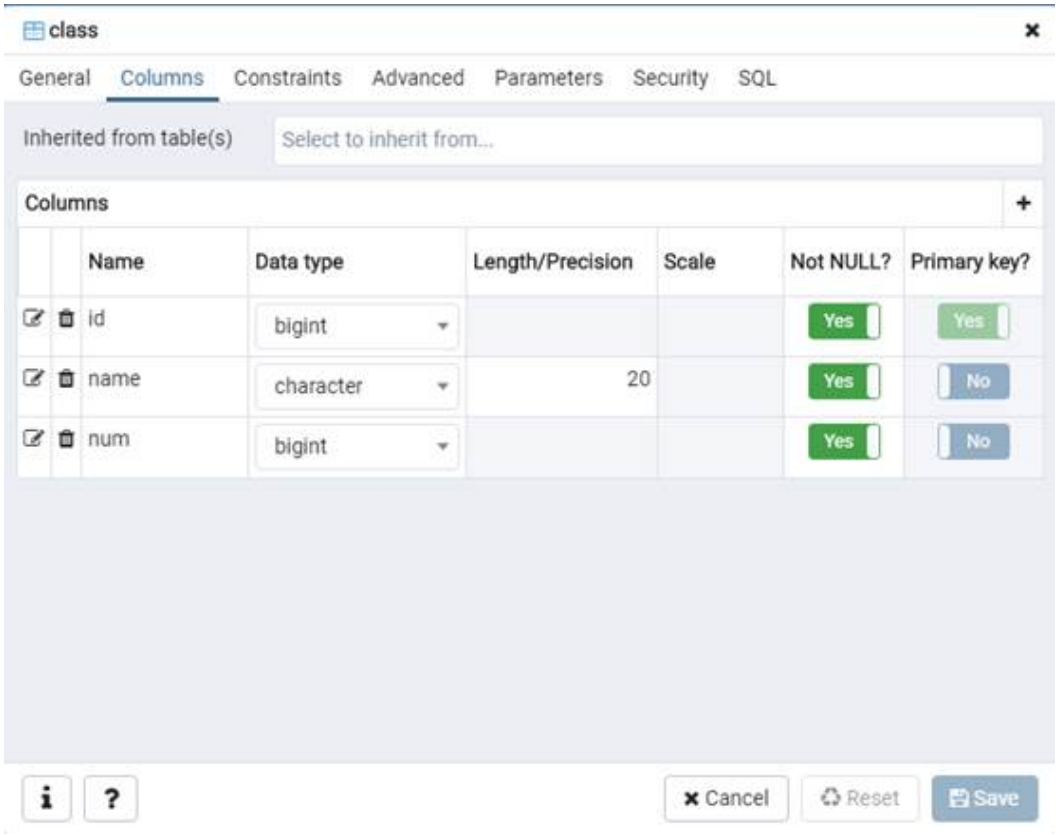
Step 1 In the navigation pane on the left, right-click the target table and choose **Create** > **Table** from the shortcut menu.

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

Figure 3-2 Basic information



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



----End

3.2.4 Using the Query Tool to Run SQL Statements

Scenarios

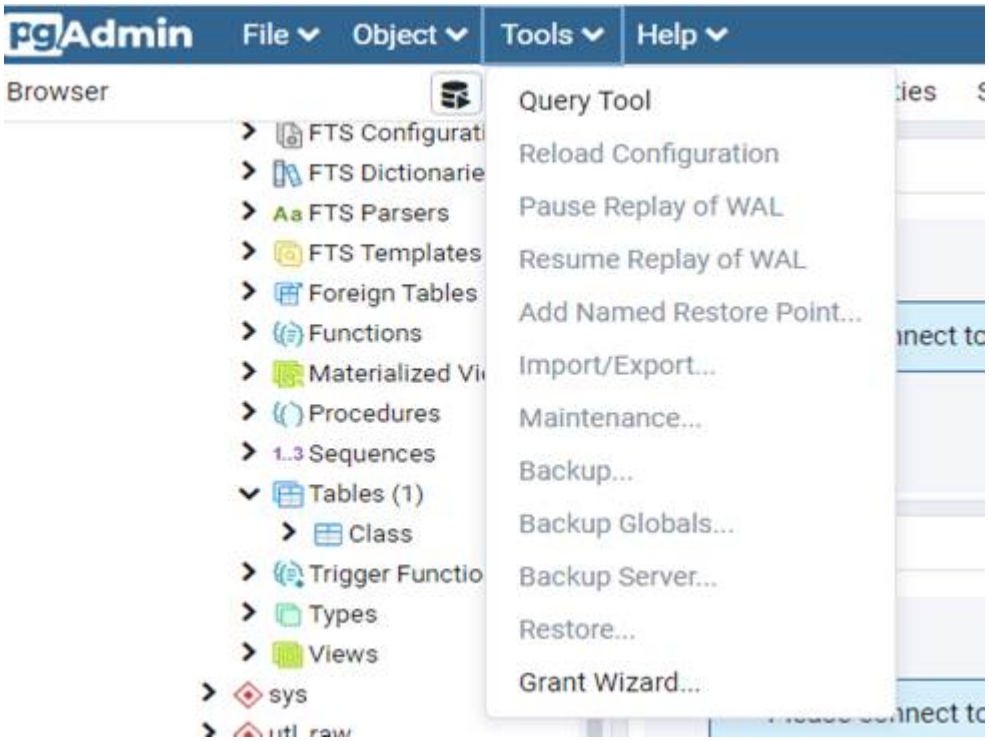
This topic describes how to use Query Tool to run SQL commands.

Prerequisites

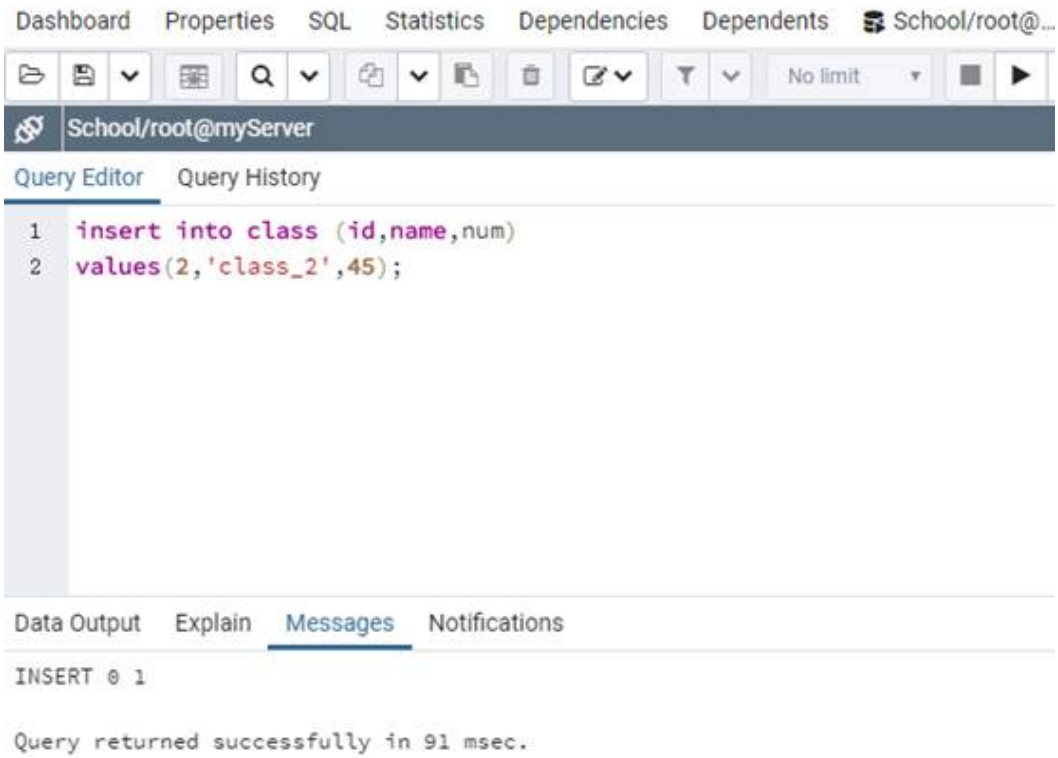
A PostgreSQL DB instance has been connected.

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.

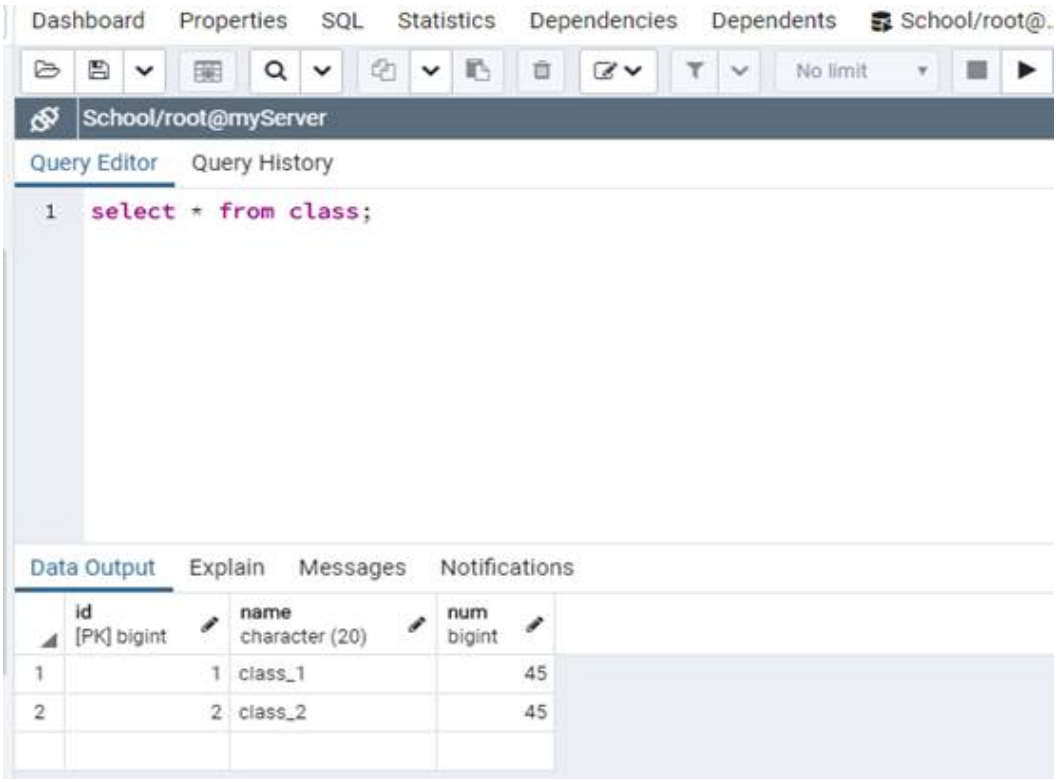


----End

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.



----End

3.2.5 Monitoring Databases

Scenarios

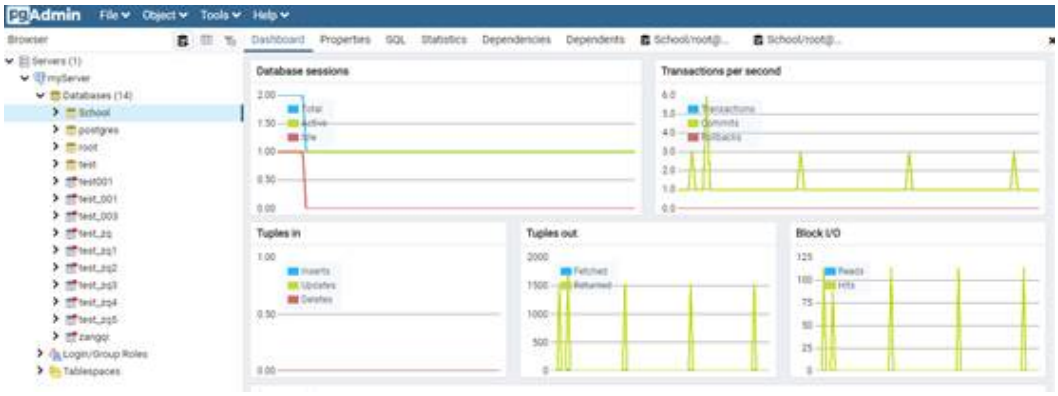
This section describes database monitoring.

Prerequisites

A PostgreSQL DB instance has been connected.

pgAdmin Monitoring

Step 1 In the navigation pane on the left, select a database and click the **DashBroad** 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

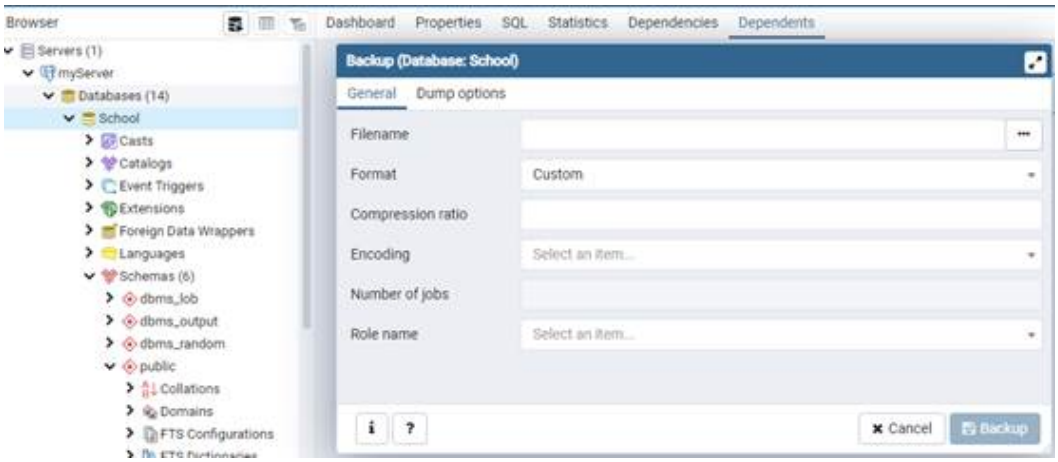
A PostgreSQL DB instance has been connected.

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**.

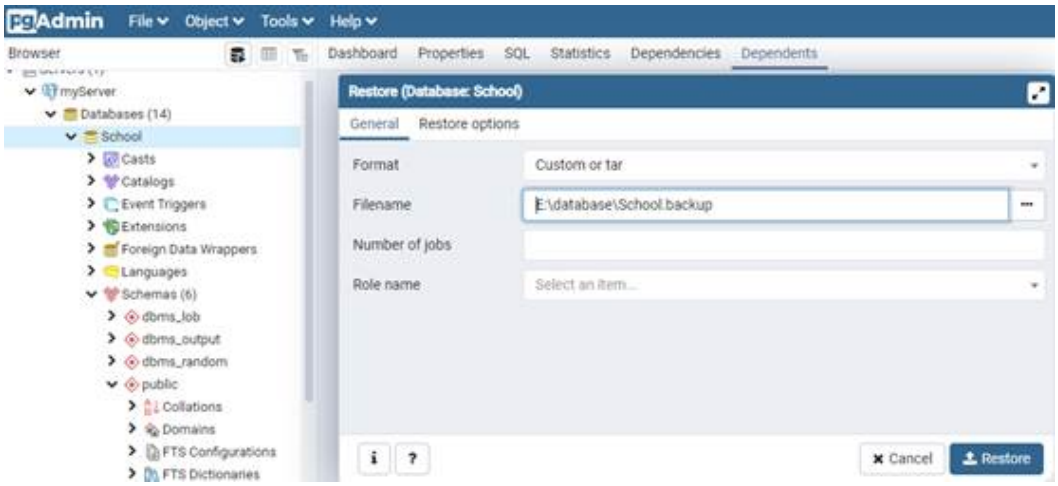


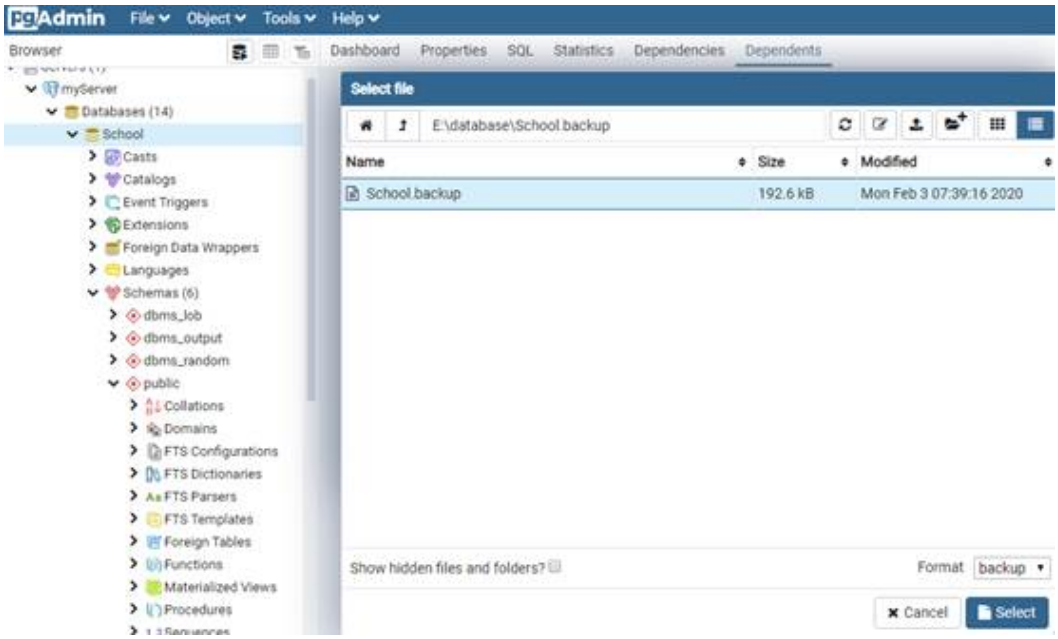
----End

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**.





----End

RDS Backup and Restoration

RDS supports DB instance backup and restoration to ensure data reliability. For more information, see **Working with Backups**.

3.3 Read/Write Splitting with PostgreSQL Proxy

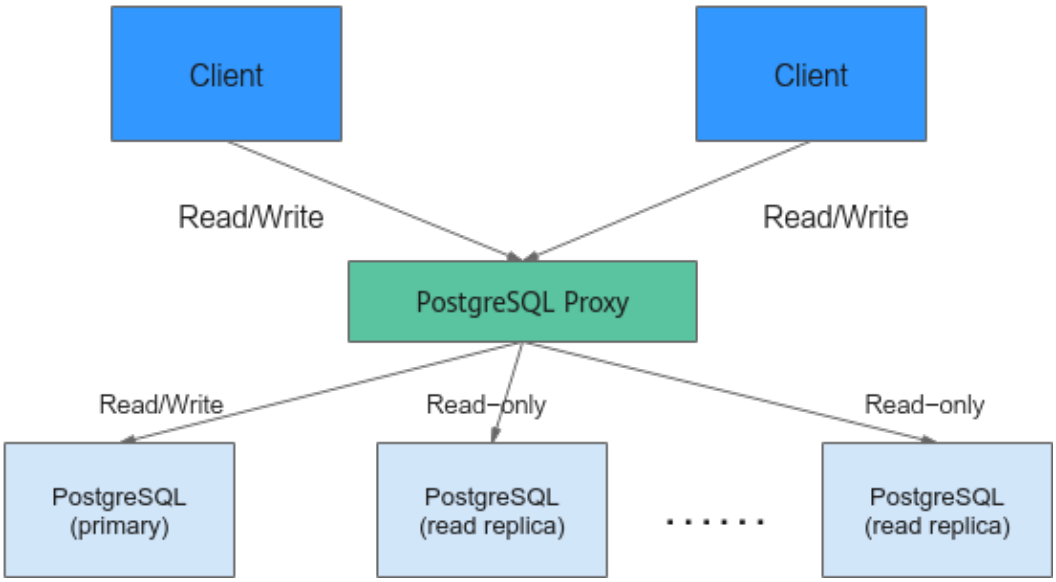
Scenarios

If the processing capability of a single PostgreSQL DB instance reaches the bottleneck and cannot bear higher read and write pressure, services will be affected. In read-intensive scenarios, you can create **read replicas** to offload read pressure from primary DB instances through PostgreSQL proxy so that primary DB instances are dedicated to processing write requests Read replicas not only elastically scale the read capability of primary DB instances, but also increase the number of concurrent database connections.

How It Works

PostgreSQL Proxy is a middleware that implements read/write splitting for PostgreSQL databases. It sits between the PostgreSQL server and PostgreSQL client. You application connects to the PostgreSQL proxy, as shown in **Figure 3-3**.

Figure 3-3 PostgreSQL Proxy principle



Advantages

PostgreSQL Proxy has the following advantages:

- It is fully compatible with the PostgreSQL protocol and is widely used.
- A unified read/write splitting address is provided to simplify the complexity of application development and database O&M.

  Your application connects to the PostgreSQL Proxy through the unified read/write splitting address and the PostgreSQL Proxy splits requests to the primary DB instance and read replicas, therefore delivering better performance and application scalability.
- Read replicas can be added online

  without any application modifications.
- Read weights can be set online to suit requirements in different scenarios.
- The heartbeat of read replicas is automatically and periodically detected to improve database availability.

  In case of a failure, such as a breakdown or a long latency that exceeds the threshold, PostgreSQL Proxy will no longer send read requests to the faulty read replica until it is recovered.

Enabling Read/Write Splitting

Step 1 Learn about **billing** and **constraints** first.

Step 2 Enable read/write splitting by referring to **Enabling Read/Write Splitting**.

Step 3 Configure the delay threshold and read weight by referring to **Configuring Delay Threshold and Distributing Read Weight**.

Step 4 Verify the read/write splitting effect by referring to **Testing Read/Write Splitting Performance**.

----End

4 Microsoft 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.

You can restore from a backup to a new DB instance only when your account balance is greater than or equal to ¥0.

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-Built SQL Server Database on an ECS to an RDS Microsoft SQL Server DB Instance

Scenarios

- You have created a Microsoft SQL Server database on an ECS.
- The self-built SQL Server database version on the ECS cannot be later than the version of the RDS Microsoft 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

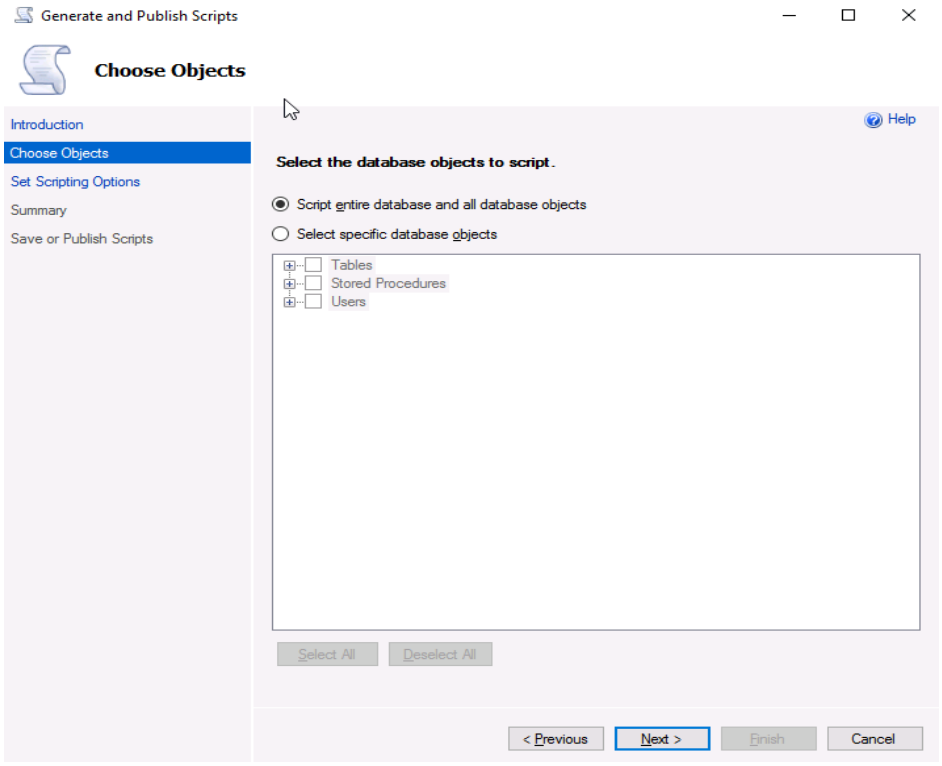
The Microsoft SQL Server installed on the ECS must be Standard or Enterprise Edition. It is recommended that the Microsoft SQL Server version be the same as the RDS DB instance version.

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

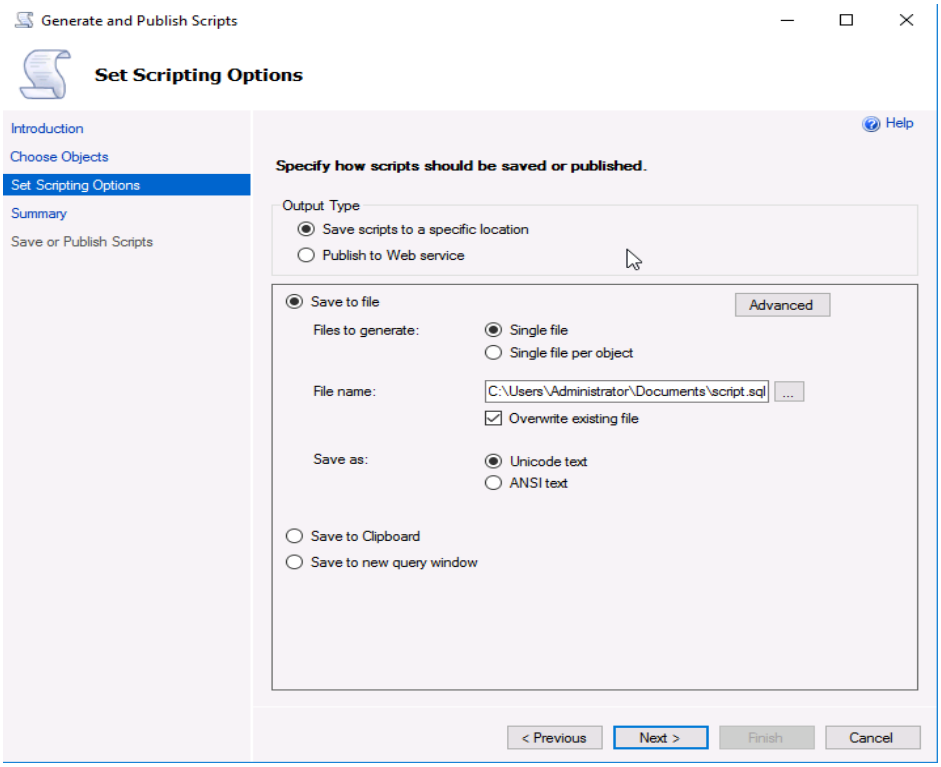


3. On the **Set Scripting Options** page, specify a directory for saving the script.

   NOTE

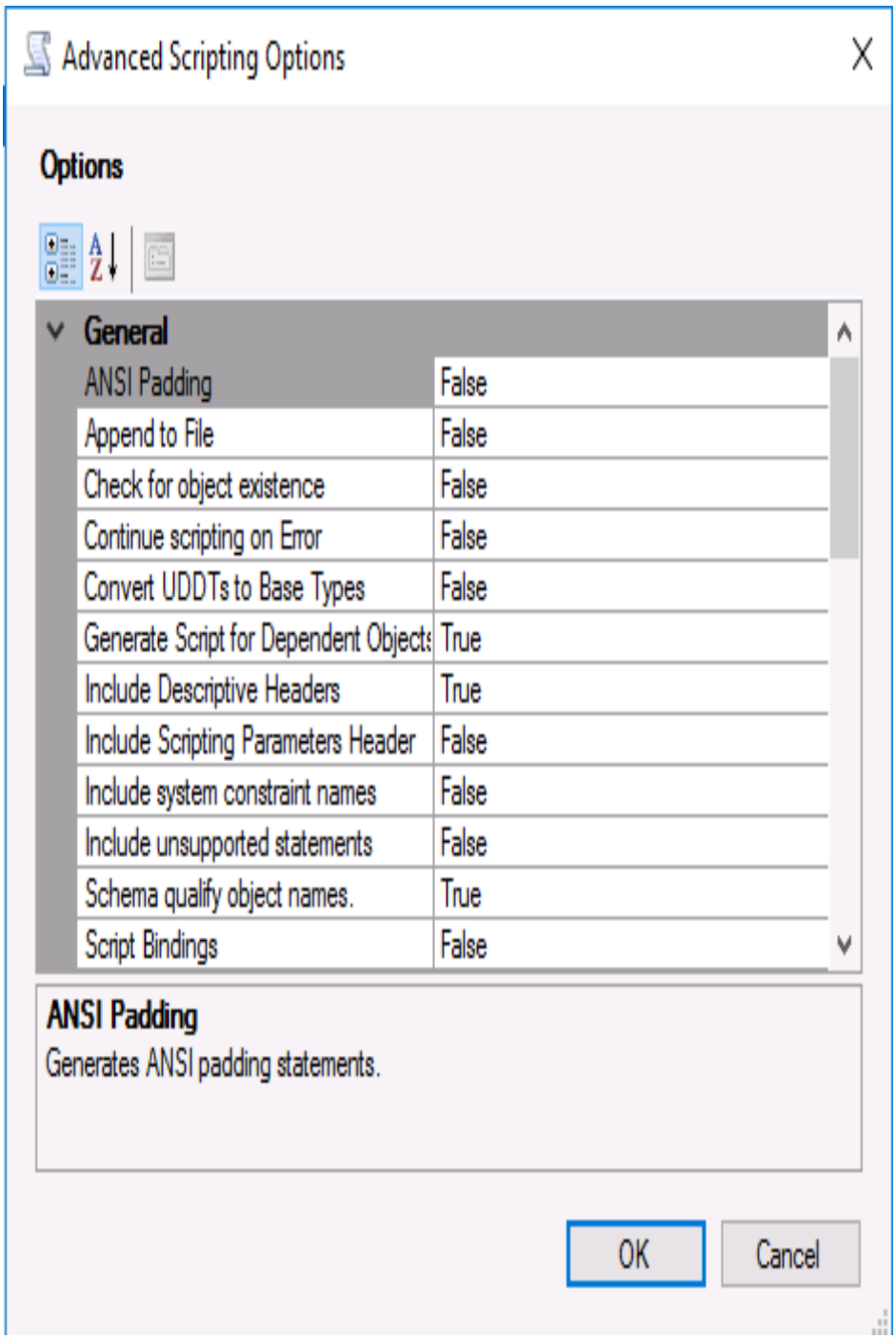
   You are advised to save the script locally and generate an SQL script for execution.

Figure 4-2 Specifying a directory for saving the script



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



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 whose data needs to be exported and choose **Tasks** > **Export Data**.
2. On the **Choose a Data Source** page, select a data source and click **Next**.

   NOTE

   Data source: Select **SQL Server Native Client** or specify the option based on your data source type.

   Server name: Enter the IP address and port number of the source DB instance. If the source DB instance is local, enter the DB instance name or **localhost**.

   Authentication: Select **Use SQL Server Authentication**. Then, set **User name** to **rdsuser**, and **Password** to the password of **rdsuser**.
3. On the **Choose a Destination** page, select a destination database and click **Next**. On the displayed **Specify Table Copy or Query** page, select **Copy data from one or more tables or views**. Then, click **Next**.

   NOTE

   The destination database indicates the database to which your data is imported.
4. On the **Select Source Tables and Views** page, select the tables and views that you want to export. Then, click **Edit Mappings**. In the displayed dialog box, select **Enable identity insert** and edit mappings based on your requirements.
5. Click **Finish** to export and import data.

   You can view the progress. About 4,000 rows can be processed per second.

----End

4.3 Modifying Parameters in a Parameter Template

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 **Service List**. Under **Database**, click **Relational Database Service** to go to the RDS console.

Step 4 On the **Instance Management** 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 Template Management** 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 Template Management** 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 an 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 a Microsoft SQL Server DB instance runs on a computer with more than one microprocessor or CPU, Microsoft 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 **View 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.

For details about the fields, visit **sys.dm_exec_query_stats (Transact-SQL)**.

----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 RDS console. On the **Instance Management** page, click the target DB instance name.

Step 2 On the **EIPs** page, click **Bind EIP**. In the displayed dialog box, select an EIP and click **OK**.

Step 3 Install the SSMS client locally and use the EIP to connect to the RDS DB instance.

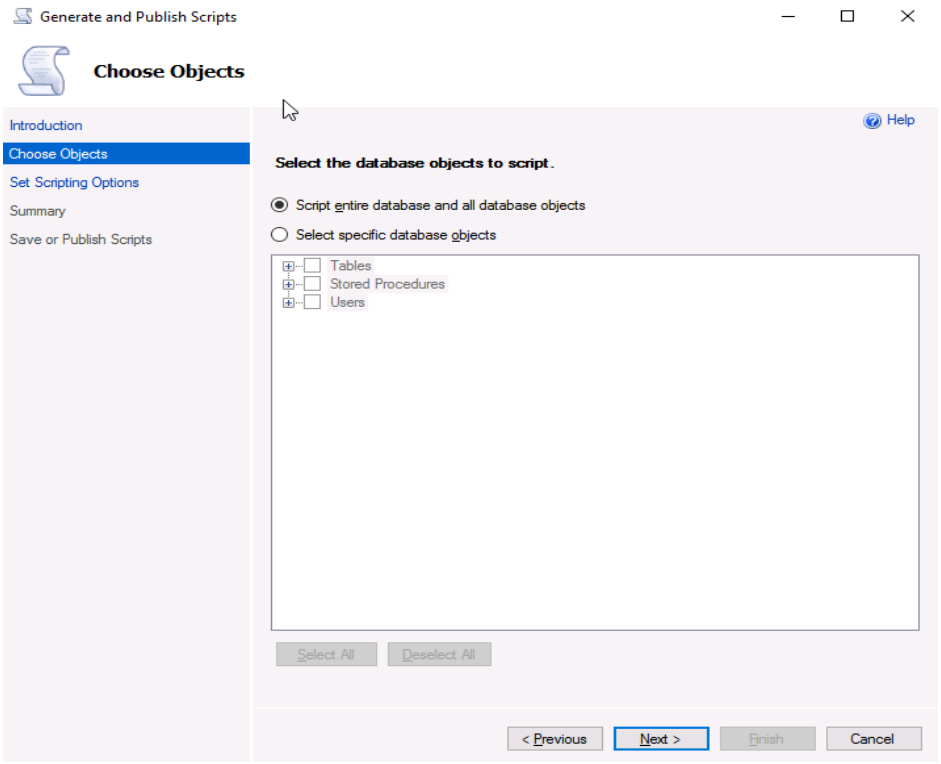
NOTE

Click **here** to download the SSMS client.

Step 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 4-4**. Then, click **Next**.

Figure 4-4 Choosing objects

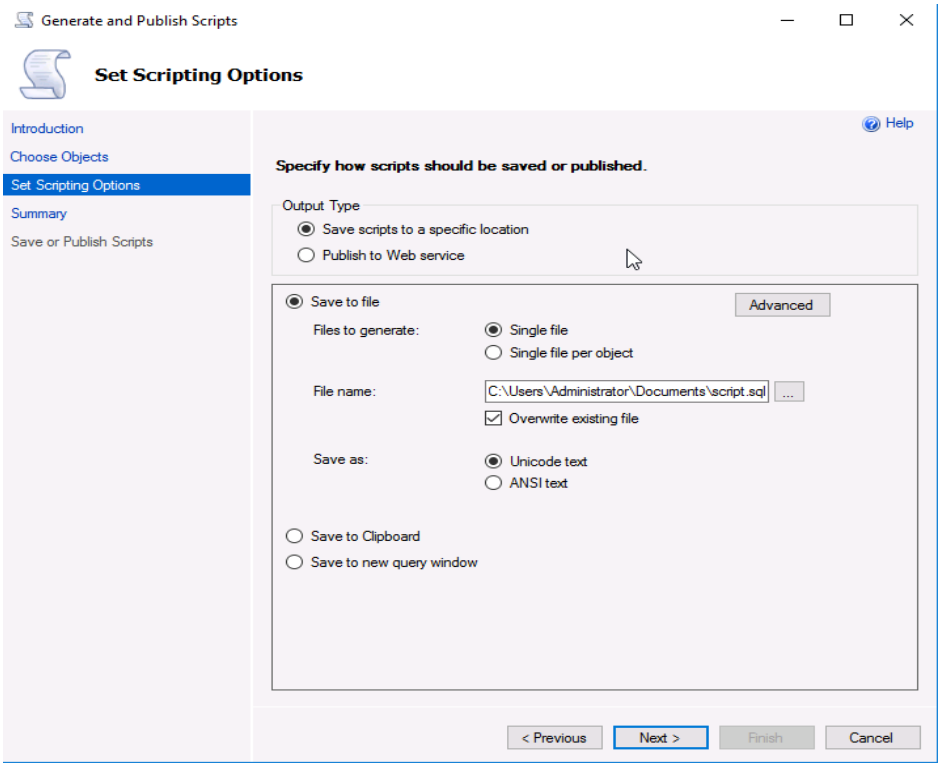


3. On the **Set Scripting Options** page, specify a directory for saving the script.

   NOTE

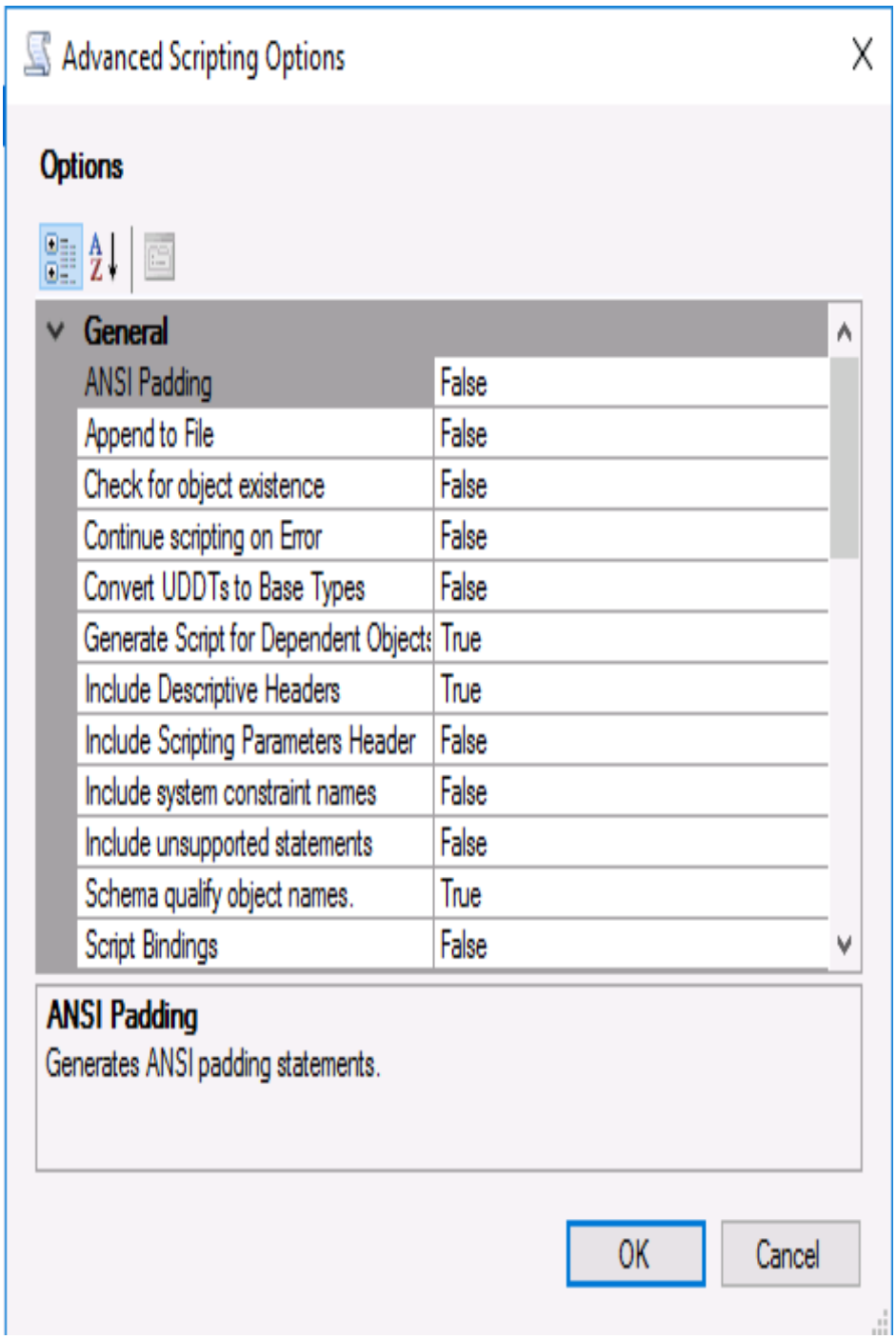
   You are advised to save the script locally and generate an SQL script for execution.

Figure 4-5 Specifying a directory for saving the script



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



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 whose data needs to be exported and choose **Tasks** > **Export Data**.
2. On the **Choose a Data Source** page, select a data source and click **Next**.

   NOTE

   Data source: Select **SQL Server Native Client** or specify the option based on your data source type.

   Server name: Enter the IP address and port number of the source DB instance. If the source DB instance is local, enter the DB instance name or **localhost**.

   Authentication: Select **Use SQL Server Authentication**. Then, set **User name** to **rdsuser**, and **Password** to the password of **rdsuser**.
3. On the **Choose a Destination** page, select a destination database and click **Next**. On the displayed **Specify Table Copy or Query** page, select **Copy data from one or more tables or views**. Then, click **Next**.

   NOTE

   The destination database indicates the database to which your data is imported.
4. On the **Select Source Tables and Views** page, select the tables and views that you want to export. Then, click **Edit Mappings**. In the displayed dialog box, select **Enable identity insert** and edit mappings based on your requirements.
5. Click **Finish** to export and 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

You want to create and manage a subaccount of **rdsuser** for an RDS Microsoft SQL Server DB instance.

This section provides a script to help you create a subaccount of **rdsuser** and manage it. **Table 4-2** lists the permissions of **rdsuser**.

Permissions of rdsuser

Table 4-2 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 SQLAentUserRole
		Model: Public
		Rdsadmin: Public
		OtherDB: Db_Owner

Creating a Subaccount

You can use the following script to quickly create a subaccount of **rdsuser**:

NOTE

The following script applies only to Microsoft SQL Server 2014 and later versions. If you want to apply the script to Microsoft SQL Server 2008 R2, you need to modify the script based on your requirements.

```
use [master]
```



```
DECLARE @DBName NVARCHAR(128)
DECLARE @SQL NVARCHAR(max)
DECLARE @Login_Name nvarchar(128)
DECLARE @Login_Password nvarchar(128)

set @Login_Name = 'TestLogin3'        --change your login name
set @Login_Password = '1qaz!QAZ'

      SET @SQL='
         USE [master]

CREATE LOGIN  '+@Login_name+' WITH PASSWORD=N'''+ @Login_Password +''',
DEFAULT_DATABASE=[master], CHECK_EXPIRATION=OFF, CHECK_POLICY=OFF

         alter server role [processadmin] add member  '+@Login_name+'
         alter server role [setupadmin] add member '+@Login_name+'
         GRANT VIEW SERVER STATE TO '+@Login_name+' WITH GRANT OPTION
         GRANT VIEW ANY DEFINITION TO '+@Login_name+' WITH GRANT OPTION
         GRANT VIEW ANY DATABASE TO '+@Login_name+' WITH GRANT OPTION
         GRANT CREATE ANY DATABASE TO '+@Login_name+' WITH GRANT OPTION
         GRANT ALTER SERVER STATE TO '+@Login_name+' WITH GRANT OPTION
         GRANT ALTER TRACE TO '+@Login_name+' WITH GRANT OPTION
         GRANT ALTER ANY SERVER ROLE TO '+@Login_name+' WITH GRANT OPTION
         GRANT ALTER ANY LOGIN TO '+@Login_name+' WITH GRANT OPTION
         GRANT ALTER ANY CONNECTION TO '+@Login_name+' WITH GRANT OPTION
         GRANT CONNECT SQL TO '+@Login_name+' WITH GRANT OPTION
         GRANT VIEW SERVER STATE TO '+@Login_name+' WITH GRANT OPTION
      '
print @SQL
      exec (@SQL)

      SET @SQL='
         use [msdb]
         if exists(select top 1 1 from sys.sysusers where name = '''+ @Login_Name +''')
         begin
            ALTER USER '+@Login_name+' with login = '+@Login_name+';
         end
         else
         begin
            CREATE USER '+@Login_name+' FOR LOGIN '+@Login_name+';
         end
         ALTER ROLE [SQLAgentUserRole] ADD MEMBER '+@Login_name+'
         GRANT ALTER ON ROLE::[SQLAgentUserRole] TO '+@Login_name+' WITH GRANT OPTION
         GRANT ALTER ANY USER TO  '+@Login_name+'  WITH GRANT OPTION
         GRANT EXEC ON msdb.dbo.sp_delete_database_backuphistory TO   '+@Login_name+'    WITH
GRANT OPTION
         GRANT EXEC ON msdb.dbo.sp_purge_jobhistory TO   '+@Login_name+'    WITH GRANT OPTION

         GRANT SELECT ON msdb.dbo.sysjobs TO '+@Login_name+'  WITH GRANT OPTION;
         GRANT SELECT ON msdb.dbo.sysschedules TO '+@Login_name+'  WITH GRANT OPTION;
         GRANT SELECT ON msdb.dbo.sysjobsteps TO '+@Login_name+'  WITH GRANT OPTION;
         GRANT SELECT ON msdb.dbo.sysjobhistory TO '+@Login_name+'  WITH GRANT OPTION;
         GRANT SELECT ON msdb.dbo.syscategories TO '+@Login_name+'  WITH GRANT OPTION;
         GRANT SELECT ON msdb.dbo.sysjobschedules TO '+@Login_name+'  WITH GRANT OPTION;
      '
print @SQL
      exec (@SQL)

SET @SQL='
         use [tempdb]
         if exists(select top 1 1 from sys.sysusers where name = '''+ @Login_Name +''')
         begin
            ALTER USER '+@Login_name+' with login = '+@Login_name+';
         end
         else
         begin
            CREATE USER '+@Login_name+' FOR LOGIN '+@Login_name+';
         end
         GRANT CONTROL TO  '+@Login_name+'
```



```
	'
print @SQL
exec (@SQL)

	declare DBName_Cursor cursor  for
	select quotename(name) from sys.databases where database_id > 4 and state = 0
	and name not like '%$%'
	and name <> 'rdsadmin'
	open DBName_Cursor
	   fetch next from DBName_Cursor into @DBName
	   WHILE @@FETCH_STATUS = 0
	   begin
	      SET @SQL='        USE ' + (@DBName) + '
	                if exists(select top 1 1 from sys.sysusers where name = '''+ @Login_Name +''')
	                begin
	                   ALTER USER '+@Login_name+' with login = '+@Login_name+';
	                   ALTER ROLE [db_owner] ADD MEMBER '+@Login_name+';
	                end
	                else
	                begin
	                   CREATE USER '+@Login_name+' FOR LOGIN '+@Login_name+';
	                   ALTER ROLE [db_owner] ADD MEMBER '+@Login_name+';
	                end
	                '
print @SQL
	      EXEC (@SQL)
	      fetch next from DBName_Cursor into @DBName
	   end
	close DBName_Cursor
	deallocate DBName_Cursor
```

4.7 Migrating Microsoft SQL Server Data Using DRS

You can use DRS to migrate on-premises databases to RDS for SQL Server with no downtime.

On the **Instance Management** page, you can click the target DB instance. On the displayed **Basic Information** page, click **Migrate Database** in the upper right corner of the page.

Microsoft SQL Server supports both online and backup migration.

- For details about online migration, see **Before You Start** in the *Data Replication Service User Guide*.
- For details about backup migration, see **Before You Start** in the *Data Replication Service User Guide*.

4.8 Create tempdb Files

Scenarios

The tempdb system database is a global resource that is available to all users connected to the instance of SQL Server or connected to 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 DB instance using SQL Server 2008, 2012, or 2014 Edition has one tempdb file, each instance using SQL Server 2016 Edition has four tempdb files, and each instance using SQL Server 2017 Edition has eight tempdb files.
- Each DB instance has only one log file no matter which SQL Server Edition they use.

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

- SQL Server Management Studio has been installed. The detailed procedure is provided in **How Can I Install SQL Server Management Studio?**
- A SQL Server 2014 Enterprise Edition instance with 32 vCPUs has been created. The detailed procedure is provided in **Buying an RDS Microsoft SQL Server 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

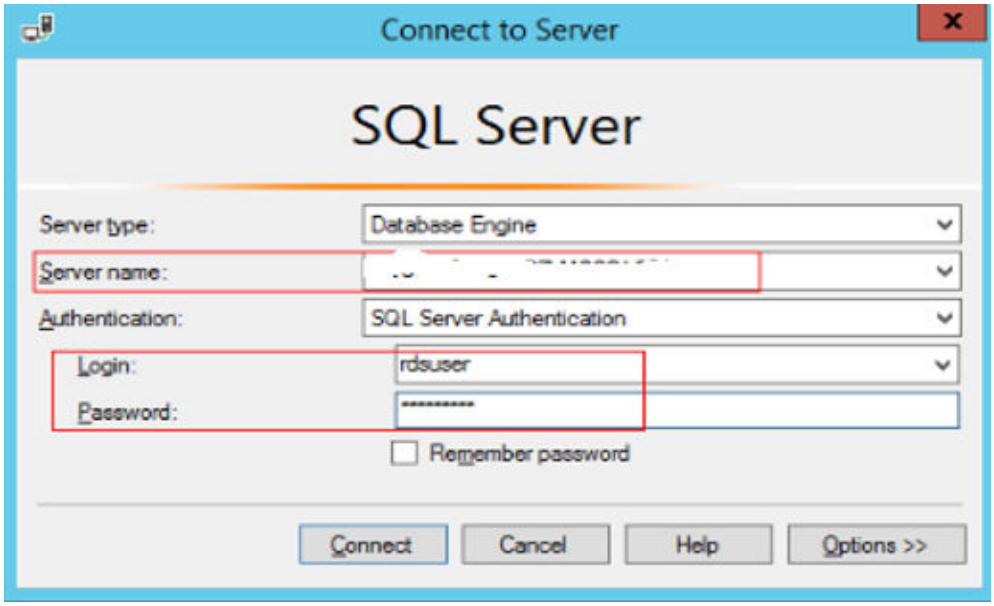


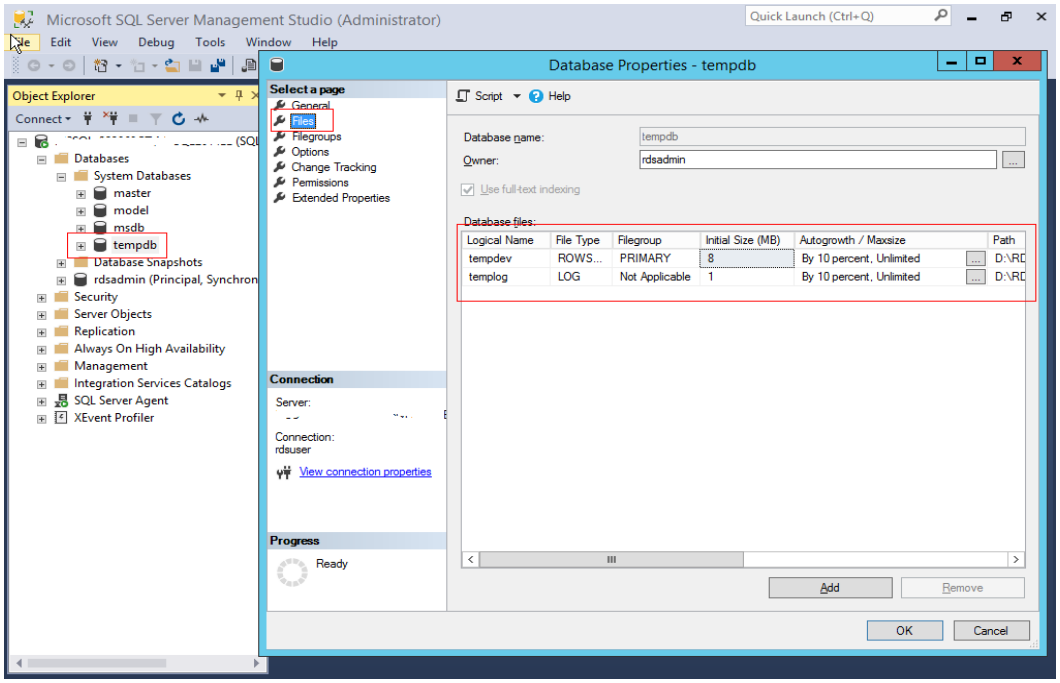
Table 4-3 Parameter description

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



- 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

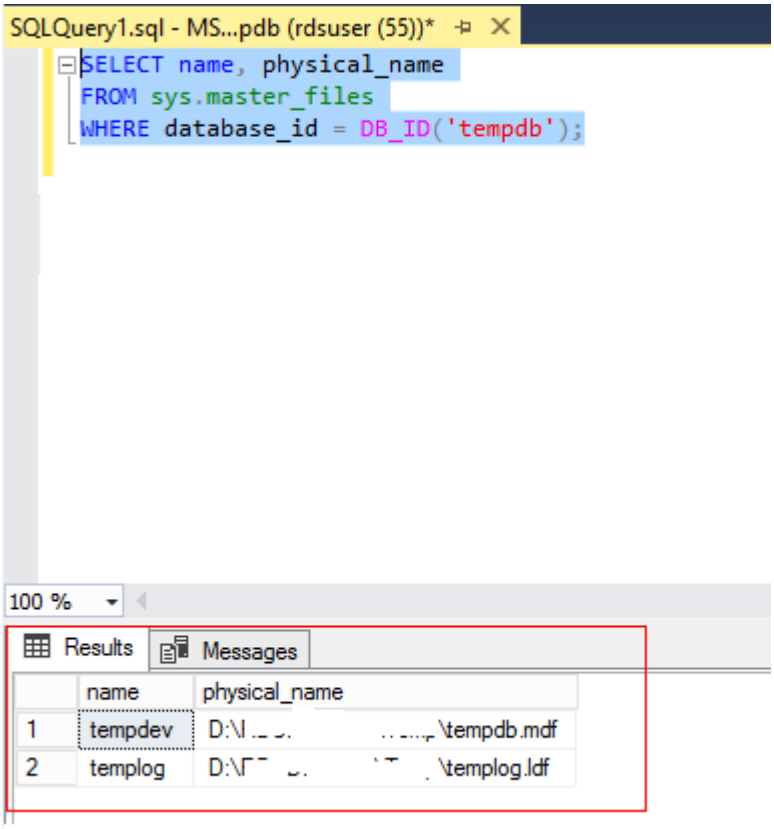
Step 4 Run the following statements to query the tempdb file name of the current DB instance:

SELECT name, physical_name

FROM sys.master_files

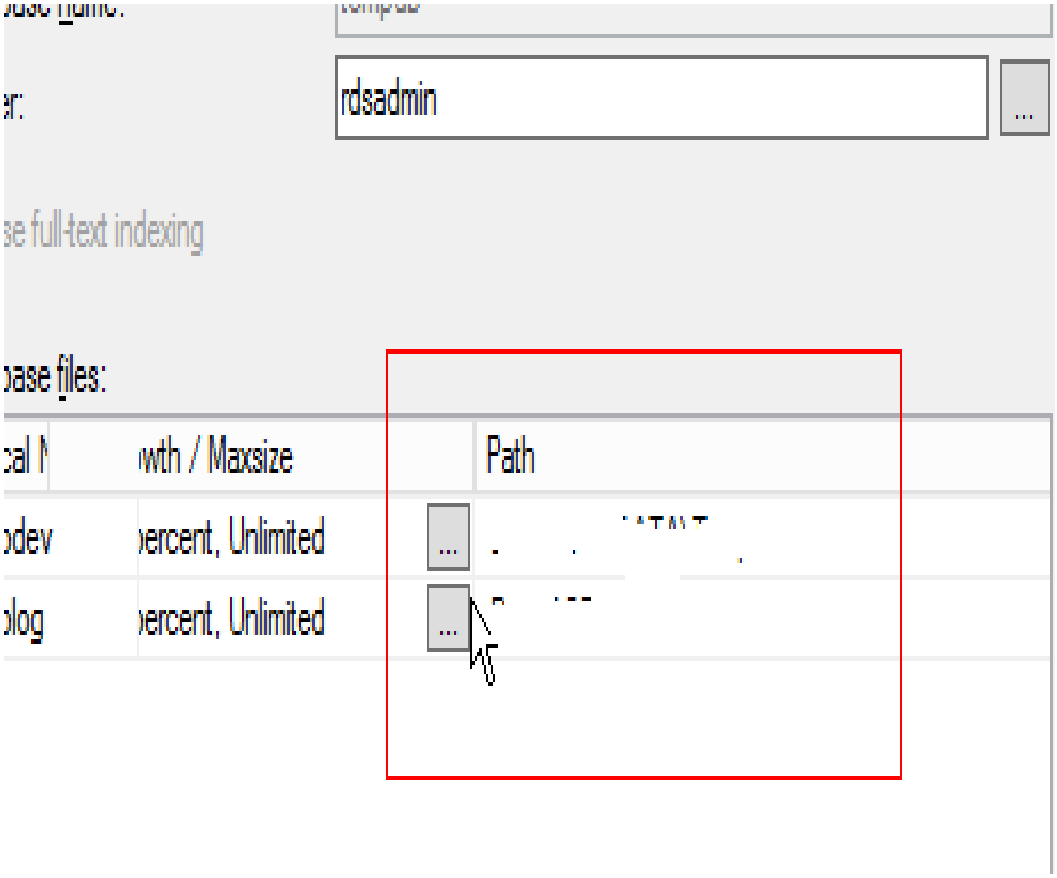
WHERE database_id = DB_ID('tempdb');

Figure 4-9 Viewing tempdb file names



Step 5 On the **Files** tab in **Step 3**, view the paths of tempdb files.

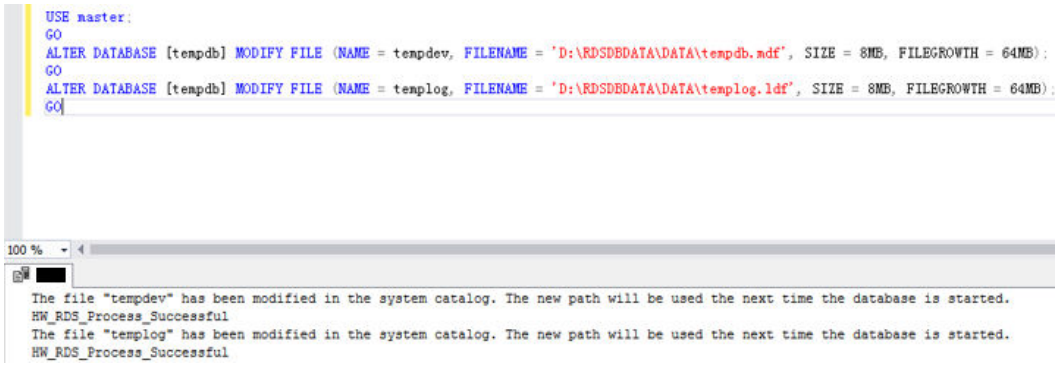
Figure 4-10 Viewing tempdb paths



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



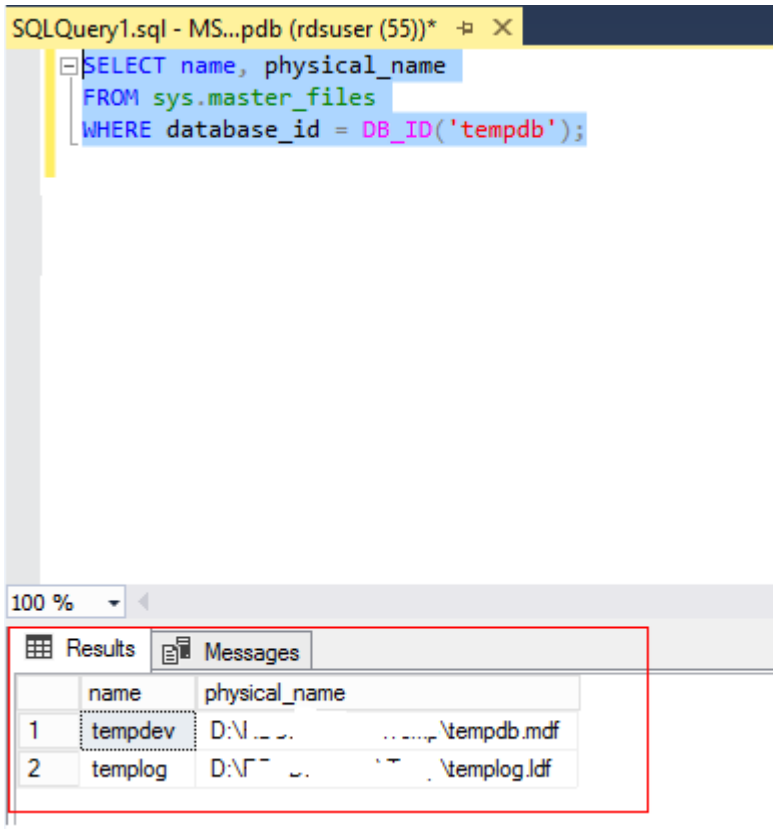
Step 7 On the **Instance Management** page of the RDS console, locate the target DB instance and choose **More** > **Restart** in the **Operation** column to reboot the DB instance.

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



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)

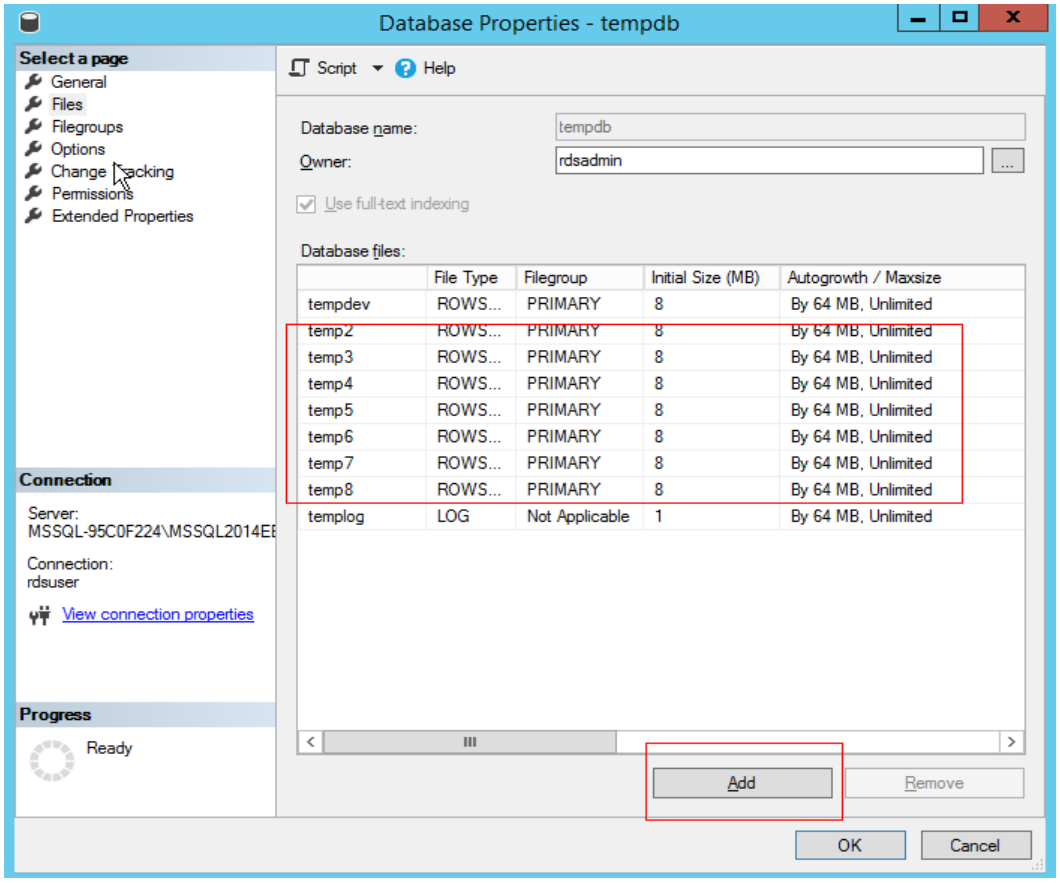
  GO

  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**.

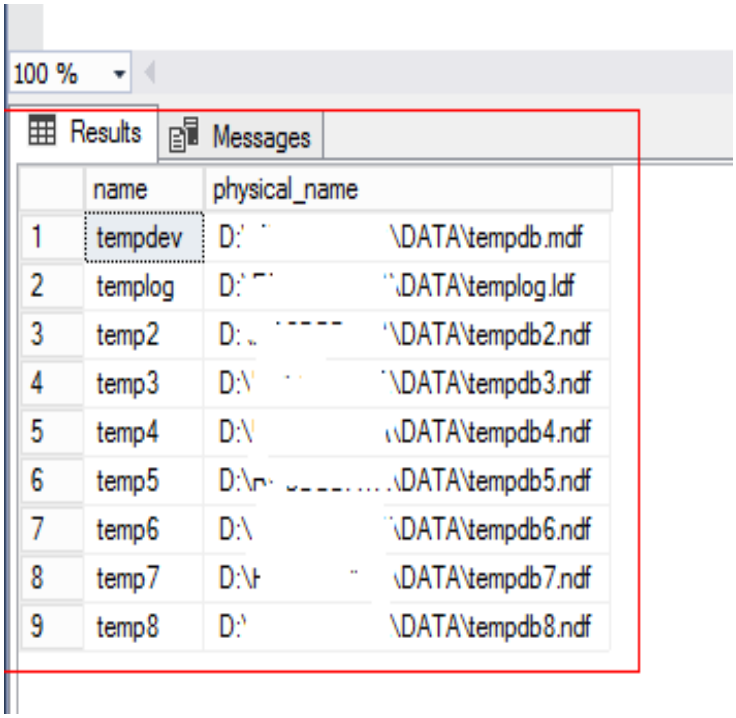
Figure 4-13 Adding tempdb files through the client



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



----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.

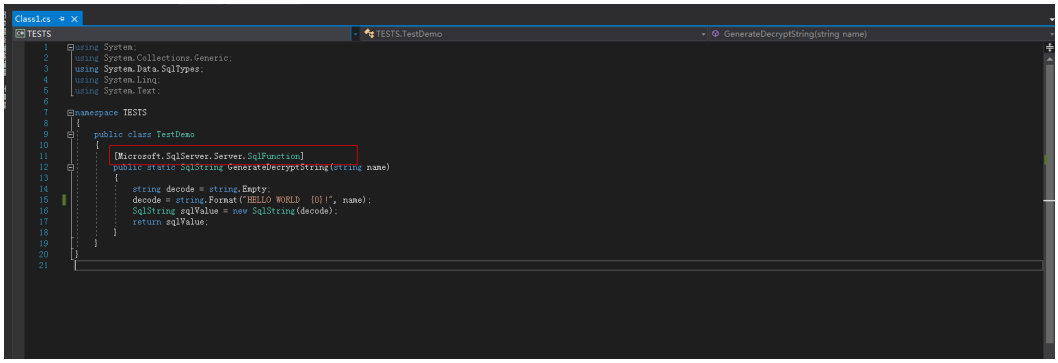
NOTE

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

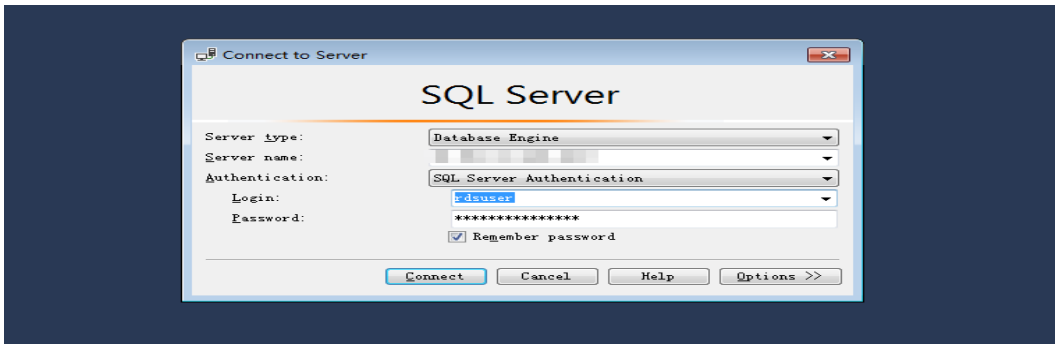


NOTICE

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



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

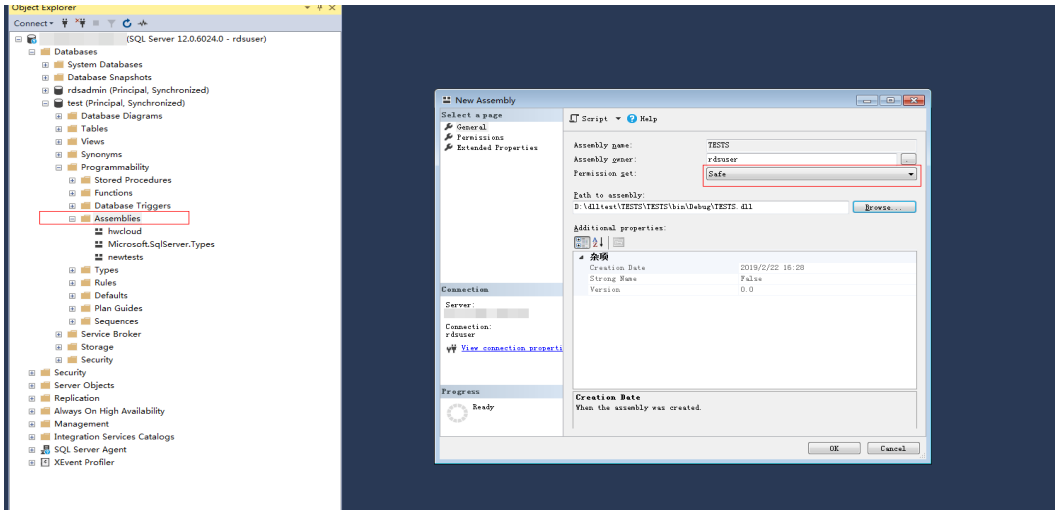


Figure 4-18 DLL file

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

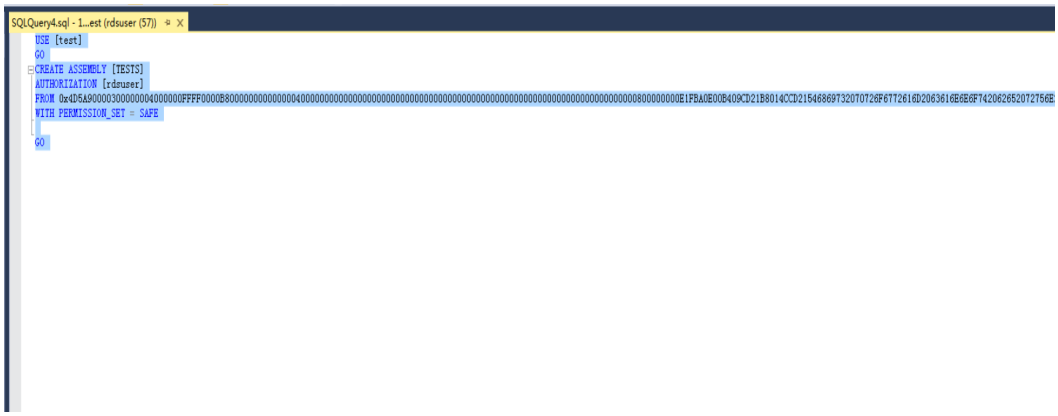
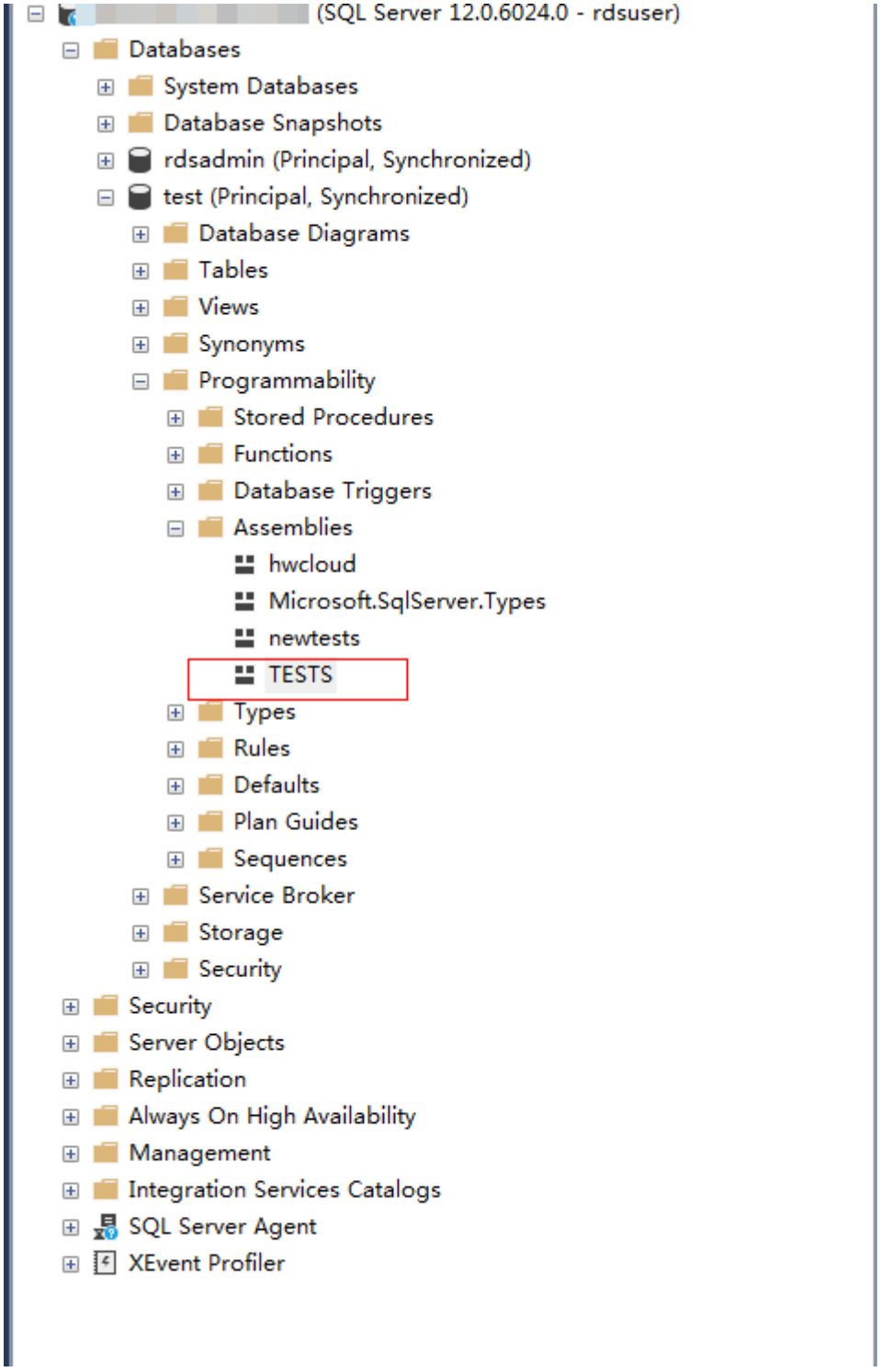


Figure 4-20 TESTS assembly



----End

4.10 Creating a Linked Server for an RDS 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**.

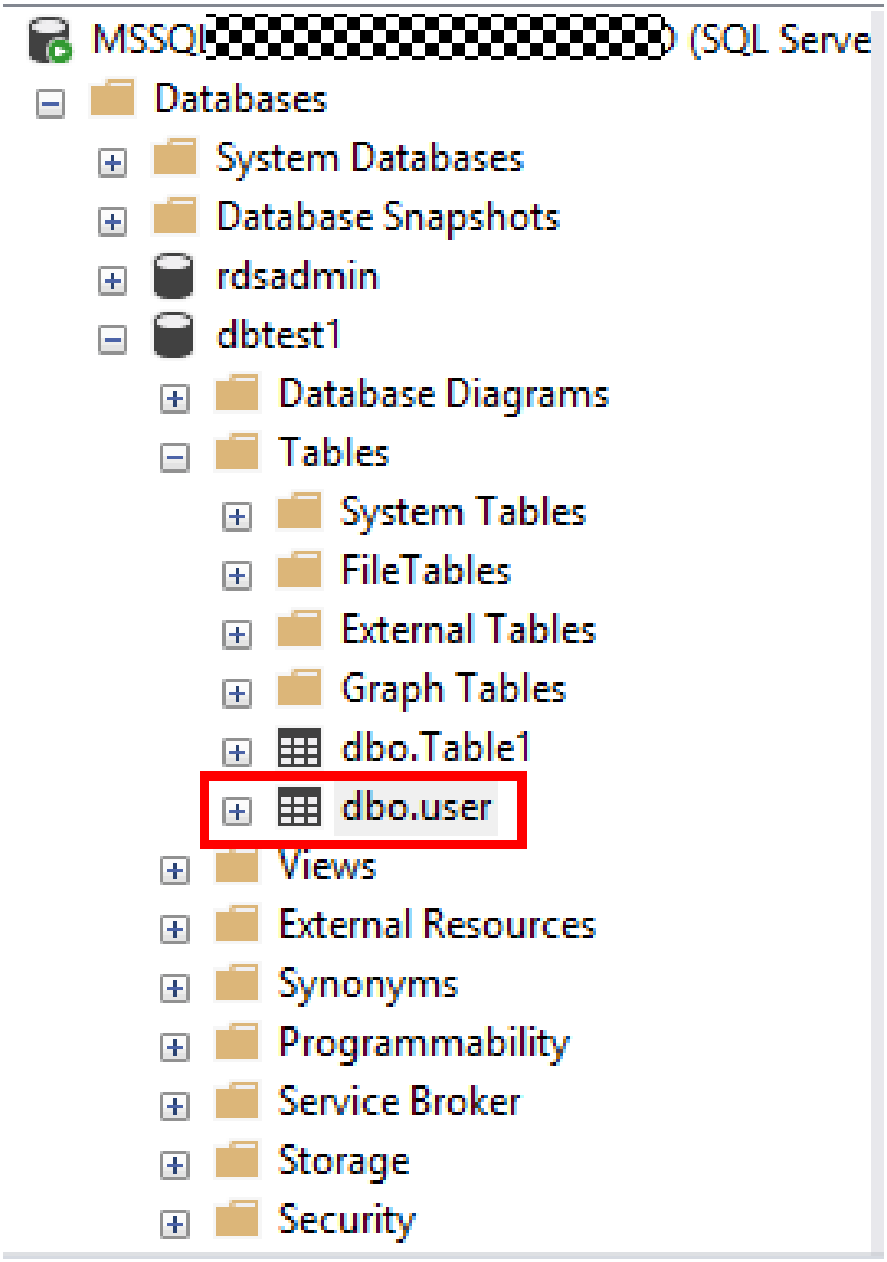
Step 3 In DB instance 2, run the following commands to create a linked server as user **rdsuser**.

```
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 4-4 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.

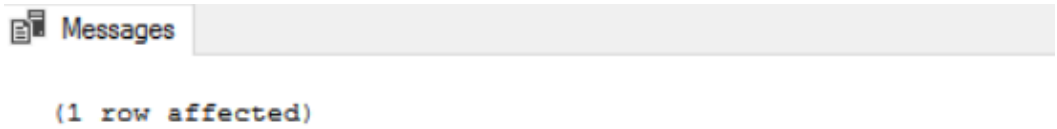
Step 4 After the DBLink is created, you can view the databases created in DB instance 1 on the linked server.



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



----End

4.11 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 256 pages.
  ```

- 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 4-5 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

Results | Messages

	DbId	FileId	CurrentSize	MinimumSize	UsedPages	EstimatedPages
1	1	1	688	512	512	512

Table 4-6 Result set parameter description

Column Name	Description
DbId	Database ID of the current shrink file.
FileId	File ID of the current shrink file.
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];

4.12 Using DAS to Create and Configure Agent Job and DBLink on the Master and Slave Databases 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 SQL Server databases, facilitating synchronization between primary and standby SQL Server databases.

Logging In to DAS

Step 1 Log in to the management console.

Step 2 Click **Service List**. Under **Database**, click **Relational Database Service**. The RDS console is displayed.

Step 3 On the **Instance Management** page, locate the target DB instance and click **Log In** in the **Operation** column.

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

Step 4 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 Operation** > **SQL Window** 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

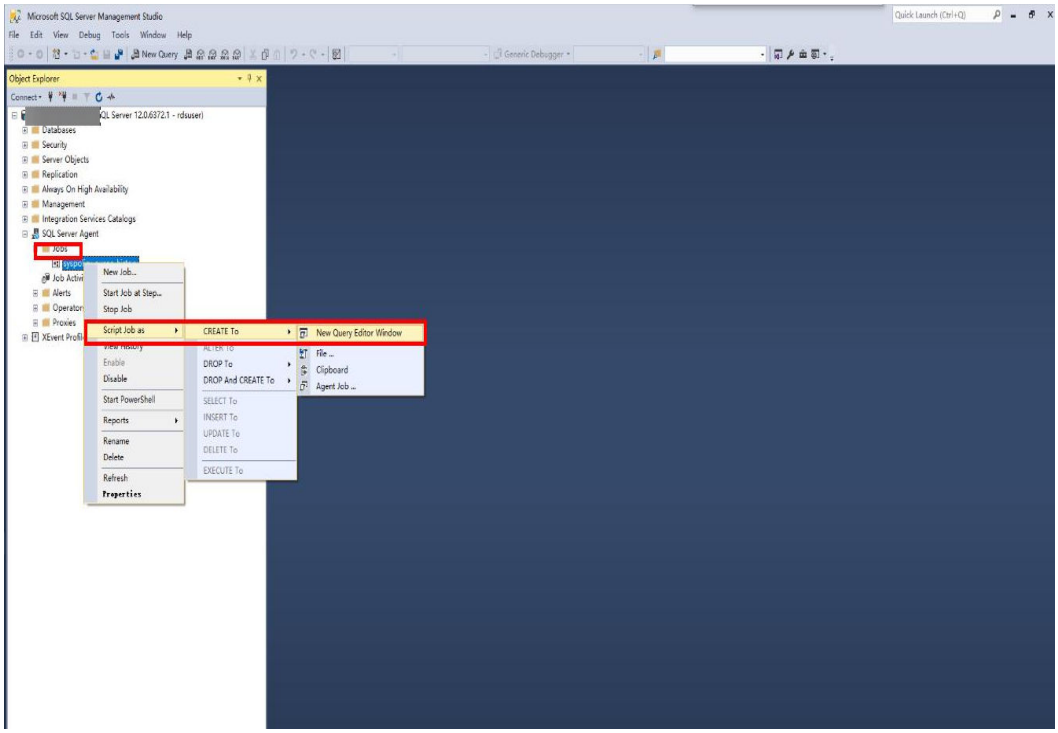
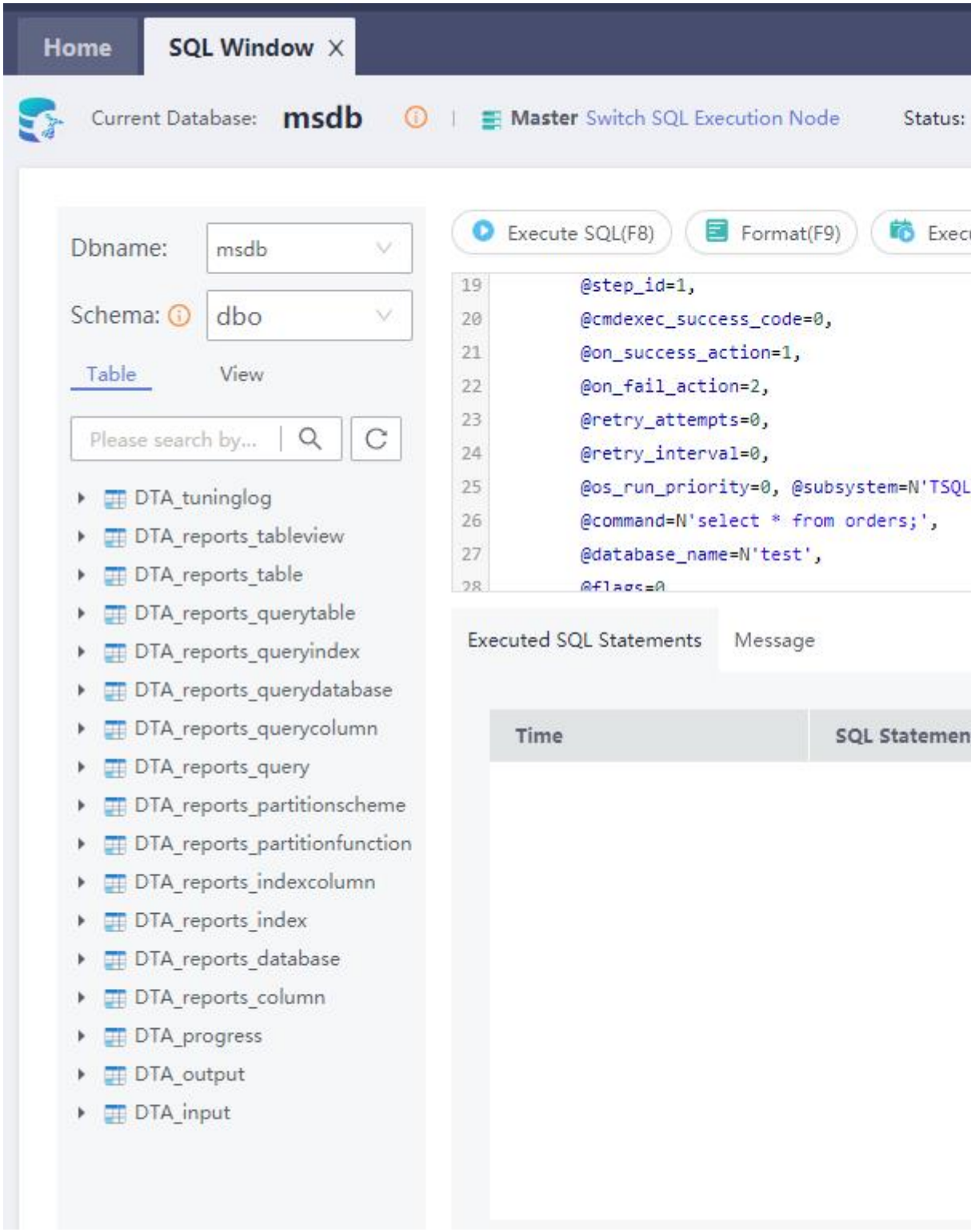


Figure 4-25 Using the DAS console to create a job on the slave database



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 SQL Server DB Instance**.

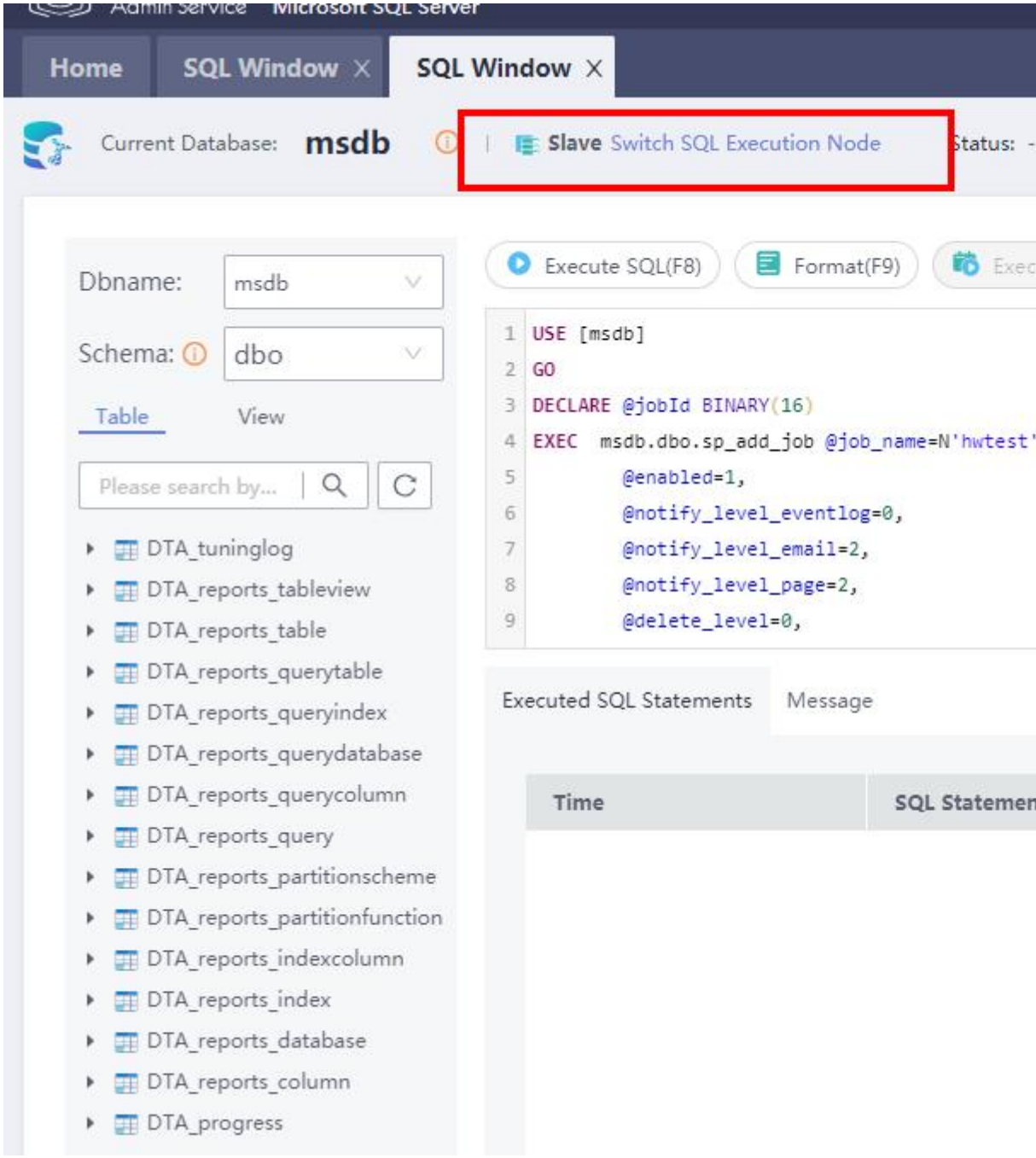
Step 1 Create a DBLink on the master database.

```
USE [master]

GO

EXEC master.dbo.sp_addlinkedserver @server = N'TEST',
@srvproduct=N'mytest', @provider=N'SQLOLEDB', @datasrc=N'xxxxxx'

EXEC master.dbo.sp_addlinkedsrvlogin @rmtsrvname = N'TEST', @locallogin =
NULL , @useself = N'False', @rmtuser = N'rdsuser', @rmtpassword = N'xxxxx'

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



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

